NoTVLA: Semantics-Preserving Robot Adaptation via Narrative Action Interfaces*

Zheng Huang* **Mingyu Liu*** **Xiaoyi Lin** **Muzhi Zhu** **Canyu Zhao** **Zongze Du**
Ye Lin **Xiaoman Li** **Yiduo Jia** **Hao Zhong** **Hao Chen†** **Chunhua Shen†**

State Key Lab of CAD & CG, Zhejiang University, China

Abstract

Robot fine-tuning can improve manipulation success while degrading the semantic structure inherited from large vision-language pretraining. This trade-off limits the reuse of vision-language-action (VLA) policies in settings that require compositional instructions, camera changes, and coordination with higher-level agents. We present **NoTVLA**, a semantics-preserving robot adaptation framework that replaces dense low-level action supervision with a sparse narrative action interface. NoTVLA converts demonstrations into sparse, semantically meaningful waypoints, grounds each decision with a task-relevant visual anchor and depth value, and reconstructs executable motion through a deterministic detokenizer. The resulting interface keeps the autoregressive vision-language backbone close to its pretrained prediction format while delegating high-frequency control to a transparent motion-rendering stage. We evaluate this design through matched and backbone-family comparisons, semantic retention probes, semantic out-of-distribution manipulation tasks, camera and depth perturbations, and deployment-oriented efficiency analysis. The results suggest that robot adaptation should be judged not only by task success, but also by how much task-relevant semantic competence is preserved during fine-tuning.

1 Introduction

Recent progress in vision-language-action (VLA) modeling suggests that large vision-language models (VLMs) provide useful priors for robotic manipulation [Driess et al., 2023, Zitkovich et al., 2023, Kim et al., 2024, Wang et al., 2025, Liu et al., 2025a]. A persistent tension remains, however: robot fine-tuning can improve task-specific execution while weakening the semantic structure that made the pretrained backbone useful. In practice, a model that initially supports compositional language and open-ended visual concepts may become substantially narrower after being optimized to imitate dense low-level trajectories. As VLM-based policies are increasingly expected to interact with general agent systems, this regression becomes a core limitation rather than a secondary side effect. A policy that succeeds only under a fixed camera setup and a narrow instruction distribution is difficult to reuse, compose, or invoke from higher-level planning systems.

The dominant response has been to strengthen the action side: larger policy heads, diffusion or flow-based decoders, richer control histories, and increasingly specialized robot-side modules [Team et al., 2024, Black et al., 2024, Chi et al., 2025]. These designs often improve short-horizon control, but they may also widen the gap between semantic reasoning and action generation. Even token-based VLAs are commonly trained to predict dense or near-dense low-level outputs whose structure differs from the narrative and compositional format of language. This mismatch changes the representational pressure placed on the backbone: dense robot supervision can pull the model toward embodiment-specific motor statistics. Recent work treats this preservation problem as a first-class question:

*ZH and ML contributed equally. HC and CS are the corresponding authors.

Conference on Neural Information Processing Systems (NeurIPS 2026).

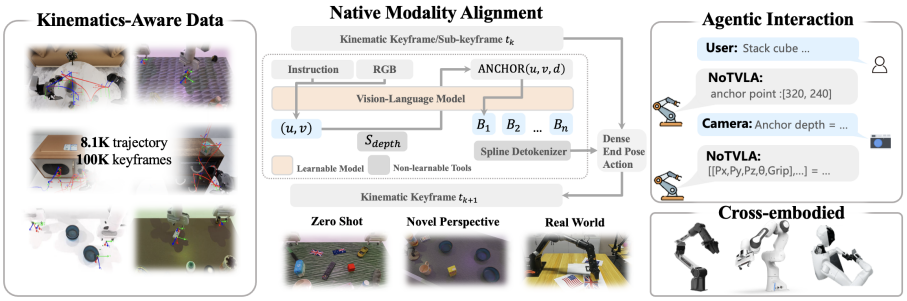


Figure 1: Conceptual overview of NoTVLA. Instead of directly supervising dense control trajectories, the model predicts a task-relevant anchor and a sparse narrative action sequence. A deterministic detokenizer converts these semantically meaningful waypoints into executable robot motion. This shifts adaptation pressure away from embodiment-specific motor statistics and toward the sparse narrative action interface. The bimanual robot depicted is the AgiBot G1 platform, which we evaluate on the AGIBOT challenge tasks reported in Appendix B.3; the remaining quantitative experiments use single-arm and bimanual (dual-arm) tabletop setups.

VLM2VLA rewrites low-level actions as language and relies on LoRA for adaptation, while MAPS constrains different VLM modules to remain close to their pretrained states through a module-wise proximity schedule [Hancock et al., 2025, Huang et al., 2025]. These methods motivate evaluating robot adaptation by semantic retention as well as execution success.

We take a complementary approach. Rather than attaching a stronger action head to a VLM, we ask whether robot fine-tuning can use a sparse narrative action interface that is closer to language and therefore potentially less disruptive to the pretrained semantic space. We call this framework **NoTVLA**: not a rejection of VLA modeling, but a test of whether robot adaptation must be centered on dense low-level action prediction. The goal is a semantics-preserving adaptation recipe that keeps the backbone in a narrative decision loop and delegates motion rendering to a lightweight deterministic stage.

NoTVLA is built around three design choices. First, robot demonstrations are compressed into sparse waypoints aligned with semantically meaningful events, such as contact formation, object lifting, and placement completion. Second, the model predicts these waypoints relative to a task-relevant 2D anchor augmented with depth, which preserves geometric grounding without requiring the backbone to regress dense metric trajectories at every step. Third, a deterministic detokenizer reconstructs smooth high-frequency motion from the sparse outputs, separating semantic decision-making from low-level trajectory rendering. In this respect, NoTVLA is closer to recent structured-interface VLAs such as CoT-VLA, HAMSTER, ACoT-VLA, SpatialVLA, and BridgeVLA than to purely dense action predictors. Its emphasis is different, however: the intermediate representation is introduced primarily as a *semantics-preserving adaptation target*, rather than only as a planner, spatial code, or action prior [Zhao et al., 2025, Li et al., 2025b, Zhong et al., 2026, Qu et al., 2025, Li et al., 2025a].

We evaluate NoTVLA as a *semantics-preserving robot adaptation interface*. The claim is intentionally narrower than solving modality mismatch in general. The study focuses on three questions:

1. **Semantic preservation.** Does replacing dense robot supervision with a narrative action interface reduce catastrophic forgetting in the underlying VLM?
2. **Generalization.** Is semantic preservation consistently accompanied by stronger performance on semantic inversion, unseen attribute composition, and camera or depth shifts?
3. **Practicality.** What are the efficiency, robustness, and deployment trade-offs of such an interface compared with stronger dense-action or specialized-policy alternatives?

We address these questions with matched-protocol comparisons centered on Qwen-PI and Qwen-OFT style baselines where available, backbone-family comparisons, semantic probing, zero-shot composition benchmarks, and deployment-oriented robustness analyses. Public benchmark numbers are used for context, while the semantic retention and representation ablations provide the primary

evidence for the mechanism. Additional protocol details, including comparison scope and backbone differences, are provided in the appendix.

Our main contributions are:

- We propose **NoTVLA**, a semantics-preserving robot adaptation framework that replaces dense low-level supervision with a sparse narrative action interface composed of semantic waypoints, anchor-conditioned depth grounding, and deterministic trajectory detokenization.
- We study *semantic preservation during robot adaptation* as an explicit evaluation target for VLAs, and provide evidence that the proposed interface retains task-relevant semantic competence more effectively than dense or flow-style robot fine-tuning under matched and backbone-family comparisons.
- We provide evidence that the sparse narrative action interface is associated with benefits beyond probe accuracy, including stronger semantic OOD manipulation, improved camera/depth robustness, and a separation between low-rate narrative replanning and high-rate continuous execution.

2 Related Work

VLA Fine-Tuning and Semantic Retention Large VLMs have become the foundation of modern embodied models, from PaLM-E and RT-style systems to open VLAs such as OpenVLA [Driess et al., 2023, Zitkovich et al., 2023, Kim et al., 2024]. A central challenge in this line of work is that robot adaptation does not simply add a new output head; it also changes the internal representation of the pretrained model. OpenVLA established an open reference point for action-token training and highlighted the fragility of semantic generalization under robot-only fine-tuning [Kim et al., 2024]. Recent work makes semantic retention an explicit objective. VLM2VLA / Actions-as-Language argues that catastrophic forgetting is rooted in a distribution mismatch between internet-scale VLM pretraining and low-level robot control, and addresses it by rewriting robot actions as natural-language strings so that lightweight LoRA adaptation can retain more VLM competence [Hancock et al., 2025]. MAPS keeps the conventional action supervision format but regularizes transfer at the optimization level, using module-wise proximity scheduling so that visual modules remain close to pretrained priors while action-oriented layers adapt more freely [Huang et al., 2025]. A complementary line of evidence comes from co-training studies: Lin et al. systematically examine how data modalities and co-training strategies shape large behavior models for manipulation, reinforcing that the form of supervision seen during robot adaptation strongly determines what the backbone retains [Lin et al., 2026]. NoTVLA is aligned with this trend but changes the intervention point: rather than verbalizing dense low-level controls or constraining adaptation only through optimization, it changes the supervision target into a sparse narrative action interface with explicit waypoints, anchor-conditioned depth grounding, and deterministic motion rendering.

Action Interfaces Beyond Dense Control Recent work explores multiple interfaces between semantic reasoning and low-level motion. Some methods discretize or chunk actions for autoregressive prediction [Zitkovich et al., 2023, Kim et al., 2024]; others use diffusion, flow matching, or latent policy heads to model continuous trajectories [Chi et al., 2025, Black et al., 2024]. Modular Qwen-based implementations, including StarVLA-α, make this comparison cleaner by instantiating chunked regression and flow-expert decoding under a shared backbone, enabling more controlled comparisons between action interfaces [Community, 2026, Ye et al., 2026]. A third direction focuses on structured intermediate representations, including traces, keyframes, subgoals, paths, and coarse action priors. Actions-as-Language, HAMSTER, and BridgeVLA are representative recent examples: they respectively rewrite low-level actions as language-format supervision, predict coarse 2D end-effector paths before low-level refinement, and align image-space cues with 3D manipulation targets [Hancock et al., 2025, Li et al., 2025b,a, Liu et al., 2026, Wang et al., 2026]. NoTVLA is closest to this broader structured-interface family, but differs in emphasis: the intermediate interface is treated primarily as a *semantics-preserving adaptation target*, not only as a more expressive action representation, path predictor, or spatial grounding module.

Trace, Planning, and Narrative Structure The abstraction of robot behavior into higher-level traces or plans appears in both classical robotics and recent multimodal foundation models. Hierarchical control decomposes planning and execution across time scales, while embodied foundation models

use trace-like signals, foresight, or latent plans to improve generalization. CoT-VLA introduces visual chain-of-thought reasoning by predicting future subgoal images before producing a short action sequence, making the intermediate representation explicitly visual [Zhao et al., 2025]. HAMSTER adopts a hierarchical split in which a high-level VLM proposes a coarse 2D end-effector path and a low-level 3D-aware controller refines it with richer geometry and proprioception [Li et al., 2025b]. This makes HAMSTER a close point of comparison for hierarchical structure, but its emphasis is open-world manipulation through a learned high/low-level split, whereas NoTVLA studies whether the high-level sequence itself can serve as the robot fine-tuning target. ACoT-VLA reasons directly in action space through coarse action intents, implemented with explicit and implicit action reasoners [Zhong et al., 2026]. TraceVLA, Magma, and related systems further suggest that trajectory-like structures can provide an effective interface between visual perception and downstream control [Li et al., 2025b, Yang et al., 2025b, **?**,a, Su et al., 2026]. RISE further couples the policy with a compositional world model and uses self-generated experience to improve high-level structure, which is complementary to our focus on what the fine-tuning target itself does to the pretrained backbone [Yang et al., 2026]. NoTVLA uses the notion of narrative more narrowly: the goal is to present robot supervision to the VLM in a format close to instruction-following and semantic disambiguation while delegating smooth motion rendering to a deterministic stage.

Spatial Grounding and Depth-Aware Manipulation Robotic manipulation requires mapping 2D observations to 3D action decisions. One strategy is to consume richer 3D inputs directly, such as point clouds, voxelized scenes, or ego-centric spatial encodings [Song et al., 2025, Qu et al., 2025, Liu et al., 2025b]. SpatialVLA is a representative example: it strengthens the observation side with Ego3D position encoding and redesigns the action side through adaptive action grids, emphasizing spatial representations and discretization rather than an explicit two-stage planner [Qu et al., 2025]. Another route is to remain image-centric but inject task-relevant geometry through keypoints, anchors, depth queries, or affordance maps. BridgeVLA follows this spirit by explicitly aligning VLM image-space outputs with 3D manipulation targets, making input-output alignment the main mechanism for efficient control [Li et al., 2025a]. Spatial-aware VLA pretraining instead injects spatial structure before robot adaptation, aligning visual and physical cues from human videos; such pretraining-time grounding is complementary to the anchor-depth grounding that NoTVLA applies at the fine-tuning interface [Feng et al., 2026]. NoTVLA also remains image-centric, but uses the 2D anchor and depth value as one component of the sparse narrative action interface rather than as the full contribution. This choice is not presented as a universal solution to spatial grounding, but as a practical compromise that lets us study semantic retention, camera robustness, and deployment flexibility under a lightweight action interface.

Inference Acceleration and Edge Deployment Recent work increasingly treats VLA latency as a systems problem rather than a fixed consequence of using large multimodal backbones. Deployment-focused VLA systems push this trend further: LiteVLA-Edge reports fully on-device multimodal control at 150.5 ms ($\sim$6.6 Hz) on Jetson AGX Orin through 4-bit quantization and a GPU-accelerated runtime, while NanoVLA argues for reducing latency through cached language computation, long-short action chunking, and dynamic routing [Williams et al., 2026, Chen et al., 2025a]. These systems are not direct method baselines for NoTVLA, but they matter for interpretation: a low narrative replanning rate in a current VLM stack should not be confused with an intrinsic limit of sparse narrative action interfaces. Our design is compatible with this deployment trend because high-level narrative inference is already decoupled from continuous execution.

3 Method

NoTVLA converts dense robot supervision into a sparse narrative action interface through three stages. Rather than adding a specialized policy head, the framework keeps the autoregressive VLM architecture unchanged and changes the prediction target. The stages are: (1) **trajectory sparsification**, which compresses demonstrations into semantically meaningful waypoints; (2) **anchor-conditioned narrative action generation**, which grounds those waypoints in image space and depth; and (3) **deterministic action detokenization**, which reconstructs executable motion without learning a second low-level controller.

3.1 Trajectory Sparsification

The first stage converts dense demonstrations into sparse waypoints that reflect semantic decision points rather than raw control frequency. Instead of supervising every low-level delta action, NoTVLA identifies a compact sequence of events such as contact formation, lift completion, and placement approach. This compression reduces the pressure on the VLM to model embodiment-specific motor rhythms.

Let a raw demonstration be a sequence of end-effector poses $\boldsymbol{T}(t) \in SE(3)$ with translation component $\boldsymbol{p}(t) \in \mathbb{R}^3$ and gripper states $G(t) \in \{0, 1\}$. We identify a set of keyframes $\{t_k\}$ based on kinematic transitions and contact events. A frame at t_k is selected if it satisfies:

$$\|\ddot{\boldsymbol{p}}(t_k)\|_2 > \alpha \quad \vee \quad \lim_{\epsilon \to 0^+} G(t_k - \epsilon) \neq \lim_{\epsilon \to 0^+} G(t_k + \epsilon), \tag{1}$$

where α is a task-dependent translational acceleration threshold. This rule retains decision-critical states while discarding redundant motion. To preserve local geometric context, NoTVLA optionally samples N sub-keyframes between major events, yielding a compact representation $\mathcal{T}_{sparse} = \{\boldsymbol{T}_0, \boldsymbol{T}_1, \ldots, \boldsymbol{T}_M\}$. The extracted sequence can be variable length; before tokenization, it is resampled to the fixed 5-point waypoint format used by the VLM interface, with endpoint keyframes retained and intermediate points sampled at uniform arc-length intervals.

We emphasize how the three stages relate at training versus inference time. Keyframe extraction is a training-time labeling procedure only: it converts demonstrations into sparse supervision targets and is never executed at deployment. At inference time, the model directly predicts the task-relevant anchor and the sparse waypoint tokens from the current observation; these predicted waypoints, not the training keyframes, are what the deterministic detokenizer interpolates into executable motion. The anchor is thus a per-decision grounding output, while keyframes only define what the model was trained to predict. The fixed waypoint budget $K{=}5$ is a single global default selected by a success–latency sweep over $K \in \{1, 2, 3, 5, 10\}$ (Appendix B.8), not a per-task hand-tuned value.

3.2 Anchor-Conditioned Narrative Action Generation

Directly regressing absolute 3D trajectories from RGB observations forces the model to absorb semantic intent and metric geometry into a single dense output space. NoTVLA instead factorizes grounding into a task-relevant 2D anchor and an associated depth value. This keeps the narrative action representation anchored to the image while avoiding dense 3D prediction at every control step.

Anchor Point Prediction (APP). Given an RGB image $\mathbf{I}$ and instruction $\mathbf{l}$, the model first predicts a 2D anchor point (u_a, v_a) representing the projected task locus (e.g., object center or contact point):

$$(u_a, v_a) = \mathcal{M}_{\text{enc}}(\mathbf{I}, \mathbf{l}), \quad (u_a, v_a) \in [0, W) \times [0, H). \tag{2}$$

The depth of this anchor, d_a, is retrieved from an external source $\mathcal{S}_{\text{depth}}$ (e.g., a depth sensor or monocular estimator), forming a depth-augmented anchor $a = (u_a, v_a, d_a)$.

Narrative Action Sequence. The action sequence is then generated autoregressively conditioned on this anchor:

$$\mathbf{S} = [\text{CLS}, \mathbf{I}_{\text{emb}}, \mathbf{l}_{\text{emb}}, \text{ANCHOR}(a), B_1, \ldots, B_M, \text{EOS}]. \tag{3}$$

Each action block B_i corresponds to a waypoint in $\mathcal{T}_{sparse}$ and contains discretized depth, image coordinates, gripper state, and rotation tokens:

$$B_i = (\mathcal{D}_i, \mathcal{U}_i, \mathcal{G}_i, \mathcal{R}_i). \tag{4}$$

The model is trained via standard next-token prediction, minimizing the negative log-likelihood over the action tokens:

$$\mathcal{L}_{\text{VLA}} = -\sum_{t=1}^{|\mathbf{S}|} \log P(s_t | s_{<t}, \mathbf{I}, \mathbf{l}). \tag{5}$$

This formulation keeps the backbone unchanged and turns robot adaptation into a next-token prediction problem over a sparse narrative action interface. The model is therefore optimized to predict *what should happen next* in a semantically meaningful sequence, while the continuous realization of that sequence is deferred to the final stage.

3.3 Deterministic Action Detokenization and Closed-Loop Correction

The final stage converts sparse narrative outputs into executable motion. NoTVLA uses a deterministic detokenizer rather than a learned low-level policy, allowing the method to separate semantic representation from motion rendering. This design makes the robustness trade-off explicit: improvements must come from the narrative representation and execution-time correction rather than from an additional learned controller.

Spline-based Reconstruction. The predicted sparse tokens $\{B_i\}$ are first de-quantized and projected into 3D waypoints $\{\boldsymbol{p}_{w,i}\}$ in the robot base frame using camera intrinsics $\mathbf{K}$ and extrinsics $\boldsymbol{T}_c^w$. Rotation tokens are de-quantized as Euler angles and converted to unit quaternions before interpolation. NoTVLA constructs a continuous-time trajectory $\hat{\boldsymbol{T}}(t)$ using cubic B-spline [Prautzsch et al., 2002] interpolation for translation and spherical linear interpolation (SLERP) [Shoemake, 1985] for rotation:

$$\begin{aligned} \hat{\boldsymbol{P}}(t) &= \text{Spline}(\{\boldsymbol{p}_{w,i}\}, t), \\ \hat{\boldsymbol{q}}(t) &= \text{SLERP}(\boldsymbol{q}_i, \boldsymbol{q}_{i+1}, \tau(t)). \end{aligned} \tag{6}$$

Here $\tau(t) \in [0, 1]$ is the normalized time within the active waypoint interval and uses the same monotone knot schedule as the translation spline. This formulation yields smooth interpolation between predicted waypoints and produces an explicit execution trace that can be inspected, constrained, or re-planned.

Closed-Loop Trajectory Merging. To support execution-time correction, NoTVLA performs asynchronous secondary inference during motion. After initiating a trajectory $\boldsymbol{T}_0(t)$ at t_0, the system triggers a new inference step. When updated waypoints $\mathcal{P}_{new}$ arrive at t_1, a merging mechanism blends the new plan with the currently executing trajectory at a future merge point $t_2 = t_1 + \Delta t$.

To prevent kinematic discontinuities, the new plan is aligned with the robot's current state. NoTVLA first identifies the optimal entry point in the new waypoint sequence $\mathcal{P}' = \{(u'_i, v'_i, d'_i)\}_{i=1}^{n'}$ relative to the robot's current end-effector position $\boldsymbol{p}_0(t_2)$, the translation component of $\boldsymbol{T}_0(t_2)$. The optimal index k^* is defined by minimizing the Euclidean distance in the robot base frame:

$$\begin{aligned} k^* = \arg\min_{1 \le i \le n'} \left\| \boldsymbol{p}'_{w,i} - \boldsymbol{p}_0(t_2) \right\|_2, \quad \boldsymbol{x}'_{c,i} = d'_i \, \mathbf{K}^{-1} [u'_i, v'_i, 1]^\top, \\ \begin{bmatrix} \boldsymbol{p}'_{w,i} \\ 1 \end{bmatrix} = \boldsymbol{T}_c^w \begin{bmatrix} \boldsymbol{x}'_{c,i} \\ 1 \end{bmatrix}. \end{aligned} \tag{7}$$

To ensure directional consistency and avoid backtracking, NoTVLA computes a local motion vector $\hat{\boldsymbol{v}}_{\text{path}}$ from the neighboring waypoints of k^* and performs a consistency test:

$$\hat{\boldsymbol{v}}_{\text{path}} = \frac{\boldsymbol{p}'_{w,k^+} - \boldsymbol{p}'_{w,k^-}}{\|\boldsymbol{p}'_{w,k^+} - \boldsymbol{p}'_{w,k^-}\|_2}, \quad \gamma = (\boldsymbol{p}'_{w,k^*} - \boldsymbol{p}_0(t_2))^\top \hat{\boldsymbol{v}}_{\text{path}}, \tag{8}$$

where $k^- = \max(1, k^* - 1)$ and $k^+ = \min(n', k^* + 1)$. The waypoint k^* is retained only if $\gamma > 0$. The final executable waypoint sequence discards stale points: $\mathcal{P}_{\text{pending}} = \{(u'_j, v'_j, d'_j) \mid j \ge k^* + \mathbb{1}[\gamma \le 0]\}$. A transition spline then connects $(\boldsymbol{T}_0(t_2), \dot{\boldsymbol{T}}_0(t_2))$ to the start of $\mathcal{P}_{\text{pending}}$, ensuring a smooth handoff. This closed-loop merging procedure enables replanning without requiring the VLM to operate at the actuator control rate.

4 Experiments

We evaluate NoTVLA along four axes: benchmark performance, semantic retention during robot fine-tuning, robustness of the narrative grounding interface, and real-robot deployment cost. Matched-protocol and backbone-family comparisons provide the main mechanism evidence; public-reference benchmarks and task-level diagnostics appear in Appendix B.

The comparisons are separated by claim strength. Matched-protocol and backbone-family comparisons support claims about the action interface, while public benchmark numbers provide external context rather than fully controlled head-to-head evidence. The central claim is not that NoTVLA outperforms every VLA implementation, but that a sparse narrative action interface better preserves

Table 1: Main benchmark comparison. RoboTwin reports clean/random settings using Easy/Hard naming; Qwen-OFT has Easy-only results. SimplerEnv uses public WidowX Visual Matching averages for context. NoTVLA and the baselines use different backbones on RoboTwin (see text).

Benchmark / Protocol	Method	Interface / Backbone	Success
RoboTwin 2.0, official 50-clean-trajectory setting (Easy / Hard)			
RoboTwin	π_0	VLA + flow expert [Black et al., 2024]	46.42% / 16.34%
RoboTwin	Qwen-OFT	Qwen3-VL-4B + chunked continuous actions [Community, 2026, Ye et al., 2026]	50.38% (Easy only)
RoboTwin	**NoTVLA (ours)**	Qwen2.5-VL-7B + narrative sparse action	**63.28% / 28.40%**
SimplerEnv, WidowX Visual Matching			
SimplerEnv	RT-1-X	large-scale RT-style robot transformer [Brohan et al., 2022]	1.1%
SimplerEnv	OpenVLA	open token VLA [Kim et al., 2024]	4.2%
SimplerEnv	Octo-Base	generalist robot policy [Team et al., 2024]	16.0%
SimplerEnv	π_0	VLA + flow expert [Black et al., 2024]	27.1%
SimplerEnv	π_0-FAST	tokenized π action variant [Black et al., 2024, Pertsch et al., 2025]	48.3%
SimplerEnv	SpatialVLA	PaliGemma2 + spatial tokens [Qu et al., 2025, Steiner et al., 2024]	42.7%
SimplerEnv	GR00T-N1.5	dual-system foundation policy [Bjorck et al., 2025]	61.9%
SimplerEnv	UniVLA	task-centric latent actions [Bu et al., 2025b]	45.6%
SimplerEnv	SoFar	language-grounded orientation [Qi et al., 2025]	58.3%
SimplerEnv	StarVLA-π	Qwen3-VL-4B + flow-matching action expert [Community, 2026, Ye et al., 2026]	60.9%
SimplerEnv	StarVLA-OFT	Qwen3-VL-4B + chunked continuous actions [Community, 2026, Ye et al., 2026]	64.6%
SimplerEnv	StarVLA-GR00T	Qwen3-VL-4B + dual-system action head [Community, 2026, Ye et al., 2026, Bjorck et al., 2025]	65.3%
SimplerEnv	**NoTVLA (ours)**	Qwen3-VL-4B + narrative sparse action	**65.7%**

task-grounded semantics and is associated with stronger semantic OOD manipulation under the tested settings. We report trial counts in the appendix. Key success rates are accompanied by Wilson 95% confidence intervals, and cross-task aggregates use stratified 95% CIs over task means (Appendix E.3). Multi-seed training of VLA-scale policies is computationally prohibitive, so we follow the single-checkpoint, multi-rollout convention and report evaluation-level uncertainty.

4.1 Benchmark Performance

Table 1 summarizes the main simulator evidence. RoboTwin is evaluated under the benchmark's shared protocol [Chen et al., 2025b] but, as discussed below, is not backbone-matched, while SimplerEnv provides public-reference context under the WidowX Visual Matching protocol [Li et al., 2024b]. Qwen-PI denotes the Qwen3-VL-4B flow-style policy interface [Bai et al., 2025a], and Qwen-OFT denotes the Qwen3-VL-4B chunked continuous-action interface. NoTVLA uses Qwen2.5-VL-7B for RoboTwin and Qwen3-VL-4B for SimplerEnv, so the RoboTwin aggregate is not by itself a pure backbone-controlled comparison. We therefore interpret this table together with the semantic retention, semantic OOD, and representation ablations below, where the evidence is closer to the action-interface question.

4.2 Semantic Retention and OOD Generalization

We first probe whether robot fine-tuning preserves task-grounded language competence. The probe contains 960 examples spanning target identification, attributes, ordering, choice-based placement, and task-stage recognition. This comparison uses the same Qwen3-VL-4B backbone family and evaluation prompt for Qwen-PI and NoTVLA, although the fine-tuning interfaces differ. Qwen-PI rapidly collapses on this probe, while NoTVLA retains 76% accuracy through the final checkpoint (Table 2). With Wilson 95% intervals over the 960 probe examples, the final-checkpoint gap is 76% [73, 79] versus 0% [0, 0.4], and the two curves' intervals are fully separated from Step 200 onward.

We do not treat the probe alone as causal evidence. It is not designed to reward the NoTVLA output format: the model answers task-grounded semantic questions under a shared prompt, decoding

Table 2: Task-grounded QA accuracy during robot fine-tuning.

Model	Step 0	Step 100	Step 200	Step 500	Step 1000	Step 4000
Qwen-PI	97%	68%	23%	0%	0%	0%
NoTVLA	97%	88%	79%	76%	76%	76%

setting, and rule-based scoring protocol. Instead, we use the probe as one part of a matched evidence chain: under comparable robot fine-tuning, the model that retains task-grounded semantics also performs better on semantic OOD manipulation. Table 3 evaluates inverse ordering, unseen color composition, choice-based placement, and novel concept transfer. Dense or flow-style baselines remain competitive on seen tasks, but NoTVLA has a substantially higher OOD average. Under the matched Qwen3-VL-4B protocol, the OOD average is 64.5% [60.0, 69.0] for NoTVLA-30k versus 25.5% [21.5, 29.5] for Qwen-PI-30k (stratified 95% CIs over task means), and the Wilson intervals of the two methods do not overlap in any of the four OOD categories, so the advantage is not driven by fluctuation in a single category.

Table 3: Semantic OOD and compositional generalization. OOD Avg. is the unweighted arithmetic mean of the four OOD categories.

Method	Seen Avg.	Inverse	Novel Color	Choice	Novel Concept	OOD Avg.
Qwen-PI-30k	61.5%	0.0%	35%	23%	44%	25.5%
Qwen-PI-5k	50.5%	0.0%	23%	45%	59%	31.8%
π_0-30k	56.5%	0.0%	30%	25%	32%	21.8%
π_0-5k	43.5%	0.0%	18%	56%	55%	32.3%
$\pi_{0.5}$-30k	64.0%	0.0%	48%	60%	57%	41.3%
$\pi_{0.5}$-5k	52.5%	1.0%	58%	59%	62%	45.0%
NoTVLA-30k	57.5%	63%	62%	49%	84%	64.5%
NoTVLA-5k	55.0%	70%	69%	62%	78%	69.8%

4.3 Anchor and Depth Grounding Ablation

NoTVLA grounds sparse narrative actions through an image anchor and its depth value. Table 4 isolates this design while keeping the policy family fixed.

Table 4: Ablation of anchor-conditioned depth grounding. All numbers are task success rates (%) over 100 trials per task-method setting.

Task	Full NoTVLA (APP + depth)	Direct UVD (w/o anchor)	APP only (no depth)
Object Recognition			
Place a2b left	**38%**	28%	30%
Place a2b right	**34%**	24%	32%
Place object scale	**43%**	39%	36%
Long-Horizon Tasks			
Blocks ranking size	**53%**	33%	28%
Blocks ranking rgb	**76%**	66%	59%
Stack block three	**49%**	32%	30%
Stack bowl three	**83%**	78%	68%

Full anchor-conditioned grounding improves the seven-task average from 42.9% for direct UVD and 40.4% for anchor-point-only prediction to 53.7%. With stratified 95% CIs over task means, the aggregates are 53.7% [50.3, 57.2] for full NoTVLA, 42.9% [39.5, 46.2] for direct UVD, and 40.4% [37.0, 43.9] for anchor-point-only prediction; the aggregate intervals do not overlap and the direction is consistent on all seven tasks, although we do not claim per-task significance for every cell. The gain is larger on the long-horizon tasks, which may reflect the repeated localization these tasks require. Table 5 shows that the 5-point deterministic detokenizer remains close to the stronger 10-point variant while keeping actions shorter.

Table 5: Compact trajectory-rendering diagnostics for the deterministic detokenizer. Full diagnostics are reported in Appendix B.6.

Metric	NoTVLA 5-pt	NoTVLA 10-pt	Magma	HAMSTER	LLARVA
Cover F1 ↑	0.9546	**0.9567**	0.9282	0.9451	0.9189
DTW ↓	0.4564	**0.4554**	0.8353	0.7945	0.4894
Endpoint error ↓	0.0739	**0.0729**	0.1784	0.1181	0.0771
LCSS similarity ↑	0.9660	**0.9671**	0.9562	0.9496	0.9447

Two further ablations attribute where the gains come from (Appendix B.7). First, disabling closed-loop trajectory merging changes the aggregate success over eight RoboTwin tasks (100 trials each) only from 72.8% to 71.6%, a 1.2pp difference within the confidence intervals. Merging is therefore an execution-continuity and smoothness component, not the source of the success-rate gains. Second, removing the subkeyframe trajectory cropping used to construct sparse narrative supervision drops stack cube from 78.0% [75.3, 80.5] to 23.0% [20.5, 25.7] and push cube from 99.0% [98.1, 99.5] to 94.0% [92.3, 95.3] on the ManiSkill3 [Tao et al., 2024] stack cube and push cube tasks under an InternVL2.5-1B backbone [Chen et al., 2024], suggesting that the training-time trajectory decomposition of the narrative interface acts as a form of data augmentation; the effect appears with a different backbone family and simulator than the main experiments.

4.4 Camera and Depth Stress Tests

We also test a precision card-grasping task. The hard split fails if the card contacts the groove boundary, making it more sensitive to depth error than ordinary grasp success.

The hard split exposes the difference between direct depth-token prediction and anchor-conditioned depth use: direct UVD drops to 16%, while full NoTVLA reaches 56%. On this depth-sensitive hard split, the Wilson intervals are fully separated: 56% [46, 65] for NoTVLA versus 26% [18, 35] for Qwen-PI and 16% [10, 24] for direct UVD. With injected depth-anchor noise, average success changes from 42% at 0 cm to 30% at 1 cm, 11% at 10 cm, and 7% at 100 cm; under mild fisheye distortion, anchor-depth grounding averages 27.8% versus 19.8% for direct UVD (stratified aggregates 27.9% [24.7, 31.0] versus 20.0% [17.1, 22.9]) (Appendix B.5). These stress tests clarify the failure mode: NoTVLA improves robustness relative to direct UVD prediction, but still needs reliable local geometry and degrades sharply under large depth corruption.

A finer-grained stress sweep and a temporal analysis of a real RealSense D435if sensor (Appendix B.5) further locate the boundary of this dependence. Injected noise of 0.5 cm, a conservative upper bound on real per-frame sensor noise, is indistinguishable from the clean baseline (35.0% [24.2, 47.6] versus 31.7% [21.3, 44.2] under the 60-trial stress protocol), and success then declines monotonically, separating from the baseline at 10 cm. Measured D435if temporal noise is millimeter-level within the 0.3–1.2 m working range and is further reduced by 3×3 ROI-median and EMA temporal filtering, whereas the dominant residual error is a systematic bias (e.g., $+25.2$ mm at 1.2 m) that filtering cannot remove: temporal filtering improves precision, while calibration is required for accuracy. The practical failure mode is therefore miscalibration and structural depth errors rather than ordinary frame-level sensor noise.

Table 6: Depth-sensitive grasping. All numbers are success rates (%). Simulated settings use 100 trials; real paper drawing uses 20 trials. DP3 refers to 3D Diffusion Policy [Ze et al., 2024].

Method	Sim Card Easy	Sim Card Hard	Real Draw Paper
DP3	67%	18%	20%
Qwen-PI	**80%**	26%	40%
NoTVLA w/ direct UVD	73%	16%	35%
NoTVLA	75%	**56%**	**45%**

4.5 Deployment Cost and Real-Robot Results

NoTVLA separates low-rate narrative replanning from high-rate execution. In our current implementation, narrative plans are refreshed at 0.25–0.5 Hz (a 2–4 s high-level replanning period) with roughly 13 GB peak inference memory, while deterministic interpolation executes at 80 Hz. High-level latency therefore affects how often the policy rethinks, not the actuator control frequency. We view this replanning rate as a systems bottleneck rather than an intrinsic limit of the interface.

The roughly 50-token, 5-waypoint narrative format is a measured operating point rather than an arbitrary budget. Sweeping the waypoint count $K \in \{1, 2, 3, 5, 10\}$ on the ManiSkill stack cube task yields 0%/0%/74%/98%/98% success at approximately 1.0/1.4/1.9/2.8/5.0 s per decision (Appendix B.8): $K \leq 2$ cannot express the approach–grasp–lift–place structure of the tasks, $K{=}3$ trails $K{=}5$ by 24pp, and $K{=}10$ brings no success gain while nearly doubling latency. The 0.25–0.5 Hz semantic refresh rate remains a limitation for dynamic scenes with moving targets: closed-loop merging corrects execution error in quasi-static scenes, but all tasks evaluated here are quasi-static, and we scope our claims accordingly.

We deploy the system on five real-robot tasks: pressing a button, placing a cube on a flag, stacking two cubes, inserting a flower, and picking fruit into a bowl. This limited study is a deployment sanity check rather than primary evidence for broad real-world generalization. The flower task fails if the flower falls; other tasks follow Appendix E.7. The aggregate and task-level results, together with a closed-loop recovery example, are reported in Appendix B.9 (Tables 24–25 and Figure 2).

Despite slower high-level inference, NoTVLA has comparable successful-trial execution time and the highest average real-robot success. With only 20 trials per task-method pair, the Wilson intervals of these real-robot results are wide and overlap across methods; we therefore treat them strictly as a deployment sanity check showing that the interface runs on hardware with trends consistent with simulation, not as evidence of a statistically significant real-world advantage. The real-robot suite deliberately avoids long-horizon sequences; long-horizon evidence comes from the multi-step simulation tasks, and real-world validation is left to future work. This supports treating narrative replanning rate and actuator control rate as separate deployment quantities.

5 Conclusion

NoTVLA studies sparse narrative actions as a semantics-preserving robot adaptation interface rather than a generic solution to action-language mismatch. Across matched comparisons, semantic retention probes, semantic OOD manipulation, grounding ablations, and deployment diagnostics, the interface retains more task-relevant semantic competence and is associated with stronger generalization beyond dense demonstration distributions. These findings motivate evaluating robot adaptation by semantic retention alongside execution success; the decoupled interface can also benefit directly from faster multimodal inference without changing its high-rate execution stage.

Limitations. NoTVLA requires a reasonably calibrated RGB-D source: filtering suppresses random depth noise but not systematic bias, and some platforms lack usable depth [Chi et al., 2024, Ha et al., 2024]. Deterministic rendering is not suited to contact-rich or dynamic manipulation such as flinging and tossing [Ha and Song, 2022]. The current 0.25–0.5 Hz semantic replanning rate limits moving-target reactivity, while mobile cameras or bases would require unvalidated online extrinsic estimation [Xiong et al., 2025, Xu et al., 2026]. Finally, the real-robot study is small ($n{=}20$ per task-method pair) and short-horizon; it is a deployment sanity check rather than evidence of broad real-world generalization.

Acknowledgments and Disclosure of Funding

HC's participation was in part supported by the National Natural Science Foundation of China (No. 62576315) and the Hangzhou National AI Application Pilot Base (Embodied Intelligence). Part of this work was done when ZH and CS were with Ant Group. The authors declare no competing interests.

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, 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, 2025a. URL `https://arxiv.org/abs/2511.21631`.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T-N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

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

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. RT-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025a.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. UniVLA: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025b.

Jiahong Chen, Jing Wang, Long Chen, Chuwei Cai, and Jinghui Lu. Nanovla: Routing decoupled vision-language understanding for nano-sized generalist robotic policies. *arXiv preprint arXiv:2510.25122*, 2025a.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025b.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024.

Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing. *arXiv preprint arXiv:2604.05014*, 2026.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, et al. PaLM-E: An embodied multimodal language model. *International Conference on Machine Learning*, 2023.

Yicheng Feng, Wanpeng Zhang, Ye Wang, Hao Luo, Haoqi Yuan, Sipeng Zheng, and Zongqing Lu. Spatial-aware VLA pretraining through visual-physical alignment from human videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 712–723, 2026.

Huy Ha and Shuran Song. FlingBot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding. In *Conference on Robot Learning*, pages 24–33. PMLR, 2022.

Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. UMI on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers. *arXiv preprint arXiv:2407.10353*, 2024.

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

Chengyue Huang, Mellon M Zhang, Robert Azarcon, Glen Chou, and Zsolt Kira. Maps: Preserving vision-language representations via module-wise proximity scheduling for better vision-language-action generalization. *arXiv preprint arXiv:2511.19878*, 2025.

Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel realsense stereoscopic depth cameras. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–10, 2017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Peiyan Li, Yixiang Chen, Hongtao Wu, Xiao Ma, Xiangnan Wu, Yan Huang, Liang Wang, Tao Kong, and Tieniu Tan. Bridgevla: Input-output alignment for efficient 3d manipulation learning with vision-language models. *arXiv preprint arXiv:2506.07961*, 2025a.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. CogACT: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024a.

Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024b.

Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Raymond Yu, Caelan Reed Garrett, Fabio Ramos, Dieter Fox, Anqi Li, et al. HAMSTER: Hierarchical action models for open-world robot manipulation. *arXiv preprint arXiv:2502.05485*, 2025b.

Fanqi Lin, Kushal Arora, Jean Mercat, Haruki Nishimura, Paarth Shah, Chen Xu, Mengchao Zhang, Mark Zolotas, Maya Angeles, Owen Pfannenstiehl, Andrew Beaulieu, and Jose Barreiros. A systematic study of data modalities and strategies for co-training large behavior models for robot manipulation. *arXiv preprint arXiv:2602.01067*, 2026.

Mingyu Liu, Zheng Huang, Xiaoyi Lin, Muzhi Zhu, Canyu Zhao, Yating Wang, Haoyi Zhu, Hao Chen, and Chunhua Shen. Gae: Unleashing physical potential of vlm with generalizable action expert. In *Forty-third International Conference on Machine Learning*, 2025a.

Mingyu Liu, Zeju Li, Jiuhe Shu, Hanqing Wang, Yuhao Chao, Hao Chen, and Chunhua Shen. Perfect demo makes poor teacher: Learning robust alignment from critical motion segments. *arXiv preprint arXiv:2606.15587*, 2026.

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. *arXiv preprint arXiv:2410.07864*, 2024.

Yang Liu, Ming Ma, Xiaomin Yu, Pengxiang Ding, Han Zhao, Mingyang Sun, Siteng Huang, and Donglin Wang. Ssr: Enhancing depth perception in vision-language models via rationale-guided spatial reasoning. *arXiv preprint arXiv:2505.12448*, 2025b.

Dantong Niu, Yuvan Sharma, Giscard Biamby, Jerome Quenum, Yutong Bai, Baifeng Shi, Trevor Darrell, and Roei Herzig. LLARVA: Vision-action instruction tuning enhances robot learning. *arXiv preprint arXiv:2406.11815*, 2024.

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

Hartmut Prautzsch, Wolfgang Boehm, and Marco Paluszny. *Bézier and B-spline techniques*. Springer Science & Business Media, 2002.

Zekun Qi, Wenyao Zhang, Yufei Ding, Runpei Dong, Xinqiang Yu, Jingwen Li, Lingyun Xu, Baoyu Li, Xialin He, Guofan Fan, Jiazhao Zhang, Jiawei He, Jiayuan Gu, Xin Jin, Kaisheng Ma, Zhizheng Zhang, He Wang, and Li Yi. Sofar: Language-grounded orientation bridges spatial reasoning and object manipulation. *CoRR*, abs/2502.13143, 2025. doi: 10.48550/ARXIV.2502.13143. URL `https://doi.org/10.48550/arXiv.2502.13143`.

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. *arXiv preprint arXiv:2501.15830*, 2025.

Claude Ambrose Rogers. *Hausdorff measures*. Cambridge University Press, 1998.

Pavel Senin. Dynamic time warping algorithm review. *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA*, 855(1-23):40, 2008.

Ken Shoemake. Animating rotation with quaternion curves. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 245–254, 1985.

Wenxuan Song, Ziyang Zhou, Han Zhao, Jiayi Chen, Pengxiang Ding, Haodong Yan, Yuxin Huang, Feilong Tang, Donglin Wang, and Haoang Li. ReconVLA: Reconstructive vision-language-action model as effective robot perceiver. *arXiv preprint arXiv:2508.10333*, 2025.

Andreas Steiner, André Susano Pinto, Michael Tschannen, Daniel Keysers, Xiao Wang, Yonatan Bitton, Alexey Gritsenko, Matthias Minderer, Anthony Sherbondy, Shangbang Long, et al. PaliGemma 2: A family of versatile VLMs for transfer. *arXiv preprint arXiv:2412.03555*, 2024.

Yue Su, Sijin Chen, Haixin Shi, Mingyu Liu, Zhengshen Zhang, Ningyuan Huang, Weiheng Zhong, Zhengbang Zhu, Yuxiao Liu, and Xihui Liu. World guidance: World modeling in condition space for action generation. *arXiv preprint arXiv:2602.22010*, 2026.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh Nagaswamy Rajesh, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. ManiSkill3: GPU parallelized robotics simulation and rendering for generalizable embodied AI. *arXiv preprint arXiv:2410.00425*, 2024.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Kaijun Wang, Liqin Lu, Mingyu Liu, Jianuo Jiang, Zeju Li, Bolin Zhang, Wancai Zheng, Xinyi Yu, Hao Chen, and Chunhua Shen. Odyssey: Open-world quadrupeds exploration and manipulation for long-horizon tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 18602–18610, 2026.

Yating Wang, Haoyi Zhu, Mingyu Liu, Jiange Yang, Hao-Shu Fang, and Tong He. Vq-vla: Improving vision-language-action models via scaling vector-quantized action tokenizers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11089–11099, 2025.

Justin Williams, Kishor Datta Gupta, Roy George, and Mrinmoy Sarkar. Litevla-edge: Quantized on-device multimodal control for embedded robotics. *arXiv preprint arXiv:2603.03380*, 2026.

Haoyu Xiong, Xiaomeng Xu, Jimmy Wu, Yifan Hou, Jeannette Bohg, and Shuran Song. Vision in action: Learning active perception from human demonstrations. *arXiv preprint arXiv:2506.15666*, 2025.

Xiaomeng Xu, Jisang Park, Han Zhang, Eric Cousineau, Aditya Bhat, Jose Barreiros, Dian Wang, Jeannette Bohg, and Shuran Song. HoMMI: Learning whole-body mobile manipulation from human demonstrations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2026.

Jiange Yang, Yansong Shi, Haoyi Zhu, Mingyu Liu, Kaijing Ma, Yating Wang, Gangshan Wu, Tong He, and Limin Wang. Como: Learning continuous latent motion from internet videos for scalable robot learning. *arXiv preprint arXiv:2505.17006*, 2025a.

Jianwei Yang, Reuben Tan, Qianhui Wu, Ruijie Zheng, Baolin Peng, Yongyuan Liang, Yu Gu, Mu Cai, Seonghyeon Ye, Joel Jang, et al. Magma: A foundation model for multimodal ai agents. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 14203–14214, 2025b.

Jiazhi Yang, Kunyang Lin, Wencong Zhang, Jinwei Li, Tianwei Lin, Longyan Wu, Ya-Qin Zhang, Hao Zhao, Ping Luo, Zhizhong Su, Hongyang Li, Xiangyu Yue, and Li Chen. Self-improving robot policy with compositional world model. In *Proceedings of Robotics: Science and Systems (RSS)*, 2026.

Jinhui Ye, Ning Gao, Senqiao Yang, Jinliang Zheng, Zixuan Wang, Yuxin Chen, Pengguang Chen, Yilun Chen, Shu Liu, and Jiaya Jia. StarVLA-α: Reducing complexity in vision-language-action systems. *arXiv preprint arXiv:2604.11757*, 2026.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D diffusion policy: Generalizable visuomotor policy learning via simple 3D representations. *arXiv preprint arXiv:2403.03954*, 2024.

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

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Linqing Zhong, Yi Liu, Yifei Wei, Ziyu Xiong, Maoqing Yao, Si Liu, and Guanghui Ren. Acot-vla: Action chain-of-thought for vision-language-action models. *arXiv preprint arXiv:2601.11404*, 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*, pages 2165–2183. PMLR, 2023.

A Appendix Overview

This appendix mirrors the main-paper evidence chain. Because the NoTVLA evaluation is centered on RoboTwin and SimplerEnv, we prioritize those two simulators in the appendix tables as well. We also include depth, camera, ablation, and real-robot diagnostics that support the main experimental discussion.

B Additional Experimental Results

B.1 Full RoboTwin Results

Table 7: Public RoboTwin 2.0 **Easy** task-level results [Chen et al., 2025b]. To reduce width, the RoboTwin appendix is split into separate Easy and Hard tables. The Qwen-OFT column reports the Easy-only run available in this setting; its average comes from the full run snapshot. Baseline families include RDT [Liu et al., 2024], ACT [Zhao et al., 2023], DP3 [Ze et al., 2024], and Qwen-style StarVLA-α variants [Community, 2026, Ye et al., 2026]. Rows with dashes indicate entries unavailable for that method. Public baseline and NoTVLA averages use the 50-task sheet; Qwen-OFT reports the available Easy-only run.

Task	**RDT**	π_0	$\pi_{0.5}$	**ACT**	**DP**	**DP3**	**Qwen-OFT**	**NoTVLA**
Adjust Bottle	81%	90%	97%	97%	97%	99%	96%	93%
Beat Block Hammer	77%	43%	3%	56%	42%	72%	58%	93%
Blocks Ranking RGB	3%	19%	63%	1%	0%	3%	45%	76%
Blocks Ranking Size	0%	7%	33%	0%	1%	2%	27%	49%
Click Alarmclock	61%	63%	22%	32%	61%	77%	91%	78%
Click Bell	80%	44%	6%	58%	54%	90%	94%	94%
Dump Bin Bigbin	64%	83%	97%	68%	49%	85%	68%	23%
Grab Roller	74%	96%	100%	94%	98%	98%	93%	97%
Handover Block	45%	45%	42%	42%	10%	70%	0%	55%
Handover Mic	90%	98%	37%	85%	53%	100%	39%	99%
Hanging Mug	23%	11%	19%	7%	8%	17%	15%	15%
Lift Pot	72%	84%	0%	88%	39%	97%	0%	95%
Move Can Pot	25%	58%	59%	22%	39%	70%	50%	76%
Move Pillbottle Pad	8%	21%	66%	0%	1%	41%	54%	12%
Move Playingcard Away	43%	53%	87%	36%	47%	68%	69%	77%
Move Stapler Pad	2%	0%	8%	0%	1%	12%	12%	30%
Open Laptop	59%	85%	14%	56%	49%	82%	31%	62%
Open Microwave	37%	80%	18%	86%	5%	61%	—	34%
Pick Diverse Bottles	2%	27%	83%	7%	6%	52%	30%	77%
Pick Dual Bottles	42%	57%	86%	31%	24%	60%	43%	88%
Place A2B Left	3%	31%	64%	1%	2%	46%	20%	38%
Place A2B Right	1%	27%	59%	0%	13%	49%	22%	34%
Place Bread Basket	10%	17%	60%	6%	14%	26%	52%	63%
Place Bread Skillet	5%	23%	59%	7%	11%	19%	56%	67%
Place Burger Fries	50%	80%	66%	49%	72%	72%	96%	69%
Place Can Basket	19%	41%	53%	1%	18%	67%	63%	77%
Place Cans Plasticbox	6%	34%	28%	16%	40%	48%	81%	51%
Place Container Plate	78%	88%	90%	72%	41%	86%	99%	92%
Place Dual Shoes	4%	15%	46%	9%	8%	13%	28%	21%
Place Empty Cup	56%	37%	96%	61%	37%	65%	72%	90%
Place Fan	12%	20%	45%	1%	3%	36%	28%	30%
Place Mouse Pad	1%	7%	40%	0%	0%	4%	9%	23%
Place Object Basket	33%	16%	67%	15%	15%	65%	40%	76%
Place Object Scale	1%	10%	73%	0%	1%	15%	19%	43%
Place Object Stand	15%	36%	80%	1%	22%	60%	48%	75%
Place Phone Stand	15%	35%	54%	2%	13%	44%	24%	28%
Place Shoe	35%	28%	77%	5%	23%	58%	63%	59%
Press Stapler	41%	62%	61%	31%	6%	69%	60%	94%
Put Bottles Dustbin	21%	54%	63%	27%	22%	60%	—	55%
Put Object Cabinet	33%	68%	34%	15%	42%	72%	35%	84%
Rotate QRcode	50%	68%	80%	1%	13%	74%	50%	73%
Scan Object	4%	18%	40%	2%	9%	31%	13%	24%
Shake Bottle Horizontally	84%	99%	100%	63%	59%	100%	98%	96%
Shake Bottle	74%	97%	100%	74%	65%	98%	98%	94%
Stack Blocks Three	2%	17%	54%	0%	0%	1%	41%	49%
Stack Blocks Two	21%	42%	85%	25%	7%	24%	83%	88%
Stack Bowls Three	51%	66%	70%	48%	63%	57%	62%	83%
Stack Bowls Two	76%	91%	93%	82%	61%	83%	90%	96%
Stamp Seal	1%	3%	48%	2%	2%	18%	27%	36%
Turn Switch	35%	27%	25%	5%	36%	46%	26%	33%
Average	34.50%	46.42%	57.00%	29.74%	28.04%	55.24%	50.38%	63.28%

Table 8: Public RoboTwin 2.0 **Hard** task-level results [Chen et al., 2025b]. To reduce width, the RoboTwin appendix is split into separate Easy and Hard tables. The $\pi_{0.5}$ column is added from the public `pi0.5` release. Rows with dashes indicate entries unavailable for that method. Public baseline and NoTVLA averages use the 50-task sheet.

Task	**RDT**	π_0	$\pi_{0.5}$	**ACT**	**DP**	**DP3**	**NoTVLA**
Adjust Bottle	75%	56%	54%	23%	0%	3%	56%
Beat Block Hammer	37%	21%	0%	3%	0%	8%	18%
Blocks Ranking RGB	0%	5%	8%	0%	0%	0%	12%
Blocks Ranking Size	0%	1%	11%	0%	0%	0%	26%
Click Alarmclock	12%	11%	62%	4%	5%	14%	27%
Click Bell	9%	3%	85%	3%	0%	0%	33%
Dump Bin Bigbin	32%	24%	30%	1%	0%	53%	12%
Grab Roller	43%	80%	63%	25%	0%	2%	36%
Handover Block	14%	8%	0%	0%	0%	0%	42%
Handover Mic	31%	13%	2%	0%	0%	3%	27%
Hanging Mug	16%	3%	1%	0%	0%	1%	11%
Lift Pot	9%	36%	0%	0%	0%	0%	49%
Move Can Pot	12%	21%	0%	4%	0%	6%	58%
Move Pillbottle Pad	0%	1%	20%	0%	0%	0%	2%
Move Playingcard Away	11%	22%	14%	0%	0%	3%	37%
Move Stapler Pad	0%	2%	4%	0%	0%	0%	0%
Open Laptop	32%	46%	3%	0%	0%	7%	12%
Open Microwave	20%	50%	14%	0%	0%	22%	23%
Pick Diverse Bottles	0%	6%	14%	0%	0%	1%	34%
Pick Dual Bottles	13%	12%	20%	0%	0%	1%	25%
Place A2B Left	1%	1%	12%	0%	0%	2%	18%
Place A2B Right	1%	6%	6%	0%	0%	0%	13%
Place Bread Basket	2%	4%	38%	0%	0%	1%	23%
Place Bread Skillet	1%	1%	19%	0%	0%	0%	34%
Place Burger Fries	27%	4%	45%	0%	0%	18%	37%
Place Can Basket	6%	5%	7%	0%	0%	2%	28%
Place Cans Plasticbox	5%	2%	27%	0%	0%	3%	23%
Place Container Plate	17%	45%	58%	1%	0%	1%	45%
Place Dual Shoes	4%	0%	3%	0%	0%	0%	19%
Place Empty Cup	7%	11%	53%	0%	0%	1%	8%
Place Fan	2%	10%	8%	0%	0%	1%	12%
Place Mouse Pad	0%	1%	11%	0%	0%	1%	20%
Place Object Basket	17%	2%	25%	0%	0%	0%	45%
Place Object Scale	0%	0%	20%	0%	0%	0%	35%
Place Object Stand	5%	11%	45%	0%	0%	0%	6%
Place Phone Stand	6%	7%	7%	0%	0%	2%	25%
Place Shoe	7%	6%	41%	0%	0%	2%	53%
Press Stapler	24%	29%	72%	6%	0%	3%	45%
Put Bottles Dustbin	4%	13%	10%	1%	0%	21%	35%
Put Object Cabinet	18%	18%	7%	0%	0%	1%	0%
Rotate QRcode	5%	15%	10%	0%	0%	1%	34%
Scan Object	1%	1%	7%	0%	0%	1%	23%
Shake Bottle Horizontally	51%	51%	96%	4%	18%	25%	44%
Shake Bottle	45%	60%	94%	10%	8%	19%	45%
Stack Blocks Three	0%	0%	14%	0%	0%	0%	18%
Stack Blocks Two	2%	1%	33%	0%	0%	0%	44%
Stack Bowls Three	17%	24%	24%	0%	0%	5%	87%
Stack Bowls Two	30%	41%	52%	0%	0%	6%	45%
Stamp Seal	0%	4%	15%	0%	0%	0%	0%
Turn Switch	15%	23%	20%	2%	1%	8%	16%
Average	13.72%	16.34%	25.68%	1.74%	0.64%	4.96%	28.40%

B.2 Public SimplerEnv Results

Table 9: Public SimplerEnv WidowX Visual Matching results used as the supplementary simulator comparison [Li et al., 2024b]. This table is intentionally compact because SimplerEnv serves as public-reference context rather than the primary mechanism study. Baseline references include OpenVLA [Kim et al., 2024], CogACT [Li et al., 2024a], SpatialVLA [Qu et al., 2025], π_0 and π_0-FAST [Black et al., 2024, Pertsch et al., 2025], GR00T-N1.5 [Bjorck et al., 2025], Magma [Yang et al., 2025b], and StarVLA-α variants [Community, 2026, Ye et al., 2026]. For NoTVLA, the success average is 65.7% and the grasp average is 82.2%.

Method	**Backbone / Family**	**WidowX VM Avg.**
OpenVLA	token VLA	4.2%
CogACT	hybrid policy	51.3%
SpatialVLA	spatial token VLA	42.7%
π_0	VLA + flow expert	27.1%
π_0-FAST	tokenized π variant	48.3%
GR00T-N1.5	dual-system VLA	61.9%
Magma	multimodal foundation model	35.8%
Qwen-OFT	Qwen3-VL-4B + chunked continuous	64.6%
Qwen-PI	Qwen3-VL-4B + flow expert	60.9%
NoTVLA	Qwen3-VL-4B + narrative sparse action	65.7%

B.3 AGIBOT / Challenge-Style Results

Platform. The AGIBOT challenge tasks are executed on the AgiBot G1 [Bu et al., 2025a], a wheeled dual-arm robot with two 7-DoF arms, a mobile chassis, and an adjustable waist and head. Its end effectors are modular (a 6-DoF dexterous hand or a two-finger gripper), and the upper body carries RGB-D and fisheye cameras, with additional cameras mounted at the end effectors. Image observations and proprioceptive states are recorded at 30 Hz. We use the platform specification published by the manufacturer and in Bu et al. [2025a].

Table 10: Official-server AGIBOT challenge-style results [Bu et al., 2025a]. These numbers are reported as an external benchmark context rather than as the primary evidence for the mechanism. UniVLA is included as the available public comparison [Bu et al., 2025b].

Method	**Total**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
UniVLA	2.795	0.097	0.020	0.033	0.350	0.260	0.400	1.000	0.080	0.375	0.180
NoTVLA	3.697	0.161	0.304	0.350	0.320	0.448	0.510	0.600	0.262	0.350	0.392

B.4 Semantic Probe Details

Table 11: Training-step probe accuracy on task-grounded QA. With Wilson 95% intervals over the 960 probe examples, the final checkpoints are 76% [73, 79] for NoTVLA and 0% [0, 0.4] for Qwen-PI, and the two intervals are fully separated from Step 200 onward.

Model	**0**	**100**	**200**	**300**	**500**	**1000**	**2000**	**4000**
Qwen-PI	97%	68%	23%	0%	0%	0%	0%	0%
NoTVLA	97%	88%	79%	80%	76%	76%	76%	76%

Table 12: Detailed semantic OOD task table with explicit Qwen-PI matched comparison. This compact table reports representative task-level entries; the aggregate OOD categories in Table 3 are averaged separately over inverse ordering, unseen color composition, choice-based placement, and novel concept transfer.

Task	π_0-30k	π_0-5k	$\pi_{0.5}$-30k	$\pi_{0.5}$-5k	Qwen-PI-30k	Qwen-PI-5k	NoTVLA-30k	NoTVLA-5k
Place mouse pad	23%	22%	35%	32%	27%	22%	25%	23%
Stack blocks two	90%	65%	93%	73%	96%	79%	90%	87%
Stack blocks two (inverse)	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	63%	70%
Stack random color blocks	30%	18%	35%	23%	48%	58%	62%	69%
Place block flagpad	32%	55%	44%	59%	57%	62%	84%	78%

B.5 Camera and Depth Robustness

Table 13: Robustness analysis against depth-anchor noise. Gaussian noise with the listed maximum amplitude is injected into the depth anchor, and each task is evaluated over 20 trials. The 0 cm row uses this stress-test protocol and trial count, so it is not expected to exactly match the 100-trial ablation averages in Table 4.

Task	Noise Level ($\sigma_{\max}$)			
	0 cm	**1 cm**	**10 cm**	**100 cm**
Place a2b left	40%	35%	20%	10%
Place a2b right	30%	25%	15%	5%
Place object scale	65%	25%	25%	15%
Stack blocks three	35%	30%	0%	0%
Stack bowl three	65%	40%	10%	10%
Blocks ranking size	20%	20%	0%	0%
Blocks ranking rgb	40%	35%	5%	10%
Average	**42%**	**30%**	**11%**	**7%**

Small depth perturbations degrade performance but do not immediately break the policy, which indicates that closed-loop re-planning can correct moderate grounding error. Large perturbations cause a steep drop, especially on stacking and ranking tasks, confirming that accurate depth is a central part of the NoTVLA interface rather than an optional sensor detail.

Finer-grained depth-noise sweep. To locate the degradation boundary more precisely, we ran a finer noise sweep on three representative tasks (Place a2b left, Stack blocks three, Blocks ranking size; 20 trials per task and noise level) and report Wilson 95% CIs over the 60 pooled trials per noise level in Table 14. The 0.5 cm level, a conservative upper bound on real per-frame sensor noise, is statistically indistinguishable from the clean baseline. Success then declines monotonically from 2 cm onward, and the interval separates from the clean baseline at 10 cm; given the small number of trials per level, we read this as a degradation trend that points to systematic bias, miscalibration, or structural depth errors rather than ordinary frame-level noise.

Table 14: Finer-grained depth-anchor noise sweep. Aggregate success over three tasks with Wilson 95% CIs, 20 trials per task (60 trials) and noise level.

Noise level	0 cm	0.5 cm	1 cm	2 cm	5 cm	10 cm	100 cm
Avg. success	31.7%	35.0%	28.3%	21.7%	11.7%	6.7%	3.3%
95% CI	[21.3, 44.2]	[24.2, 47.6]	[18.5, 40.8]	[13.1, 33.6]	[5.8, 22.2]	[2.6, 15.9]	[0.9, 11.4]

Real-sensor temporal noise decomposition. We measured the temporal behavior of the deployed RealSense D435if against flat targets at nominal distances (Table 15). Within the 0.3–1.2 m working range, the per-frame random noise is millimeter-level (Std 0.62 mm at 600 mm and 3.82 mm at 1200 mm) and EMA filtering (α=0.2) reduces it by 70.2% and 64.5%, respectively. The dominant residual error is a systematic bias ($+25.2$ mm at 1200 mm) that temporal filtering cannot remove. On object edges, single-pixel depth reads are valid in only 131/150 frames (87.3%) at 600 mm, and a 3×3 ROI median restores validity to 150/150 (100%). Together with the sweep above, this separates the

two error regimes: temporal filtering improves precision, while calibration is required for accuracy; the recommended deployment chain is 3×3 ROI median, EMA temporal filtering, and calibration.

Table 15: Temporal depth statistics of the RealSense D435if at nominal target distances (150 frames per setting).

Nominal distance	Systematic bias	Raw Std	EMA-0.2 Std	Std reduction
300 mm	$+3.0$ mm	0.00 mm	0.00 mm	—
600 mm	-2.9 mm	0.62 mm	0.19 mm	70.2%
1200 mm	$+25.2$ mm	3.82 mm	1.36 mm	64.5%

Camera-distance trend (preliminary). We additionally moved the real camera closer to the workspace on two precision tasks (Table 16). Success improves monotonically as the camera approaches, but with only 10 trials per cell the intervals overlap heavily, and the manipulation confounds two factors (lower depth noise and higher effective visual resolution), so we report this only as a preliminary trend: both factors favor fine-grained manipulation, and we do not attribute the gain to depth alone.

Table 16: Preliminary camera-distance study on two real-robot precision tasks (10 trials per cell).

Camera distance	Pick Pen	Draw Paper
1.2 m	2/10	5/10
0.6 m	5/10	6/10
0.3 m	6/10	8/10

Table 17: Camera robustness under mild fisheye distortion. Anchor-based grounding is compared with direct UVD prediction under the same distorted image condition.

Task	Original Camera + Depth	Fisheye + Anchor Depth	Fisheye + Direct UVD
Place a2b left (clean)	39%	32%	16%
Place a2b right (clean)	28%	24%	13%
Place a2b randomly (hard)	36%	30%	27%
Place a2b randomly (clean)	34%	25%	23%
Average	**34.3%**	**27.8%**	**19.8%**

With stratified 95% CIs over task means, the fisheye aggregates are 27.9% [24.7, 31.0] for anchor-depth grounding versus 20.0% [17.1, 22.9] for direct UVD, so the aggregate intervals of the two methods do not overlap under distortion.

B.6 Ablations

Table 18: Representation ablation summary over the seven tasks in Table 4. The same policy family is used while changing the sparse action grounding format.

Variant	Object-Recognition Avg.	Long-Horizon Avg.	Overall Avg.
Sparse keyframes + direct UVD	30.3%	52.3%	42.9%
Sparse keyframes + anchor point only	32.7%	46.3%	40.4%
Full NoTVLA	**38.3%**	**65.3%**	**53.7%**

In the anchor-point-only ablation, depth is removed from the narrative action interface and the detokenizer uses the current calibrated depth observation at each predicted image anchor to construct executable 3D targets. This isolates the value of explicitly predicting depth-related tokens while keeping the same projection pipeline.

The depth-sensitive grasping results are reported in Table 6; the appendix does not repeat the same table to avoid duplicate result reporting.

B.7 Closed-Loop Merging and Subkeyframe Attribution

To attribute the source of NoTVLA's success-rate gains, we ablate the two execution- and training-side mechanisms separately. First, Table 19 disables closed-loop trajectory merging and executes each narrative plan one-shot until the next full replan, over eight RoboTwin tasks with 100 trials each. The aggregate changes only from 72.8% with merging to 71.6% without, a 1.2pp difference within the confidence intervals. This indicates that execution-time trajectory merging is not the main source of the reported gains; it is better understood as a trajectory-continuity and smoothness component.

Table 19: Closed-loop merging ablation on eight RoboTwin tasks (success %, 100 trials per task).

	Grab Roller	**Handover Block**	**Handover Mic**	**Hanging Mug**	**Stack Blocks 3**	**Stack Blocks 2**	**Stack Bowls 3**	**Stack Bowls 2**	**Aggregate**
With merge	97	55	99	15	49	88	83	96	**72.8%**
No merge	99	61	98	11	45	85	80	94	**71.6%**

Second, Table 20 removes the subkeyframe trajectory cropping used to construct sparse narrative supervision, while keeping everything else fixed. The effect is large: stack cube drops from 78.0% [75.3, 80.5] to 23.0% [20.5, 25.7] and push cube from 99.0% [98.1, 99.5] to 94.0% [92.3, 95.3]. This experiment is conducted on the ManiSkill3 stack cube and push cube tasks with an InternVL2.5-1B backbone (1000 demonstrations, 1000 evaluation rollouts per task; Appendix E.6), so the effect is not specific to a single backbone family. Together, the two ablations are consistent with the gains coming from the narrative representation and its subkeyframe training-time decomposition, which acts as a form of data augmentation, rather than from execution-layer machinery.

Table 20: Subkeyframe trajectory-cropping ablation on the ManiSkill stack cube and push cube tasks (success % with Wilson 95% CIs over 1000 rollouts; InternVL2.5-1B backbone).

Task	**With subkeyframe**	**Without subkeyframe**
Stack cube	**78.0% [75.3, 80.5]**	23.0% [20.5, 25.7]
Push cube	**99.0% [98.1, 99.5]**	94.0% [92.3, 95.3]

B.8 Waypoint and Token Budget Sweep

Each narrative decision generates roughly 50 tokens for the 5-point waypoint format, which bounds the replanning rate. To test whether a smaller budget could increase reactivity, we sweep the waypoint count K on the ManiSkill3 stack cube task with the 7B backbone (Appendix E.6; Table 21). $K \leq 2$ cannot encode the approach–grasp–lift–place structure of the tasks and fails completely; $K{=}3$ completes some tasks but trails $K{=}5$ by 24pp; $K{=}10$ brings no success gain while nearly doubling latency. $K{=}5$ is therefore the global default selected from this success–latency sweep, not a per-task hand-picked value. Adaptive per-task budgets are a reasonable direction for future work.

Table 21: Waypoint-count sweep on the ManiSkill stack cube task: success rate and per-decision latency.

K **(waypoints)**	**1**	**2**	**3**	**5**	**10**
Success rate	0%	0%	74%	**98%**	98%
Per-decision latency	∼1.0 s	∼1.4 s	∼1.9 s	∼2.8 s	∼5.0 s

Table 22: Trajectory rendering quality for the deterministic detokenizer. The 10-point variant is slightly stronger, but the 5-point variant keeps the narrative action shorter while remaining close in quality. ↑ indicates higher is better and ↓ indicates lower is better. Baseline references include Magma [Yang et al., 2025b], HAMSTER [Li et al., 2025b], and LLARVA [Niu et al., 2024]; DTW and Hausdorff distance are standard trajectory distance diagnostics [Senin, 2008, Rogers, 1998].

Metric	NoTVLA 5-pt	NoTVLA 10-pt	Magma	HAMSTER	LLARVA
Cover F1 ↑	0.9546	**0.9567**	0.9282	0.9451	0.9189
Cover precision ↑	0.9558	**0.9612**	0.9484	0.9504	0.9201
Cover recall ↑	0.9550	**0.9604**	0.9534	0.9440	0.9415
DTW ↓	0.4564	**0.4554**	0.8353	0.7945	0.4894
Endpoint error ↓	0.0739	**0.0729**	0.1784	0.1181	0.0771
Frechet ↓	0.1035	**0.0945**	0.2296	0.1311	0.1673
Hausdorff ↓	0.0809	**0.0795**	0.1282	0.0822	0.0903
Max orth. dist. ↓	0.0744	**0.0725**	0.1106	0.1232	0.0923
Mean orth. dist. ↓	0.0322	**0.0313**	0.0476	0.0356	0.0677
Median orth. dist. ↓	0.0286	**0.0277**	0.0439	0.0410	0.0532
Startpoint error ↓	0.0778	**0.0769**	0.1449	0.1411	0.1743
LCSS similarity ↑	0.9660	**0.9671**	0.9562	0.9496	0.9447

Calibrating absolute DTW values. DTW measures trajectory-shape similarity and is not monotonically related to task success. To calibrate what an absolute DTW value means, we computed DTW between *successful* NoTVLA rollouts and RoboTwin replay trajectories (Table 23). Point-goal tasks such as Click Bell and Lift Pot yield very low DTW even for a single canonical solution, whereas multi-step, multi-solution tasks such as ranking and stacking naturally reach 0.6–0.8 even when the rollout succeeds, because several distinct trajectories solve the same task. The cross-task mean of about 0.455 in Table 22 therefore falls within the natural range of successful trajectories and reflects valid multi-solution variation rather than execution failure; relative comparisons under the same protocol remain valid.

Table 23: DTW between successful NoTVLA rollouts and RoboTwin replay trajectories, per task.

Task	Successful-rollout DTW
Click Bell	0.025
Lift Pot	0.061
Handover Block	0.295
Blocks Ranking RGB	0.629
Stack Blocks Three	0.772

B.9 Real-Robot Details

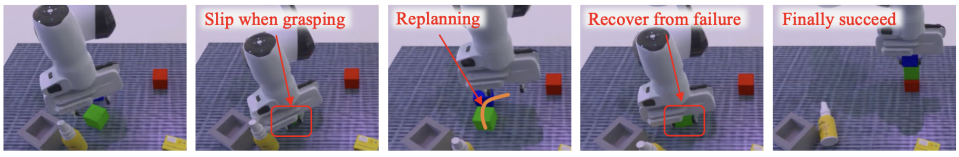


Figure 2: Closed-loop recovery from an initial grasping error through re-planning.

Table 24: Real-robot summary over five manipulation tasks. Time is averaged over successful trials.

Method	Tasks	Success Avg.	Avg. Time
DP3	5	44%	45.3s
Qwen-PI	5	49%	54.7s
NoTVLA w/ direct UVD	5	45%	37.5s
NoTVLA	5	**53%**	52.3s

Table 25: Real-robot task-level success. Each task uses task-specific termination criteria.

Method	Press Button	Place Cube Flag	Stack Cube Two	Insert Flower	Pick Fruit Bowl
DP3	80%	40%	30%	5%	**65%**
Qwen-PI	85%	**70%**	25%	**10%**	55%
NoTVLA w/ direct UVD	75%	55%	30%	5%	60%
NoTVLA	**90%**	60%	**50%**	**10%**	55%

Table 26: Average execution time for successful real-robot trials. Failed trials are excluded from the time average.

Method	Press Button	Place Cube Flag	Stack Cube Two	Insert Flower	Pick Fruit Bowl
DP3	13.6s	46.2s	60.4s	65.6s	40.6s
Qwen-PI	15.9s	60.6s	70.6s	75.6s	50.6s
NoTVLA w/ direct UVD	14.4s	47.9s	43.8s	45.4s	35.9s
NoTVLA	26.3s	62.5s	65.5s	61.5s	45.5s

B.10 Deferred Diagnostics

The current evaluation focuses on the evidence needed to test the central claim: a sparse narrative action interface can preserve more task-relevant semantics during robot adaptation and can improve semantic out-of-distribution manipulation under the tested protocols. Several diagnostics would further refine the mechanism analysis. The no-merge closed-loop ablation is reported in Appendix B.7. In particular, we leave IK invalid or out-of-workspace rates and failure-mode counts separating wrong target grounding, depth error, interpolation failure, and late recovery to future work. These diagnostics would help quantify where the interface fails; the present evidence chain distinguishes representation choice, semantic retention, grounding robustness, and deployment cost, but does not exhaustively explain every failure mode.

C Limitations and Scope

NoTVLA is designed around a semantics-preserving adaptation goal, not around maximizing performance on every manipulation regime. The method is most appropriate when a task benefits from retaining compositional language and visual semantics while delegating smooth execution to a transparent motion-rendering stage. It is less suited to regimes where success depends primarily on high-bandwidth contact dynamics, tactile feedback, very fast reactive control, or precise force regulation. The reported results should therefore be read as evidence for this interface choice in object-centric manipulation settings, not as a claim of universal VLA superiority.

The most important limitation is depth dependence. Anchor-conditioned grounding improves robustness compared with direct UVD prediction in our ablations, but NoTVLA still requires the depth value associated with the predicted anchor to be accurate enough for downstream projection and execution. The depth-noise results in Section B.5 show that small perturbations can be partially corrected by replanning, while large errors substantially reduce success. This also means that the method inherits calibration, occlusion, reflective-surface, and sensor-range failures from the depth source.

A second limitation is inference latency. In the current implementation, the VLM replans at a low narrative frequency and the deterministic detokenizer executes at a much higher actuator frequency. This separation makes the approach usable in the tested settings, but highly dynamic scenes may require faster multimodal inference, cached computation, or a learned recovery module. We therefore interpret the current latency as a systems bottleneck rather than a solved deployment problem.

Finally, the real-robot study is intentionally limited in scale. It is used to verify that the interface can be executed on hardware and that the latency/control-frequency separation is meaningful, not as comprehensive evidence of broad real-world generalization. The strongest evidence in this paper comes from matched simulation protocols, semantic retention probes, semantic OOD manipulation, and targeted grounding ablations; broader hardware coverage, multi-seed training, and larger real-world task suites remain future work. Evaluation-level confidence intervals are reported throughout (Appendix E.3). In addition, the deterministic detokenizer is not suited to contact-rich or dynamic

manipulation such as flinging and tossing [Ha and Song, 2022], depth is unavailable on some platforms such as UMI-style wrist-mounted grippers [Chi et al., 2024, Ha et al., 2024], and mobile heads, active perception, or mobile bases would require online extrinsic estimation that we have not validated [Xiong et al., 2025, Xu et al., 2026]; a consolidated limitations statement also appears in the main text.

D Broader Impact and Responsible Use

NoTVLA is intended as a research framework for studying how robot adaptation can preserve semantic competence in VLM-based policies. Potential positive impacts include more reusable robot policies, more interpretable action interfaces, and reduced dependence on embodiment-specific low-level supervision. At the same time, any improvement in general-purpose robot manipulation can increase the risk of unsafe deployment if systems are used outside tested environments or without appropriate physical safeguards.

The current system is not designed for unsupervised deployment in safety-critical settings. It depends on calibrated sensors, bounded workspaces, and task-specific termination criteria. We recommend using the method with conventional robot safety mechanisms, workspace constraints, and human oversight when transferring beyond the reported benchmarks. The real-robot results in this paper are presented as limited deployment checks rather than evidence that the system is ready for broad autonomous use.

E Data and Implementation Details

The code and reproduction instructions are available at `https://github.com/husk-huz/not-vla`.

E.1 Comparison Protocol and Claim Scope

We separate the experimental comparisons into three categories. First, *matched-protocol comparisons* use the same simulator task definitions, evaluation splits, and success criteria as NoTVLA. These comparisons are the primary basis for claims about action-interface design. Second, *backbone-family comparisons* compare NoTVLA with Qwen-style dense, chunked, or flow-based VLA variants. These results are used to study the effect of the adaptation interface under a closely related VLM family, while acknowledging that exact parameter count, pretraining version, and implementation details may differ across public systems. Third, *public-reference comparisons* report published or public-server numbers to contextualize performance on community benchmarks such as RoboTwin and SimplerEnv.

The central claim of the paper does not require NoTVLA to outperform every public method on every benchmark. Instead, the claim is that changing the robot adaptation target from dense low-level control to a sparse narrative action interface can preserve more task-grounded semantic competence and that this preservation is associated with stronger semantic OOD manipulation. Backbone differences are a possible confound for aggregate benchmark tables, especially when Qwen2.5-VL-7B and Qwen3-VL-4B results are shown together. For this reason, we treat semantic retention probes, semantic OOD tasks, and anchor/depth ablations as the primary evidence for the mechanism, and use public benchmark and real-robot results as external validation and deployment context rather than as standalone causal proof.

E.2 Data Sources and Coverage

We organize training and evaluation around three sources: RoboTwin-style simulated manipulation [Chen et al., 2025b], AGIBOT-style challenge data [Bu et al., 2025a], and private real-robot demonstrations collected under matched task definitions. RoboTwin is used for the main controlled manipulation study because it provides a broad set of object-centric tasks with official clean/random evaluation settings, reported in our tables using the corresponding Easy/Hard naming convention. SimplerEnv is used as an additional public simulator reference. AGIBOT-style results are included as challenge-style evaluation from the official server rather than as the primary mechanism study. Real-robot tasks are used as a deployment sanity check.

RoboTwin training follows the official task setting with 50 clean demonstrations per task, using the official trajectory filtering and evaluation split. We report NoTVLA on 50 official RoboTwin tasks; several public baselines are available only for the 39-task public snapshot, which is marked with dashes in the task-level tables. The semantic OOD suite uses 9 custom tasks derived from held-out instruction compositions, including inverse ordering, unseen color composition, target choice, and novel concept transfer. Each semantic OOD task is evaluated with 100 trials. The real-robot study uses 20 demonstrations and evaluates each task-method pair with 20 trials.

E.3 Statistical Reporting

Most manipulation results are reported as success proportions over finite evaluation rollouts. RoboTwin, semantic OOD, anchor/depth ablation, and simulated card-grasping results use 100 trials per task-method setting. SimplerEnv follows the same trial-count convention as the corresponding public benchmark results. Real-robot task-method pairs use 20 trials. Aggregate success values are reported as unweighted averages over tasks; in particular, the semantic OOD average is the unweighted mean over the four OOD categories, and the real-robot summary averages over the five hardware tasks.

For proportion-valued success rates, we report Wilson 95% confidence intervals computed from the reported success proportions and trial counts, and cross-task aggregates use stratified 95% CIs over task means, i.e., a normal-approximation interval $\pm 1.96\,\mathrm{SE}$ with $\mathrm{SE} = \frac{1}{T}\sqrt{\sum_{i=1}^{T} p_i(1-p_i)/n_i}$ for the mean of T task-level success proportions p_i over n_i trials. Interval endpoints are quoted alongside the key results in the main text and in the corresponding appendix tables; converting archived percentages back to counts can introduce at most a ± 1-count discrepancy, which does not change any interval-separation conclusion reported in this paper. We do not include multi-seed training variance: full multi-seed fine-tuning of VLA-scale policies is computationally prohibitive, so we follow the single-checkpoint, multi-rollout convention and report evaluation-level uncertainty explicitly. The reported numbers should therefore be interpreted as single-checkpoint, multi-rollout estimates rather than full training-and-evaluation uncertainty. This distinction is especially important for the real-robot study, where 20-trial estimates have wide, overlapping intervals and are used as deployment sanity checks rather than primary evidence for broad real-world generalization.

Accordingly, we interpret large and protocol-matched gaps, together with semantic retention and ablation evidence, as the main support for the mechanism claim. The key semantic OOD comparison is NoTVLA against Qwen-PI, π_0, and $\pi_{0.5}$ style baselines under the same task suite. Small differences in public-reference benchmark tables are not treated as statistically significant head-to-head comparisons.

E.4 Semantic Probe Construction

The task-grounded QA probe is designed to test whether robot fine-tuning preserves the VLM's ability to parse task-relevant visual and linguistic semantics. Each probe instance contains the current observation, the task instruction, and a short question about the target object, attribute, relation, or intended manipulation. Example question types include identifying the target object, selecting the correct color or attribute, determining the requested ordering relation, and resolving choice-based placement instructions.

Probe questions are generated from the same task family used for robot adaptation but are evaluated as language or short-answer predictions rather than as control rollouts. This makes the probe a diagnostic for semantic retention, not a substitute for manipulation performance. We therefore pair the probe with semantic OOD manipulation in the main text. The probe set contains 960 examples, balanced across five task categories. The question types cover target-object identification, color and attribute recognition, ordering relations, choice-based placement, and task-stage recognition. For example, a probe may ask "What is your target object, bread or clock?" with the answer "bread". Answers are scored by a rule-based regular-expression parser and reported as accuracy; all compared checkpoints are evaluated with the same prompt template and decoding settings.

E.5 Action Representation and Keyframe Extraction

Dense demonstrations are converted into sparse narrative action sequences before fine-tuning. We select decision-relevant keyframes using gripper-state transitions and kinematic changes, then optionally insert sub-keyframes to preserve local geometric context. The default NoTVLA representation uses a fixed 5-point sparse waypoint format per narrative prediction. Variable-length extracted sequences are resampled to this format by retaining the endpoints and selecting intermediate waypoints at uniform arc-length intervals. Each waypoint contains image coordinates, depth-related tokens, gripper state, and rotation tokens.

The keyframe acceleration threshold in Eq. 1 is set to $1.0\,\mathrm{m/s^2}$ for the main RoboTwin experiments and is applied to the translational component of the end-effector pose. The number of inserted sub-keyframes ranges from 2 to 10 before the final sparse formatting. Coordinate and depth tokens use a 0–1000 normalized token range for Qwen3-VL-4B experiments, while Qwen2.5-VL experiments use absolute image-coordinate values with the corresponding depth representation. Rotation is represented with Euler angles at the token level and converted to unit quaternions before SLERP-based interpolation. The gripper state is represented with open/close tokens. These choices are kept fixed for the representation ablations unless otherwise stated.

E.6 Training Recipe

NoTVLA fine-tunes an autoregressive VLM by next-token prediction over the narrative action sequence. We use Qwen3-VL-4B and Qwen2.5-VL-7B backbones across the reported experiments [Bai et al., 2025b]. SimplerEnv and the semantic QA probe use Qwen3-VL-4B; RoboTwin and AGIBOT use Qwen2.5-VL-7B. Unless otherwise stated, NoTVLA is trained for 3 epochs using AdamW with learning rate 1×10^{-5}, batch size 32, and weight decay 1×10^{-6}. We do not use a separate learning-rate scheduler unless otherwise noted. All training runs use H100 GPUs, with peak training memory around 96% of device memory. Training for nearly all reported tasks takes about 32 H100 GPU-hours in total.

The ManiSkill experiments use the ManiSkill3 simulator [Tao et al., 2024]. The subkeyframe ablation uses an InternVL2.5-1B backbone [Chen et al., 2024], whereas the waypoint-count sweep uses the 7B backbone (Qwen2.5-VL-7B), so absolute stack-cube success rates differ between the two experiments. Each model is trained on 1000 demonstrations for 3 epochs, and each task is evaluated over 1000 rollouts.

For Qwen-PI and Qwen-OFT style baselines, we follow the corresponding StarVLA public setting [Community, 2026, Ye et al., 2026]. Qwen-PI uses a Qwen3-VL-4B backbone with a flow-style action interface, while Qwen-OFT uses a Qwen3-VL-4B backbone with chunked continuous actions. The -30k and -5k suffixes denote the training-step checkpoints used for semantic OOD and retention probes. RoboTwin comparisons use the same 50-clean-demonstration setting where available. SimplerEnv comparisons for other methods are public-reference results rather than fully reproduced matched runs. Public-reference methods are cited as benchmark context and are not used as the sole basis for mechanism claims.

At inference time, each NoTVLA narrative decision contains roughly 50 generated tokens for the 5-point waypoint format. Narrative plans are refreshed at approximately 0.25–0.5 Hz, corresponding to a 2–4 s high-level replanning period in the current implementation, while the deterministic detokenizer produces commands at 80 Hz. The main deployment experiments use peak inference memory of approximately 13 GB.

E.7 Deployment Setup

The real-robot setup consists of a Franka arm with a Franka Panda gripper. We use an Intel RealSense D435if camera for RGB and depth input [Keselman et al., 2017]. Camera intrinsics and extrinsics are estimated using QR-code-based calibration. The predicted anchor point is paired with the corresponding depth value and projected into the robot base frame using the calibrated camera model.

Sparse narrative waypoints are de-quantized into 3D end-effector targets. Translation is rendered with cubic B-spline interpolation and orientation is rendered with SLERP. The controller executes the resulting trajectory at 80 Hz, while high-level VLM replanning runs asynchronously at approximately 0.25–0.5 Hz. When a new plan arrives during execution, the trajectory-merging procedure described

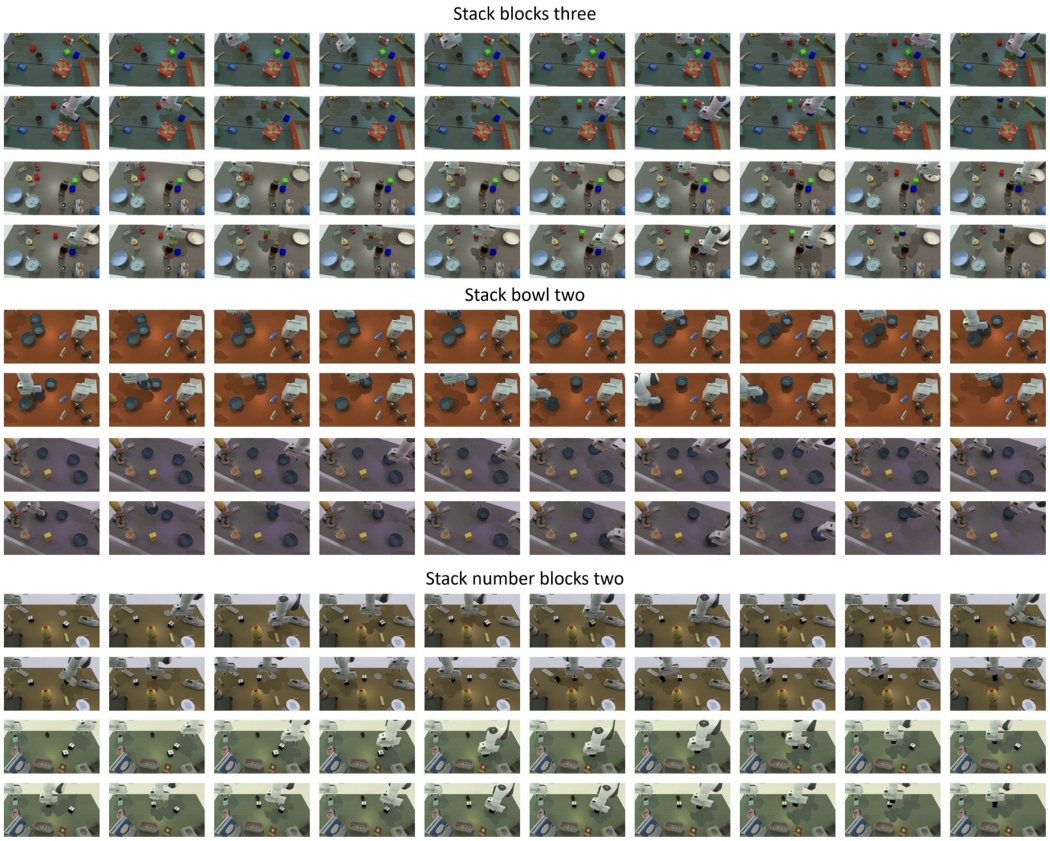


Figure 3: NoTVLA operation in simulation.

in Section 3.3 aligns the pending waypoints with the current robot state and connects them through a smooth transition segment.

Real-robot trials use task-specific termination criteria. The insertion task is counted as a failure if the flower falls onto the table; other tasks are counted as failures if the robot does not complete the specified goal before the evaluation cutoff or if execution violates the task-specific safety condition. Successful-trial execution time excludes failed trials, matching the reporting convention in Table 26.

F More Results and Real-World Performance

Figure 3 shows NoTVLA executions under randomized main-camera poses in simulation, and Figure 4 shows representative executions on the real robot.

G Task Descriptions

G.1 AGIBOT Challenge

Clear the countertop waste - task 1 Remove loose trash and food scraps from the countertop, place them into the appropriate waste or recycling container, and wipe the surface clean.

Open drawer and store items - task 2 Open the specified drawer, place the target items inside following organizational guidelines, arrange them neatly, and close the drawer securely.

Heat the food in the microwave - task 3 Place the food in a microwave-safe container, set the appropriate power and time, start the microwave, and carefully remove and check the food temperature when finished.

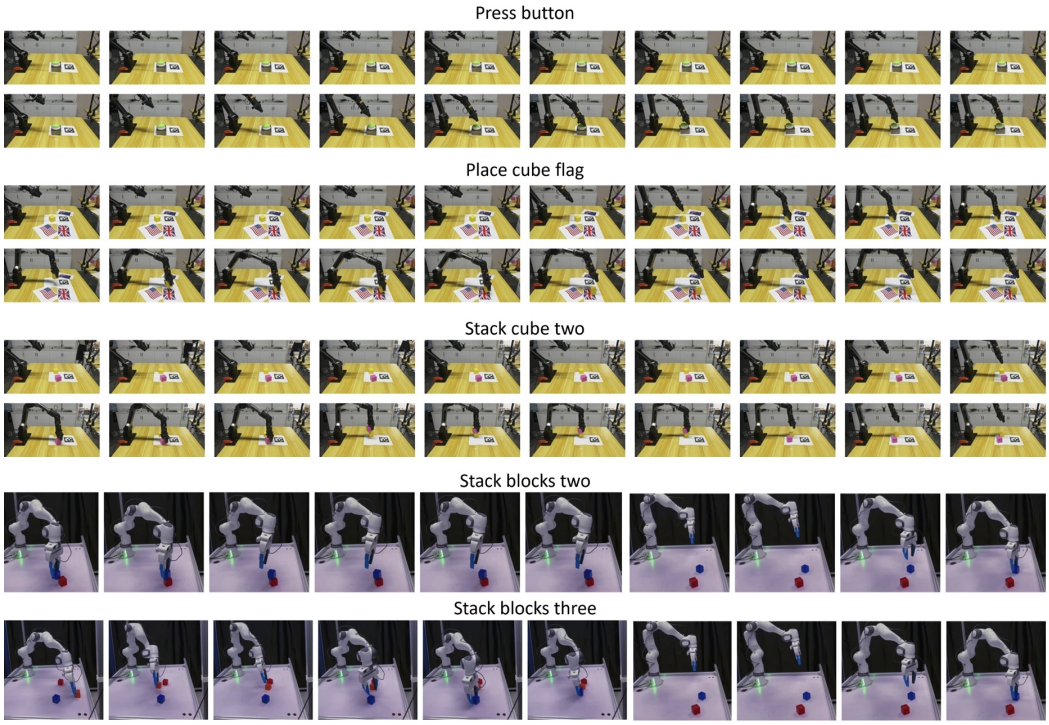


Figure 4: NoTVLA operation in reality.

Pack moving objects from conveyor - task 4 Identify and pick moving objects from the conveyor belt, place them into designated packing boxes or trays while maintaining speed and stability.

Pickup items from the freezer - task 5 Open the freezer, locate and retrieve specified frozen items using proper protection (e.g., gloves), move them to the target location, and close the freezer.

Restock supermarket items - task 6 Retrieve products from stock, refill shelves according to merchandising rules (facing front, orderly arrangement), replace expired or mislabeled items as needed.

Pack in the supermarket - task 7 Pack purchased goods for a customer by grouping items sensibly, protecting fragile goods, and arranging bags or boxes for safe and easy carrying.

Make a sandwich - task 8 Prepare bread and fillings according to the recipe, assemble ingredients in order, cut if required, and present a clean, ready-to-eat sandwich.

Clear table in the restaurant - task 9 Remove dishes, utensils, and leftovers from the table, clear and sanitize the surface, and reset place settings if necessary for the next guest.

Stamp the seal - task 10 Retrieve the stamp, align it with the required location on the document, press firmly to produce a clear impression, and verify alignment and clarity.

G.2 RoboTwin Official Tasks

Beat Block Hammer Use a hammer to strike the block target repeatedly until the action is completed or the block is secured.

Blocks Ranking RGB Rank blocks by visual RGB features and sort them according to the specified criterion.

Blocks Ranking Size Rank blocks by size and arrange them in the required order.

Click Alarmclock Locate the alarm clock and press or click the designated button to silence or activate the alarm.

Click Bell Press the bell actuator at the indicated position to produce a ringing signal.

Dump Bin Bigbin Open the large bin and empty its contents into the designated disposal area.

Grab Roller Reach for and grasp the roller object securely, then lift or move it to the target location.

Handover Mic Grasp the microphone and transfer it to another agent or specified handover position.

Hanging Mug Pick up a mug from a hanging position and place it on the table or target surface.

Lift Pot Grasp the pot handle(s) and lift it carefully to the required height or location.

Move Can Pot Move the can and the pot to their respective target positions as specified.

Move Pillbottle Pad Pick up the pill bottle and place it on the target pad or designated area.

Move Playingcard Away Remove the playing card from its current location and place it at the specified away location.

Move Stapler Pad Pick up the stapler and position it on the given pad or surface.

Open Laptop Open the laptop lid to a functional angle and ensure it is ready for use.

Open Micro Wave Open the microwave door and prepare the interior for loading or unloading items.

Place A2B Left Pick up object A and place it at location B on the left-side target as specified.

Place A2B Right Pick up object A and place it at location B on the right-side target as specified.

Place Bread Basket Place the bread item into the basket, arranging it neatly.

Place Bread Skillet Place the bread onto the skillet or pan and position it correctly for cooking or serving.

Place Burger Fries Pack or arrange the burger and fries together in the specified container.

Place Can Basket Place the can into the basket at the designated spot.

Place Cans Plasticbox Place the cans into the plastic box following placement constraints.

Place Container Plate Put the container onto the plate or place the plate into the container as required.

Place Dual Shoes Pick up the pair of shoes and place them together at the target location.

Place Empty Cup Place an empty cup at the specified position without spilling or tilting.

Place Fan Position the fan at the target location and orientation.

Place Mouse Pad Place the mouse pad flat at the designated workspace area.

Place Object Basket Place the specified object into the basket, ensuring stable placement.

Place Object Scale Place the object on the scale platform for weighing.

Place Object Stand Position the object on the stand securely and centered.

Place Phone Stand Place the phone onto the stand in the correct orientation.

Place Shoe Pick up the shoe and place it at the designated spot.

Press Stapler Press down the stapler mechanism to staple the target documents or materials.

Put Object Cabinet Place the object into the cabinet compartment and close the door if required.

Rotate QR Code Rotate the QR code marker to the specified orientation for scanning.

Scan Object Position the scanner or object and perform a scan to capture required data.

Shake Bottle Grasp the bottle and shake it vertically or as specified to mix contents.

Shake Bottle Horiz. Grasp the bottle and shake it horizontally to achieve the required motion.

Stack Block Three Stack three blocks vertically in the specified order and ensure stability.

Stack Block Two Stack two blocks as required and confirm alignment.

Stack Bowl Three Stack three bowls together with stable nesting.

Stack Bowl Two Stack two bowls together with proper alignment.

Stamp Seal Align the seal with the document and press to produce a clear stamp impression.

Turn Switch Rotate or flip the switch to the target position to change device state.

G.3 RoboTwin Custom Tasks

Stack Blocks Two Inverse Stack two blocks in the inverse of a previously seen order, requiring the policy to reinterpret the ordering instruction rather than replay the training distribution.

Stack Random Color Blocks Following an unseen color-order instruction, stack two random color blocks in the required order.

Move Block Random Color Pad Pick the colorful block and place it onto the correspondingly colored pad, matching color pairing and ensuring stable placement.

Stack Numberblocks Two Pick the two specified number blocks and stack them in required numerical order, bottom to top.

Stack Random Blocks Two From Three Given three candidate blocks, select the two specified by the instruction and stack them in the required color order.

Place Block Flagpad Pick the target block and place it onto the flag-marked pad, centered and fully supported.

Move Colorful Block Colorful Pad Pick the colorful block and place it onto the pad with the matching color pattern, aligning edges for stable contact.

Place Block Flagpad (choice) From multiple pads, choose a flag-labeled pad and place the block there.

Place A2B Randomly Pick an object from region A and place it at a random valid pose within region B, staying within bounds and avoiding obstacles.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: The abstract and introduction state the scope as semantics-preserving robot adaptation rather than a generic solution to VLA action-language mismatch. The main results and limitations are aligned with this narrowed claim.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: Limitations are discussed in the main-text conclusion and Appendix C, including depth dependence, inference latency, contact-rich and dynamic manipulation limits, single-checkpoint evaluation, and the limited scale of the real-robot study.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A].

Justification: The paper is empirical and does not present formal theoretical results requiring proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes].

Justification: The method, action representation, training recipe, evaluation protocols, and deployment setup are described in Sections 3–4 and Appendix E.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes].

   Justification: The implementation and reproduction instructions are publicly available at `https://github.com/husk-huz/not-vla`. The appendix documents the training recipe, data coverage, evaluation protocols, and deployment setup.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: Training details, data coverage, action representation choices, probe construction, and deployment settings are specified in Appendix E.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes].

   Justification: The paper reports Wilson 95% confidence intervals for the key proportion-valued results and stratified 95% confidence intervals over task means for aggregate metrics. Appendix E.3 states how the intervals are computed and clarifies that they quantify evaluation uncertainty for single trained checkpoints rather than multi-seed training variance.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: The appendix reports GPU type and count, peak training memory, peak inference memory, replanning rate, and control frequency for the main implementation, and the total training cost (about 32 H100 GPU-hours for nearly all reported tasks; Appendix E.6).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: The work is conducted as robot learning research with bounded simulation and laboratory robot evaluations. It does not involve human subjects, crowdsourcing, or sensitive personal data.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes].

Justification: Broader impacts and responsible-use considerations are discussed in Appendix D, including the need for physical safeguards and human oversight outside tested settings.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A].

Justification: The paper does not release a high-risk general-purpose model or scraped dataset as part of this submission.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes].

Justification: The paper cites existing models, datasets, simulators, and benchmark methods used for comparison. Public-reference results are identified as such in the experimental discussion and appendix.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes].

    Justification: The released code is documented in the public repository, including setup and reproduction instructions; the paper and repository state the scope and limitations of the released implementation.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

    Justification: The work does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A].

Justification: The work does not involve human-subject experiments.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes].

Justification: The core method uses autoregressive VLM backbones as robot policies, and this use is described throughout the method and experiment sections.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.