SimVLA: A Simple VLA Baseline for Robotic Manipulation

Anonymous Authors[1]

Abstract

Vision-Language-Action (VLA) models have emerged as a promising paradigm for general-purpose robotic manipulation, leveraging large-scale pre-training to achieve strong performance. The field has rapidly evolved with additional spatial priors and diverse architectural innovations. However, these advancements are often accompanied by varying training recipes and implementation details, which can make it challenging to disentangle the precise source of empirical gains. In this work, we introduce **SimVLA**, a streamlined baseline designed to establish a transparent reference point for VLA research. By strictly decoupling perception from control—using a standard vision-language backbone and a lightweight action head—and standardizing critical training dynamics, we demonstrate that a minimal design can achieve state-of-the-art performance. Despite having only 0.5B parameters, SimVLA outperforms multi-billion-parameter models on standard simulation benchmarks without robot pretraining. SimVLA also reaches on-par real-robot performance compared to $\pi_{0.5}$. Our results establish SimVLA as a robust, reproducible baseline that enables clear attribution of empirical gains to future architectural innovations.

1. Introduction

The field of Vision-Language-Action (VLA) learning has advanced rapidly, driven by a wave of architectural innovations. Recent methods have proposed increasingly sophisticated designs, ranging from mechanisms that enrich perception with temporal context, to modules that inject explicit 3D spatial awareness, to high-capacity decoders that model complex action distributions. While these contributions have pushed the boundaries of robot capabilities, they also introduce a significant challenge for the research community: attributing performance gains to specific components. Since architectural changes are frequently introduced alongside confounding variables—such as varying pretraining datasets, differing backbone scales, or ad-hoc optimization schedules—it can be difficult to disentangle the impact of a novel mechanism from the underlying training recipe.

Figure 1. **Out-of-box real-robot task examples.** We deploy SimVLA without any additional fine-tuning on our held-out scenes and evaluate it on a set of multi-stage tasks that require both dexterous manipulation and semantic understanding.

To facilitate clearer comparisons and accelerate progress, we introduce **SimVLA**, a streamlined baseline designed to serve as a transparent reference point for VLA research. Our goal is not to suggest that architectural complexity is unnecessary, but to establish a high-performance "lower bound" of complexity against which future innovations can be measured. By providing a clean, minimal design that achieves state-of-the-art results, we hope to help the community better quantify the true added value of sophisticated architectural components when they are introduced.

SimVLA adopts a modular philosophy that decouples perception from control: a standard pretrained vision-language backbone produces fused representations, which are then processed by a lightweight action head to predict continuous actions. This design offers a critical advantage in future-proofing: as vision-language models (VLMs) evolve, SimVLA allows researchers to swap in the latest SOTA backbones (e.g., upgrading from a 0.5B to a 7B model) without

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Table 1. **Performance and efficiency summary.** We report LIBERO success (%) and peak training VRAM at Batch=8 (GB) under a matched evaluation setup.

Model	Backbone	LIBERO Avg	VRAM (GB)
OpenVLA-OFT	7B	97.1	62.0 GB
$\pi_{0.5}$	3B	96.9	51.3 GB
VLA-Adapter	0.5B	97.3	24.7 GB
SimVLA (Ours)	**0.5B**	**98.6**	**9.3 GB**

redesigning complex cross-modal adapters. Furthermore, we rigorously standardize the often-overlooked training dynamics—such as data shuffling strategies, action space normalization, and optimization schedules—demonstrating that these implementation details often outweigh architectural differences in their impact on performance.

Despite its simplicity, SimVLA is both effective and highly efficient. Figure 1 illustrates SimVLA's zero-shot scene generalization capability when deployed on the Galaxea R1 Lite, additional details about the robot platform and training data are provided in Section 4.1.2. Table 1 further presents a representative example in which our model outperforms multi-billion-parameter baselines while maintaining a compact memory footprint.

Our main contributions are:

- We propose **SimVLA**, a modular VLA baseline that decouples perception from control, enabling a flexible and future-proof design that can easily adapt to new vision-language backbones.
- We identify and standardize the "silent" drivers of VLA performance—specifically data shuffling, normalization, and optimization dynamics—providing a rigorous training recipe that enables fair cross-model comparisons.
- We show that this minimal design achieves state-of-the-art performance, surpassing larger and more complex models on simulation benchmarks while enabling efficient real-robot transfer with zero-shot scene generalization.

The rest of our paper is organized as follows: Section 2 reviews recent VLA advances. Section 3 introduces SimVLA and our standardized training recipe. Section 4 evaluates SimVLA in simulation and on real robots with detailed ablations. We draw our conclusion in Section 5.

2. Related Work

The development of VLA models has accelerated rapidly, with numerous approaches proposing diverse strategies to improve robotic control through multimodal learning. In this section, we organize recent advances along three complementary axes: (1) visual and temporal augmentation methods that enrich perceptual inputs with motion cues, predictive modeling, or memory mechanisms; (2) geometric and 3D prior integration that injects explicit spatial understanding into the VLA framework; and (3) complex action representations and architectural innovations that explore more expressive decoders and efficient designs. For a more comprehensive review, readers are encouraged to consult recent surveys (Xu et al., 2025; Sapkota et al., 2025; Zhang et al., 2025b) that offer systematic analyses of the VLA research landscape.

Visual and Temporal Augmentation. Early VLA models, such as OpenVLA (Kim et al., 2024), typically rely on static RGB inputs, which limits their ability to reason about fine-grained physical dynamics or long-horizon dependencies. To bridge this gap, recent approaches have focused on augmenting the visual modality with explicit motion cues or temporal reasoning. For instance, FlowVLA (Zhong et al., 2025), CoT-VLA (Zhao et al., 2025), and TraceVLA (Zheng et al., 2024) introduce a "visual chain-of-thought" by explicitly predicting optical flow, sub-goal image, or overlaying trajectory traces onto input images, while 4D-VLA (Zhang et al., 2025a) integrates 4D spatiotemporal information to mitigate state chaos. Additionally, PixelVLA (Liang et al., 2025a) and ReconVLA (Song et al., 2025b) enhance visual grounding via auxiliary segmentation or reconstruction tasks.

Beyond immediate visual augmentation, several works incorporate predictive modeling to enhance planning. WorldVLA (Cen et al., 2025) and Dream-VLA (Zhang et al., 2025c) integrate world models to predict future states, while ThinkAct (Huang et al., 2025) and IntentionVLA (Chen et al., 2025) generate plans, or intention descriptions before acting. To handle long contexts, FPC-VLA (Yang et al., 2025), MemoryVLA (Shi et al., 2025), and HAMLET (Koo et al., 2025) propose dedicated memory modules or dual-stream architectures to make history-aware predictions. While these methods significantly improve state tracking and capture features at multiple temporal scales, they often incur substantial computational overhead and architectural complexity, requiring auxiliary estimators or multi-stage inference processes that can complicate real-time deployment.

Geometric and 3D Priors. Recognizing that 2D vision-language backbones may lack precise spatial understanding, a growing body of research explicitly injects 3D geometric priors into the VLA framework. Both 4D-VLA (Zhang et al., 2025a) and SpatialVLA (Qu et al., 2025a) apply positional encodings to enhance the spatial awareness. 4D-VLA fuses positional encoded 3D coordinates with visual tokens, whereas SpatialVLA introduces egocentric 3D position encodings that does not rely on camera extrinsics. GraspVLA (Deng et al., 2025) and MolmoAct (Lee et al., 2025) both introduce additional spatial priors to

enchance chain-of-though reasoning capabilities. Specifically, GraspVLA improves 3D awareness in its unified CoT progress through auxiliary training tasks such as detection and target grasp pose estimation, while MolmoAct (Lee et al., 2025) utilizes depth-aware perception tokens and visual trace to enable reasoning in space. Other approaches, such as GeoVLA (Sun et al., 2025), FALCON (Zhang et al., 2025d), and DepthVLA(Yuan et al., 2025) go a step further by processing point clouds, geometric tokens, or depth maps alongside RGB data. Although these spatially-aware architectures demonstrate superior precision in geometric tasks, they often introduce dependencies on specific sensors or heavy 3D encoders. This can reduce the model's generality across diverse embodiments compared to RGB-only baselines which are easier to scale and deploy.

Action Representations and Architectures. To address the limitations of simple discrete action tokenization, recent work has explored two primary directions: optimizing architectural efficiency and enhancing the expressivity of action distributions.

Focusing on efficiency and adaptation, several works modify the underlying architecture or tokenization scheme to reduce computational overhead. For instance, FAST (Pertsch et al., 2025) employs frequency-domain tokenization to compress trajectories, while PD-VLA (Song et al., 2025a) accelerates inference via parallel decoding. OpenVLA-OFT (Kim et al., 2025) forgoes action tokenization and directly regress continuous actions instead. On the architectural side, VLA-Adapter (Wang et al., 2025a) and FLOWER (Reuss et al., 2025) introduce lightweight adapters or action head that lowers computational burden, and x-VLA (Zheng et al., 2025b) utilizes soft prompts for scalable cross-embodiment learning. Specialized efficient models like NORA (Hung et al., 2025), SmolVLA (Shukor et al., 2025), and GR00T-N1 (Bjorck et al., 2025) further optimize performance on specific hardware or humanoid embodiments.

Parallel to efficiency, a significant body of work has investigated methods to model continuous multimodal distributions. Diffusion-based policies have emerged as a dominant paradigm in this area, with models like Diffusion Policy (Chi et al., 2023), π_0 (Black et al., 2024), $\pi_{0.5}$ (Black et al., 2025), and DD-VLA (Liang et al., 2025b). Meanwhile, Unified-VLA (Wang et al., 2025b) and UniVLA (Bu et al., 2025) explore unified tokenization and latent action spaces to capture causal dynamics, while UniAct (Zheng et al., 2025a) and VQ-VLA (Wang et al., 2025c) learn universal discrete action codebooks with vector quantization.

Despite their performance gains, these models often introduce challenges such as increased inference latency (e.g., iterative diffusion steps) or training instability. Our work offers a counterpoint to this trend. We demonstrate that a *simple but strong* baseline, built on standard flow matching and a empirically validated training recipe, can achieve competitive performance without the need for additional visual cues, spatiotemporal priors, or major architectural change.

3. SimVLA: A Simple VLA Baseline

As introduced in Section 2, recent VLA models have rapidly evolved with increasingly complex architectural components and additional priors. While these additions often yield empirical improvements, they also complicate fair comparisons across methods, making it difficult to disentangle gains from architectural novelty versus optimization and implementation choices. In this work, we deliberately take a conservative stance. Rather than introducing new architectural components, we ask a more fundamental question: *how strong can a minimal VLA design be when core modeling and training choices are carefully standardized?* Our goal is not to diminish the importance of richer inductive biases, but to establish a clean and reproducible baseline that clarifies what performance is achievable without additional architectural complexity.

To this end, we propose **SimVLA**, a simple and modular VLA baseline that decouples perception and control via a lightweight action head conditioned on vision-language representations. Despite its simplicity, SimVLA achieves competitive—and in several cases state-of-the-art—performance across standard benchmarks, while offering substantial improvements in training efficiency and inference throughput. We hope that this baseline can serve as a strong reference point for future work, enabling more precise evaluation of architectural innovations in VLA systems.

3.1. Preliminaries

Problem Formulation. We model the conditional distribution of a future action chunk $A_t = [a_t, a_{t+1}, \dots, a_{t+H-1}] \in \mathbb{R}^{H \times d_a}$ given an observation o_t. The observation contains the respective multi-view RGB images $I_t^1, \dots, I_t^n$, the pairing language instruction ℓ_t, and the robot proprioception (state) s_t: $o_t = [I_t^1, \dots, I_t^n, \ell_t, s_t]$.

VLM Backbone Encoder. Following the standard late-fusion paradigm (Black et al., 2024), we employ a pre-trained VLM E_ϕ to map multi-view RGB observations and the corresponding language instruction into a shared token representation:

$$Z_t = E_\phi(I_t^1, \dots, I_t^n, \ell_t).$$

Importantly, we deliberately treat the VLM as a *perception-language encoder*, rather than as an action-generating module. This design choice reflects a principled separation of responsibilities: high-level semantic understanding is handled by the VLM, while continuous control is delegated to a

lightweight action head. Such decoupling enables modularity, simplifies inference, and facilitates controlled analysis of downstream action modeling choices.

Although the VLM is used in an encoder-only role, it is not frozen by default. We jointly fine-tune the backbone together with the action head, optionally using a short warm-up stage for training stability. This preserves adaptability to the target robotic domain while maintaining a clear architectural boundary between perception-language representation and action generation.

Action Head. SimVLA uses a vanilla Transformer encoder (Vaswani et al., 2017) as an action head to model action chunks in continuous space. The action head consumes the fused VLM tokens Z_t, proprioception s_t, a timestep embedding, and a noised action chunk, and predicts the corresponding denoising vector.

Flow Matching. We model continuous action generation using conditional flow matching (Lipman et al., 2022; Black et al., 2024), which learns a deterministic vector field that transforms noise into data under the conditioning of the current observation. Compared to discrete autoregressive decoding or stochastic diffusion-based formulations, flow matching offers a lightweight and stable abstraction that is particularly well-suited for continuous control. Concretely, let x denote a normalized action chunk and $\epsilon \sim \mathcal{N}(0, I)$ denote Gaussian noise. We sample a noise level $t \in (0, 1]$ and construct a noised action $x_t = t\epsilon + (1-t)x$. The action head $v_\theta(x_t, o_t, t)$ is trained to predict the corresponding denoising vector field using a standard ℓ_2 objective:

$$\mathcal{L}(\theta) = \mathbb{E}\left[\|v_\theta(x_t, o_t, t) - (\epsilon - x)\|_2^2\right].$$

We emphasize that our goal is not to capture highly multi-modal action distributions, but rather to provide a simple and reliable mechanism for generating smooth and temporally consistent action chunks. At inference time, we integrate the learned vector field from noise to data using a small number of Euler steps, resulting in efficient and stable action generation suitable for real-time execution.

3.2. SimVLA Architecture

Design Principle. SimVLA adopts an intentionally minimal architectural design as illustrated in Figure 2: a vision-language encoder produces fused representations once per control step, and a lightweight action transformer generates continuous action chunks via flow matching. Our goal is not to introduce new architectural mechanisms, but to establish a clean and neutral baseline that isolates the effects of action modeling and training dynamics.

At each timestep, the fused vision-language tokens Z_t are first obtained from the VLM backbone. We then construct the input sequence to the action head by concatenating

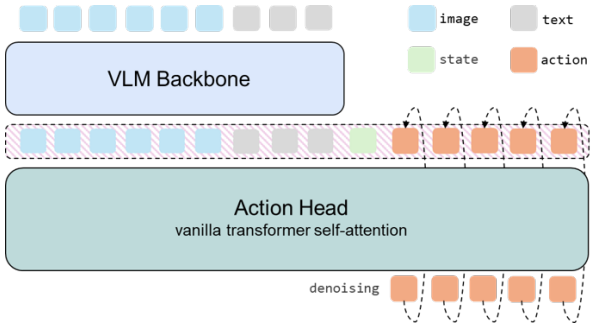

Figure 2. **SimVLA overview.** SimVLA is a minimal baseline: a VLM encoder produces fused vision-language tokens once per control step, and a lightweight action transformer performs flow-matching denoising to generate a continuous action chunk.

projected VLM tokens, a broadcasted proprioceptive state embedding, a sinusoidal time embedding, and the noised action chunk. This unified token sequence is processed by a vanilla Transformer encoder with pure self-attention (Vaswani et al., 2017), without cross-attention, memory modules, or modality-specific routing.

Rationale. We intentionally rely on self-attention over concatenated tokens as a neutral information integration mechanism. While more specialized architectures—such as cross-attention bridges or conditional normalization—may offer additional inductive biases, they also introduce confounding factors that complicate fair comparison. By using a single self-attention transformer, SimVLA allows the model to learn modality interactions directly from data, while keeping architectural assumptions minimal.

Practical Advantages. This design further yields practical benefits for deployment. Since the vision-language backbone is executed only once per control step, all subsequent denoising iterations are handled by the lightweight action head, resulting in reduced latency and improved inference throughput.

3.3. Training and Inference Recipe

A central takeaway of this work is that strong VLA performance can often be achieved through careful standardization of training and inference details, even with minimal architectural design. In practice, we find that several seemingly minor choices can dominate performance differences if left under-specified. Accordingly, we explicitly control and report the following factors across all experiments.

Action Representation and Normalization. We train the flow model in a normalized continuous action space, using per-dimension statistics computed from the training set. Proprioceptive states are normalized when applicable to improve optimizer conditioning. We predict action chunks of horizon H and execute them in a receding-horizon manner;

we emphasize that the choice of H is a major performance knob and must be tuned per benchmark.

Data Handling. Beyond action chunking, we carefully control data shuffling during training. Since demonstration trajectories exhibit strong temporal correlations, improper shuffling can lead to brittle optimization and poor long-horizon generalization. We find that consistent shuffling is critical for stable training and fair benchmarking.

Optimization Dynamics. We systematically sweep learning rates, warm-up schedules, and learning rate schedulers while keeping batch size and total training steps fixed across comparisons. Notably, we observe that learning rate selection alone can overshadow architectural differences if not properly tuned, underscoring the importance of reporting optimization details for reproducibility.

Architecture Configuration. While SimVLA employs a minimal action head by default, we ablate action transformer scale, VLM backbone choice, and information injection mechanisms (token concatenation, cross-attention, and conditional normalization). We view these variations as implementation choices rather than architectural novelties, and we report them to contextualize performance differences.

Inference. At inference time, SimVLA follows an *encode-once, denoise-in-the-head* workflow. Given an observation, the VLM backbone is executed once to obtain fused tokens, after which the lightweight action head performs a small number of Euler integration steps to generate a clean action chunk. The resulting actions are post-processed and executed in a receding-horizon fashion, enabling efficient real-time control.

4. Evaluation

To empirically validate the effectiveness of SimVLA, we conduct a comprehensive evaluation across standard simulation benchmarks and real-world robotic settings.

4.1. Experimental Setup

4.1.1. Simulation Benchmarks

We follow standard evaluation protocols for each simulation benchmark and keep the comparison settings consistent to support fair and reproducible results.

LIBERO (Liu et al., 2023). We utilize all four standard suites: LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long, comprising 10 tasks per suite with 500 expert demonstrations each. This serves as our primary testbed to evaluate the model's long-horizon consistency and generalization performance.

LIBERO-PRO (Zhou et al., 2025). To address the memorization issue in current VLA studies, we evaluate on LIBERO-PRO, a robust benchmark that introduces systematic perturbations across four dimensions: object appearance (Obj), spatial layout (Pos), language instructions (Sem), and task goals (Task).

SimplerEnv (Li et al., 2024). We evaluate on Simpler-Fractal (Google Robot) and Simpler-Bridge (WidowX) to assess the model's performance in high-fidelity simulated environments. For Simpler-Fractal, we report variant aggregation scores to test the policy's robustness against diverse scene variations. For Simpler-Bridge, the evaluation focuses on real-to-sim transfer across tasks.

Training Setup. Following recent works (Zheng et al., 2025b; Liang et al., 2025b; Wang et al., 2025a), we train a *single generalist policy* on the union of all standard LIBERO datasets (Spatial, Object, Goal, Long). For LIBERO-PRO, we directly evaluate this policy without additional fine-tuning to strictly test zero-shot robustness. For SimplerEnv, we train our model on the Fractal (Brohan et al., 2022) and BridgeData-V2 datasets (Walke et al., 2023), strictly following the rent work (Zheng et al., 2025b) . Across all benchmarks, we train directly from a pretrained VLM backbone, without any robotic data pre-training.

Critical Hyperparameters. A key finding of our research is that performance is highly sensitive to a small set of training and design choices, which can easily overshadow architectural differences if left under-tuned. To make our comparisons transparent, we summarize the concrete settings we tried:

- **Data & representation.** We vary the *action-chunk horizon* $H \in \{10, 20, 30\}$. We ablate *data shuffling* (shuffle vs. no shuffle). We ablate *normalization* (on/off) for both actions and proprioception; when enabled, we compute per-dimension statistics (mean/std, with robust quantile estimates) on the training set and normalize accordingly.
- **Optimization dynamics.** We sweep the *learning rate* over $\{5\times10^{-5}, 10^{-4}, 2\times10^{-4}, 5\times10^{-4}\}$, *warm-up steps* over $\{0, 1000\}$, and *the scheduler* over {cosine decay, none}. For all variants, we keep the training budget fixed (batch size and total steps/epochs); the exact budgets are reported in the appendix.
- **Architecture configuration.** *Action transformer scale:* we use two configurations (small vs. large), as in our training scripts, e.g., $\{768, 12, 12\}$ ($\sim$80M params) vs. $\{1024, 24, 16\}$ ($\sim$300M params) for {hidden size, depth, # heads}. *VLM backbone:* we compare Florence-2 (0.9B) (Xiao et al., 2024) and SmolVLM-0.5B (Marafioti et al., 2025). *Information injection:* we compare (i) token concatenation with pure self-attention (default), (ii) cross-attention injection, and (iii) conditional AdaLN.

Detailed hyperparameters are provided in Appendix A.

Baselines. We compare SimVLA against a spectrum of rep-

Table 2. **Comparison on the LIBERO benchmark.** We report the success rate (%) on the official test episodes for each suite (Spatial/Object/Goal/Long), and the overall average across the four suites. **Bold*** denotes the best performance, and **Bold** denotes the second-best. "B" indicates the backbone scale in billions.

Model	B	Spatial	Object	Goal	Long	Avg
Large Models (≥ 4B)						
UniVLA (Bu et al., 2025)	9	96.5	96.8	95.6	92.0	95.2
FlowVLA (Zhong et al., 2025)	8.5	93.2	95.0	91.6	72.6	88.1
UnifiedVLA (Wang et al., 2025b)	8.5	95.4	98.8	93.6	94.0	95.5
OpenVLA (Kim et al., 2024)	7	84.7	88.4	79.2	53.7	76.5
OpenVLA-OFT (Kim et al., 2025)	7	97.6	98.4	97.9	94.5	97.1
DD-VLA (Liang et al., 2025b)	7	97.2	98.6	97.4	92.0	96.3
MemoryVLA (Shi et al., 2025)	7	98.4	98.4	96.4	93.4	96.7
PD-VLA (Song et al., 2025a)	7	95.5	96.7	94.9	91.7	94.7
MolmoAct (Lee et al., 2025)	7	87.0	95.4	87.6	77.2	86.6
ThinkAct (Huang et al., 2025)	7	88.3	91.4	87.1	70.9	84.4
CoT-VLA (Zhao et al., 2025)	7	87.5	91.6	87.6	69.0	81.1
WorldVLA (Cen et al., 2025)	7	87.6	96.2	83.4	60.0	81.8
TraceVLA (Zheng et al., 2024)	7	84.6	85.2	75.1	54.1	74.8
FPC-VLA (Yang et al., 2025)	7	86.2	87.0	92.0	82.2	86.9
4D-VLA (Zhang et al., 2025a)	4	88.9	95.2	90.9	79.1	88.6
SpatialVLA (Qu et al., 2025a)	4	88.2	89.9	78.6	55.5	78.1
Small Models (< 4B)						
π_0 (Black et al., 2024)	3	96.8	98.8	95.8	85.2	94.2
π_0-FAST (Pertsch et al., 2025)	3	96.4	96.8	88.6	60.2	85.5
$\pi_{0.5}$ (Black et al., 2025)	3	**98.8**	98.2	**98.0**	92.4	96.9
NORA (Hung et al., 2025)	3	92.2	95.4	89.4	74.6	87.9
SmolVLA (Shukor et al., 2025)	2.2	93.0	94.0	91.0	77.0	88.8
GR00T-N1 (Bjorck et al., 2025)	2	94.4	97.6	93.0	90.6	93.9
GraspVLA (Deng et al., 2025)	1.8	-	94.1	91.2	82.0	89.1
FLOWER (Reuss et al., 2025)	1	97.1	96.7	95.6	93.5	95.7
Tiny Models (< 1B)						
X-VLA (Zheng et al., 2025b)	0.9	98.2	98.6	97.8	**97.6***	**98.1**
VLA-Adapter (Wang et al., 2025a)	0.5	97.8	**99.2**	97.2	95.0	97.3
VLA-OS (Gao et al., 2025)	0.5	87.0	96.5	92.7	66.0	85.6
UniAct (Zheng et al., 2025a)	0.5	77.0	87.0	77.0	70.0	77.8
SimVLA	0.5	**99.6***	**99.8***	**98.6***	**96.4**	**98.6***

Table 3. **Robustness evaluation on the LIBERO-PRO benchmark.** We report the success rate (%) across five perturbation dimensions: Original (Ori), Object (Obj), Position (Pos), Semantic (Sem), and Task (Task). **Bold** indicates the best performance.

Task Suite	Method	Ori	Obj	Pos	Sem	Task
Spatial	OpenVLA (Kim et al., 2024)	98.0	97.0	0.0	97.0	0.0
	$\pi_{0.5}$ (Black et al., 2025)	98.0	97.0	20.0	97.0	**1.0**
	SimVLA	**99.0**	**98.0**	**29.0**	**98.0**	0.0
Object	OpenVLA (Kim et al., 2024)	99.0	**98.0**	0.0	98.0	0.0
	$\pi_{0.5}$ (Black et al., 2025)	98.0	**98.0**	**17.0**	96.0	1.0
	SimVLA	**100.0**	85.0	1.0	**100.0**	**4.0**
Goal	OpenVLA (Kim et al., 2024)	98.0	96.0	0.0	98.0	0.0
	$\pi_{0.5}$ (Black et al., 2025)	97.0	**97.0**	**38.0**	97.0	0.0
	SimVLA	**99.0**	82.0	0.0	**99.0**	**10.0**
Long	OpenVLA (Kim et al., 2024)	93.0	81.0	0.0	96.0	0.0
	$\pi_{0.5}$ (Black et al., 2025)	93.0	**92.0**	**8.0**	93.0	1.0
	SimVLA	**96.0**	61.0	3.0	**98.0**	**10.0**

resentative policies, ranging from standard baselines to recent complex architectures: RT-1-X / RT-2-X (O'Neill et al., 2024), OpenVLA (Kim et al., 2024), Octo-Small / Octo-Base (Team et al., 2024), TraceVLA (Zheng et al., 2024), SpatialVLA (Qu et al., 2025b), UnifiedVLA (Wang et al., 2025b), UniVLA (Bu et al., 2025), X-VLA (Zheng et al., 2025b), VLA-Adapter (Wang et al., 2025a), VLA-OS (Gao et al., 2025), UniAct (Zheng et al., 2025a), NORA (Hung et al., 2025), MemoryVLA (Shi et al., 2025), ThinkAct (Huang et al., 2025), CoT-VLA (Zhao et al., 2025), WorldVLA (Cen et al., 2025), SmolVLA (Shukor et al., 2025), MolmoAct (Lee et al., 2025), π_0 (Black et al., 2024), π_0-FAST (Pertsch et al., 2025), $\pi_{0.5}$ (Black et al., 2025), DD-VLA (Liang et al., 2025b), OpenVLA-OFT (Kim et al., 2025), RoboVLM (Liu et al., 2025), GR00T-N1 (Bjorck et al., 2025), FlowVLA (Zhong et al., 2025), PD-VLA (Song et al., 2025a), FPC-VLA (Yang et al., 2025), 4D-VLA (Zhang et al., 2025a), GraspVLA (Deng et al., 2025), FLOWER (Reuss et al., 2025).

Baseline results are either sourced directly from the original papers or reproduced using open-source implementations under the identical input modalities described above.

4.1.2. Real-robot (Galaxea R1 Lite)

Beyond simulation, we evaluate zero-shot cross-scene generalization on a real mobile bimanual robot. We train two policies on the 500 hour Galaxea Open-World Dataset collected with the same embodiment (Jiang et al., 2025): (i) our SimVLA initialized from Florence-2, and (ii) a $\pi_{0.5}$ baseline initialized from the publicly released $\pi_{0.5}$ weights. We then deploy both policies in our own held-out scenes without additional fine-tuning.

We evaluate on eight multi-stage manipulation tasks that emphasize dexterous, fine-grained manipulation under diverse scenes and objects: *store the dolls*, *arrange eggs*, *put the flowers in the vase*, *put the pen into the pen holder*, *wipe the desktop*, *fold the clothes*, *pick up garbage on the ground*, and *open the drawer*. For each task, we report task success under a fixed time budget. Additional dataset/robot details and per-task rubrics are provided in Appendix A.

4.2. Simulation Results

Results on LIBERO. Table 2 shows that SimVLA establishes a new SOTA with an average success rate of 98.5%. Despite using a compact 0.5B backbone, it surpasses large baselines ($\geq$ 4B) like OpenVLA-OFT (97.1%) and MemoryVLA (96.7%), validating the efficacy of our simple, well-tuned architecture. SimVLA achieves near-perfect scores on *Spatial* (99.4%), *Object* (99.8%), and *Goal* (98.2%), ranking first overall. On the challenging *Long* suite, it attains a robust 96.4%, demonstrating strong temporal consistency without explicit memory modules.

Robustness on LIBERO-PRO. Table 3 presents our evaluation on the rigorous LIBERO-PRO benchmark. SimVLA demonstrates superior generalization, particularly in **Semantic** and **Task** robustness. While baselines like OpenVLA and $\pi_{0.5}$ collapse to near-zero performance on Task-level

Table 4. **Comparison on WidowX robot tasks**; success rates (%).

Method	Spoon	Carrot	Stack	Eggplant	Avg
Large Models (≥ 4B)					
RT-1-X (O'Neill et al., 2024)	0.0	4.2	0.0	0.0	6.3
Octo-Base (Team et al., 2024)	12.5	8.3	0.0	43.1	31.3
OpenVLA (Kim et al., 2024)	0.0	0.0	0.0	4.1	7.8
OpenVLA-OFT (Kim et al., 2025)	12.5	4.2	8.3	37.5	39.6
DD-VLA (Liang et al., 2025b)	29.2	29.2	20.8	70.8	54.2
RoboVLM (Liu et al., 2025)	29.2	25.0	12.5	58.3	38.5
SpatialVLA (Qu et al., 2025a)	16.7	25.0	29.2	100.0	42.7
MemoryVLA (Shi et al., 2025)	75.0	75.0	37.5	100.0	71.9
ThinkAct (Huang et al., 2025)	58.3	37.5	8.7	70.8	43.8
FPC-VLA (Yang et al., 2025)	58.3	45.8	79.2	75.0	64.6
Small Models (< 4B)					
Octo-Small (Team et al., 2024)	47.2	9.7	4.2	56.9	43.9
π_0 (Black et al., 2024)	29.1	0.0	16.7	62.5	40.1
π_0-FAST (Pertsch et al., 2025)	29.1	21.9	10.8	66.6	48.3
GR00T-N1 (Bjorck et al., 2025)	62.5	45.8	16.7	20.8	49.5
FLOWER (Reuss et al., 2025)	71.0	13.0	8.0	88.0	45.0
Tiny Models (< 1B)					
X-VLA (Zheng et al., 2025b)	100	91.7	95.8	95.8	95.8
SimVLA	**100**	**91.7**	**91.7**	**100**	**95.8**

Table 5. **Comparison on Google Robot tasks;** success rates (%).

Model (Variant Aggregation)	**Pick**	**Move**	**Open**	**Avg**
Large Models (≥ 4B)				
RT-1-X (O'Neill et al., 2024)	49.0	32.3	29.4	36.9
RT-2-X (O'Neill et al., 2024)	82.3	79.2	35.3	65.6
Octo-Base (Team et al., 2024)	0.6	3.1	1.1	1.6
OpenVLA (Kim et al., 2024)	54.5	47.7	17.7	40.0
OpenVLA-OFT (Kim et al., 2025)	65.3	59.0	12.2	45.5
RoboVLM (Liu et al., 2025)	75.6	60.0	10.6	48.7
TraceVLA (Zheng et al., 2024)	60.0	56.4	31.0	49.1
DD-VLA (Liang et al., 2025b)	82.5	64.6	23.6	56.9
SpatialVLA (Qu et al., 2025a)	88.0	72.7	41.8	67.5
ThinkAct (Huang et al., 2025)	84.0	63.8	47.6	65.1
Small Models (< 4B)				
π_0 (Black et al., 2024)	75.2	63.7	25.6	54.8
π_0-FAST (Pertsch et al., 2025)	77.6	68.2	31.3	59.0
GR00T-N1 (Bjorck et al., 2025)	78.8	62.5	13.2	51.5
Tiny Models (< 1B)				
X-VLA (Zheng et al., 2025b)	85.5	79.8	61.9	75.7
SimVLA	**87.4**	**65.2**	**75.9**	**76.1**

perturbations (indicating reliance on trajectory memorization), SimVLA achieves emergent generalization, reaching **10.0%** success on both *Goal* and *Long* suites. Furthermore, SimVLA consistently dominates in Semantic robustness (ranking first across all suites with ∼99% success) and outperforms peers in Spatial Position robustness (i.e., **29.0%** in *Spatial*). These results suggest that our simplified architecture effectively grounds instructions into policy execution.

Results on WidowX. As detailed in Table 4, SimVLA achieves state-of-the-art performance with an overall average of 95.8%, effectively tying with the heavily pre-trained X-VLA. Despite strictly adhering to a **no pre-training** protocol, SimVLA matches X-VLA's average success rate and even secures perfect scores (100%) on the *Put Spoon on Towel* and *Put Eggplant in Basket* tasks. This result is particularly significant as SimVLA outperforms large-scale baselines by substantial margins—surpassing MemoryVLA (71.9%) and FPC-VLA (64.6%).

Results on Google Robot. In the Google Robot variant aggregation tasks shown in Table 5, SimVLA achieves an average success rate of 76.1%, outperforming strong baselines such as SpatialVLA (67.5%), RT-2-X (65.6%), and ThinkAct (65.1%). SimVLA is also slightly higher than X-VLA (75.7%) on the overall average. Together, these results suggest that a simple, data-efficient baseline can remain competitive on challenging benchmarks without relying on extensive robotic pre-training.

4.3. Ablation Studies

To examine which parts of our training recipe matter in practice (Sec. 3.3), we conduct controlled ablations on LIBERO by varying one knob at a time while keeping the remaining settings fixed (Table 6). Overall, we find that several implementation details are indispensable: changing a single knob can lead to a substantial performance drop, often larger than the gains attributed to architectural changes.

Key Findings. Table 6 highlights a few dominant knobs that largely determine performance.

- **Data shuffling and normalization are critical.** Disabling either shuffling or action normalization causes a near-collapse in performance, suggesting that stable optimization and consistent action scaling are prerequisites for a strong baseline.
- **Optimization dynamics dominate.** The learning rate must be tuned: too large (5×10^{-4}) degrades sharply, while too small (5×10^{-5}) also underperforms. Likewise, removing the small VLM learning-rate multiplier (setting it to 1.0) hurts substantially, indicating that preserving the pretrained backbone while adapting the action head is important for stable end-to-end training.
- **Some architecture choices matter, but are secondary to the above.** Scaling down the action head (large→small) only slightly reduces performance, whereas alternative conditioning mechanisms (AdaLN / cross-attention) are noticeably worse than simple token concatenation under our setup. Switching the VLM backbone to Florence-2 remains competitive, consistent with the modular "VLM encoder + action head" design.

4.4. Real-robot Results

In the following section, we report real-robot evaluation results on Galaxea R1 Lite under the zero-shot, cross-scene protocol described above. We compare our SimVLA against a $\pi_{0.5}$ baseline on eight tasks: *store the dolls*, *arrange eggs*,

Table 6. **Ablations on LIBERO.** Each row corresponds to one ablation setting with the remaining knobs fixed to the default configuration.

Knob	Value	Spatial	Object	Goal	Long	Avg
SimVLA	Default settings	**99.4**	**99.8**	**98.6**	**96.4**	**98.6**
Data & representation (default: H=10, shuffling on, normalization on)						
Action chunk horizon H	$H = 20$	99.2	89.6	92.4	88.4	92.4
	$H = 30$	95.4	93.8	80.6	79.2	87.3
Data shuffling	off	6.2	0.0	13.6	0.0	9.9
Action normalization	off	22.6	3.2	23.2	0.0	12.3
Optimization dynamics (default: lr 2×10^{-4}, warm-up none, scheduler none, VLM LR multiplier 0.1)						
Learning rate	5×10^{-5}	98.0	97.6	96.2	70.4	90.6
	1×10^{-4}	99.6	98.2	98.4	85.6	95.5
	5×10^{-4}	84.4	91.8	76.0	38.4	72.7
Warm-up steps	1000	97.4	99.6	96.4	93.8	96.8
Scheduler	cosine	99.2	99.4	98.4	93.0	97.5
VLM LR multiplier	1.0	41.2	80.8	46.4	8.2	44.2
Architecture configuration (default: large head (1024,24,16), concat injection, SmolVLM-0.5B)						
Action transformer scale	small (768,12,12)	98.8	99.6	98.6	94.8	98.0
Condition injection	conditional AdaLN	99.2	98.0	96.6	70.4	91.1
	cross-attention	98.4	96.6	94.8	76.2	91.5
VLM backbone	Florence-2	99.8	99.2	98.0	93.8	97.7

put the flowers in the vase, *put the pen into the pen holder*, *wipe the desktop*, *fold the clothes*, *pick up garbage on the ground*, and *open the drawer*. Fig. 1 shows example out-of-box deployments of SimVLA in our held-out real-world scenes.

Results on Real Robot. Overall, SimVLA achieves performance that is broadly comparable to $\pi_{0.5}$ under the same zero-shot protocol (Fig. 3). Beyond *fold the clothes*, *put the pen into the pen holder* and *put the flowers in the vase*, which remain challenging, the other tasks are typically around 80% success. Notably, our SimVLA is trained end-to-end directly from a pretrained VLM backbone (without any VLA/robot-data pre-training), whereas the $\pi_{0.5}$ baseline uses its publicly released initialization.

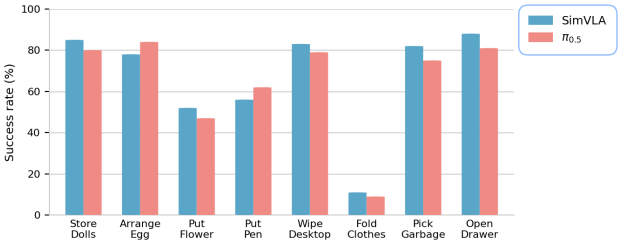

Figure 3. **Real-robot zero-shot results on Galaxea R1 Lite.**

5. Conclusion

In this work, we introduced **SimVLA**, a minimalist Vision-Language-Action baseline designed to address the challenge of performance attribution in the rapidly evolving VLA landscape. By strictly decoupling perception from control and adhering to a standardized training recipe, we demonstrated that a small model can rival or even outperform multi-billion-parameter SOTA baselines. Our extensive evaluations on the simulation benchmarks and real-world robotic tasks confirm that SimVLA achieves superior performance and scene generalization while maintaining a low memory footprint.

Crucially, our findings highlight that "silent" implementation details—such as data shuffling strategies, action normalization, and optimization dynamics—are often as influential as architectural novelties. By isolating these factors, SimVLA provides the community with a reproducible reference point. We hope that this baseline enables researchers to more rigorously quantify the added value of future architectural innovations.

Impact Statement

This paper presents work whose goal is to advance the field of Graph Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Bjorck, J., Castañeda, F., Cherniadev, N., Da, X., Ding, R., Fan, L., Fang, Y., Fox, D., Hu, F., Huang, S., et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Black, K., Brown, N., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B., Jakubczak, S., Jones, T., Ke, L., Levine, S., Li-Bell, A., Mothukuri, M., Nair, S., Pertsch, K., Shi, L. X., Tanner, J., Vuong, Q., Walling, A., Wang, H., and Zhilinsky, U. π_0: A Vision-Language-Action Flow Model for General Robot Control. *arXiv preprint arXiv:2410.24164*, 2024.

Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Galliker, M. Y., Ghosh, D., Groom, L., Hausman, K., Ichter, B., Jakubczak, S., Jones, T., Ke, L., LeBlanc, D., Levine, S., Li-Bell, A., Mothukuri, M., Nair, S., Pertsch, K., Ren, A. Z., Shi, L. X., Smith, L., Springenberg, J. T., Stachowicz, K., Tanner, J., Vuong, Q., Walke, H., Walling, A., Wang, H., Yu, L., and Zhilinsky, U. $\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Bu, Q., Yang, Y., Cai, J., Gao, S., Ren, G., Yao, M., Luo, P., and Li, H. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

Cen, J., Yu, C., Yuan, H., Jiang, Y., Huang, S., Guo, J., Li, X., Song, Y., Luo, H., Wang, F., Zhao, D., and Chen, H. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

Chen, Y., Gu, K., Wen, Y., Zhao, Y., Wang, T., and Nie, L. Intentionvla: Generalizable and efficient embodied intention reasoning for human-robot interaction. *arXiv preprint arXiv:2510.07778*, 2025.

Chi, C., Feng, S., Du, Y., Xu, Z., Cousineau, E., Burchfiel, B., and Song, S. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

Deng, S., Yan, M., Wei, S., Ma, H., Yang, Y., Chen, J., Zhang, Z., Yang, T., Zhang, X., Zhang, W., Cui, H., Zhang, Z., and Wang, H. Graspvla: a grasping foundation model pre-trained on billion-scale synthetic action data. *arXiv preprint arXiv:2505.03233*, 2025.

Gao, C., Liu, Z., Chi, Z., Huang, J., Fei, X., Hou, Y., Zhang, Y., Lin, Y., Fang, Z., Jiang, Z., and Shao, L. Vla-os: Structuring and dissecting planning representations and paradigms in vision-language-action models. *arXiv preprint arXiv:2506.17561*, 2025.

Huang, C.-P., Wu, Y.-H., Chen, M.-H., Wang, Y.-C. F., and Yang, F.-E. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. *arXiv preprint arXiv:2507.16815*, 2025.

Hung, C.-Y., Sun, Q., Hong, P., Zadeh, A., Li, C., Tan, U.-X., Majumder, N., and Poria, S. Nora: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

Jiang, T., Yuan, T., Liu, Y., Lu, C., Cui, J., Liu, X., Cheng, S., Gao, J., Xu, H., and Zhao, H. Galaxea open-world dataset and g0 dual-system vla model. *arXiv preprint arXiv:2509.00576*, 2025.

Kim, M. J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., Rafailov, R., Foster, E., Lam, G., Sanketi, P., Vuong, Q., Kollar, T., Burchfiel, B., Tedrake, R., Sadigh, D., Levine, S., Liang, P., and Finn, C. Openvla: An open-source Vision-Language-Action model. *arXiv preprint arXiv:2406.09246*, 2024.

Kim, M. J., Finn, C., and Liang, P. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Koo, M., Choi, D., Kim, T., Lee, K., Kim, C., Seo, Y., and Shin, J. Hamlet: Switch your vision-language-action model into a history-aware policy. *arXiv preprint arXiv:2510.00695*, 2025.

Lee, J., Duan, J., Fang, H., Deng, Y., Liu, S., Li, B., Fang, B., Zhang, J., Wang, Y. R., Lee, S., Han, W., Pumacay, W., Wu, A., Hendrix, R., Farley, K., VanderBilt, E., Farhadi, A., Fox, D., and Krishna, R. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

Li, X., Hsu, K., Gu, J., Mees, O., Pertsch, K., Walke, H. R., Fu, C., Lunawat, I., Sieh, I., Kirmani, S., Levine, S., Wu, J., Finn, C., Su, H., Vuong, Q., and Xiao, T. Evaluating real-world robot manipulation policies in simulation. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=LZh48DTg71.

Liang, W., Sun, G., He, Y., Dong, J., Dai, S., Laptev, I., Khan, S., and Cong, Y. Pixelvla: Advancing pixel-level understanding in vision-language-action model. *arXiv preprint arXiv:2511.01571*, 2025a.

Liang, Z., Li, Y., Yang, T., Wu, C., Mao, S., Nian, T., Pei, L., Zhou, S., Yang, X., Pang, J., Mu, Y., and Luo, P. Discrete diffusion vla: Bringing discrete diffusion to action decoding in vision-language-action policies. *arXiv preprint arXiv:2508.20072*, 2025b.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Liu, B., Zhu, Y., Gao, C., Feng, Y., Liu, Q., Zhu, Y., and Stone, P. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Liu, H., Li, X., Li, P., Liu, M., Wang, D., Liu, J., Kang, B., Ma, X., Kong, T., and Zhang, H. Towards generalist robot policies: What matters in building vision-language-action models. 2025.

Marafioti, A., Zohar, O., Farré, M., Noyan, M., Bakouch, E., Cuenca, P., Zakka, C., Allal, L. B., Lozhkov, A., Tazi, N., et al. Smolvlm: Redefining small and efficient multimodal models. *arXiv preprint arXiv:2504.05299*, 2025.

O'Neill, A., Rehman, A., Maddukuri, A., Gupta, A., Padalkar, A., Lee, A., Pooley, A., Gupta, A., Mandlekar, A., Jain, A., et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Pertsch, K., Stachowicz, K., Ichter, B., Driess, D., Nair, S., Vuong, Q., Mees, O., Finn, C., and Levine, S. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Qu, D., Song, H., Chen, Q., Yao, Y., Ye, X., Ding, Y., Wang, Z., Gu, J., Zhao, B., Wang, D., and Li, X. Spatialvla: Exploring spatial representations for visual-language-action model, 2025a. URL https://arxiv.org/abs/2501.15830.

Qu, D., Song, H., Chen, Q., Yao, Y., Ye, X., Ding, Y., Wang, Z., Gu, J., Zhao, B., Wang, D., and Li, X. Spatialvla: Exploring spatial representations for Vision-Language-Action model. *arXiv preprint arXiv:2501.15830*, 2025b.

Reuss, M., Zhou, H., Rühle, M., Ömer Erdinç Yağmurlu, Otto, F., and Lioutikov, R. Flower: Democratizing generalist robot policies with efficient vision-language-action flow policies. *arXiv preprint arXiv:2509.04996*, 2025.

Sapkota, R., Cao, Y., Roumeliotis, K. I., and Karkee, M. Vision-language-action models: Concepts, progress, applications and challenges, 2025. URL https://arxiv.org/abs/2505.04769.

Shi, H., Xie, B., Liu, Y., Sun, L., Liu, F., Wang, T., Zhou, E., Fan, H., Zhang, X., and Huang, G. Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. *arXiv preprint arXiv:2508.19236*, 2025.

Shukor, M., Aubakirova, D., Capuano, F., Kooijmans, P., Palma, S., Zouitine, A., Aractingi, M., Pascal, C., Russi, M., Marafioti, A., Alibert, S., Cord, M., Wolf, T., and Cadene, R. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Song, W., Chen, J., Ding, P., Zhao, H., Zhao, W., Zhong, Z., Ge, Z., Ma, J., and Li, H. Accelerating vision-language-action model integrated with action chunking via parallel decoding. *arXiv preprint arXiv:2503.02310*, 2025a.

Song, W., Zhou, Z., Zhao, H., Chen, J., Ding, P., Yan, H., Huang, Y., Tang, F., Wang, D., and Li, H. Reconvla: Reconstructive vision-language-action model as effective robot perceiver. *arXiv preprint arXiv:2508.10333*, 2025b.

Sun, L., Xie, B., Liu, Y., Shi, H., Wang, T., and Cao, J. Geovla: Empowering 3d representations in vision-language-action models. *arXiv preprint arXiv:2508.09071*, 2025.

Team, O. M., Ghosh, D., Walke, H., Pertsch, K., Black, K., Mees, O., Dasari, S., Hejna, J., Kreiman, T., Xu, C., Luo, J., Tan, Y. L., Sanketi, P. R., Vuong, Q., Xiao, T., Sadigh, D., Finn, C., and Levine, S. Octo: An open-source generalist robot policy. *ArXiv*, abs/2405.12213, 2024.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Walke, H. R., Black, K., Zhao, T. Z., Vuong, Q., Zheng, C., Hansen-Estruch, P., He, A. W., Myers, V., Kim, M. J., Du, M., et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pp. 1723–1736. PMLR, 2023.

Wang, Y., Ding, P., Li, L., Cui, C., Ge, Z., Tong, X., Song, W., Zhao, H., Zhao, W., Hou, P., Huang, S., Tang, Y., Wang, W., Zhang, R., Liu, J., and Wang, D. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model. *arXiv preprint arXiv:2509.09372*, 2025a.

Wang, Y., Li, X., Wang, W., Zhang, J., Li, Y., Chen, Y., Wang, X., and Zhang, Z. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025b.

Wang, Y., Zhu, H., Liu, M., Yang, J., Fang, H.-S., and He, T. Vq-vla: Improving vision-language-action models via scaling vector-quantized action tokenizers. *arXiv preprint arXiv:2507.01016*, 2025c.

Xiao, B., Wu, H., Xu, W., Dai, X., Hu, H., Lu, Y., Zeng, M., Liu, C., and Yuan, L. Florence-2: Advancing a unified representation for a variety of vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4818–4829, 2024.

Xu, C., Zhang, S., Liu, Y., Sun, B., Chen, W., Xu, B., Liu, Q., Wang, J., Wang, S., Luo, S., Peters, J., Vasilakos, A. V., Zafeiriou, S., and Deng, J. An anatomy of vision-language-action models: From modules to milestones and challenges, 2025. URL https://arxiv.org/abs/2512.11362.

Yang, Y., Duan, Z., Xie, T., Cao, F., Shen, P., Song, P., Jin, P., Sun, G., Xu, S., You, Y., and Liu, J. Fpc-vla: A vision-language-action framework with a supervisor for failure prediction and correction. *arXiv preprint arXiv:2509.04018*, 2025.

Yuan, T., Liu, Y., Lu, C., Chen, Z., Jiang, T., and Zhao, H. Depthvla: Enhancing vision-language-action models with depth-aware spatial reasoning. *arXiv preprint arXiv:2510.13375*, 2025.

Zhang, J., Chen, Y., Xu, Y., Huang, Z., Zhou, Y., Yuan, Y.-J., Cai, X., Huang, G., Quan, X., Xu, H., and Zhang, L. 4d-vla: Spatiotemporal vision-language-action pretraining with cross-scene calibration. *arXiv preprint arXiv:2506.22242*, 2025a.

Zhang, K., Yun, P., Cen, J., Cai, J., Zhu, D., Yuan, H., Zhao, C., Feng, T., Wang, M. Y., Chen, Q., Pan, J., Zhang, W., Yang, B., and Chen, H. Generative artificial intelligence in robotic manipulation: A survey, 2025b. URL https://arxiv.org/abs/2503.03464.

Zhang, W., Liu, H., Qi, Z., Wang, Y., Yu, X., Zhang, J., Dong, R., He, J., Lu, F., Wang, H., Zhang, Z., Yi, L., Zeng, W., and Jin, X. Dreamvla: A vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025c.

Zhang, Z., Li, H., Dai, Y., Zhu, Z., Zhou, L., Liu, C., Wang, D., Tay, F. E. H., Chen, S., Liu, Z., Liu, Y., Li, X., and Zhou, P. From spatial to actions: Grounding vision-language-action model in spatial foundation priors. *arXiv preprint arXiv:2510.17439*, 2025d.

Zhao, Q., Lu, Y., Kim, M. J., Fu, Z., Zhang, Z., Wu, Y., Li, Z., Ma, Q., Han, S., Finn, C., Handa, A., Liu, M.-Y., Xiang, D., Wetzstein, G., and Lin, T.-Y. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. *CVPR 2025*, 2025. arXiv:2503.22020.

Zheng, J., Li, J., Liu, D., Zheng, Y., Wang, Z., Ou, Z., Liu, Y., Liu, J., Zhang, Y.-Q., and Zhan, X. Universal actions for enhanced embodied foundation models. *arXiv preprint arXiv:2501.10105*, 2025a.

Zheng, J., Li, J., Wang, Z., Liu, D., Kang, X., Feng, Y., Zheng, Y., Zou, J., Chen, Y., Zeng, J., Zhang, Y.-Q., Pang, J., Liu, J., Wang, T., and Zhan, X. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025b.

Zheng, R., Liang, Y., Huang, S., Gao, J., Daum'e, H., Kolobov, A., Huang, F., and Yang, J. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *ArXiv*, abs/2412.10345, 2024.

Zhong, Z., Yan, H., Li, J., Liu, X., Gong, X., Zhang, T., Song, W., Chen, J., Zheng, X., Wang, H., and Li, H. Flowvla: Visual chain of thought-based motion reasoning for vision-language-action models. *arXiv preprint arXiv:2508.18269*, 2025.

Zhou, X., Xu, Y., Tie, G., Chen, Y., Zhang, G., Chu, D., Zhou, P., and Sun, L. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

A. Datasets and Experimental Details

A.1. Galaxea Open-World Dataset

We train on the Galaxea Open-World Dataset (Jiang et al., 2025), a large-scale real-world mobile-manipulation dataset with ∼500 hours of demonstrations (about 100K trajectories). The dataset spans 150 task categories across 50 real-world scenes and covers more than 1,600 objects and 58 skills. Importantly for our setting, data are collected under a single consistent embodiment, so that perception streams, action/state signals, and language annotations are naturally aligned for end-to-end VLA training.

Platform. All demonstrations are recorded on the Galaxea R1 Lite platform, which is also the real robot we use for our zero-shot evaluation in Sec. 4. R1 Lite is a mobile bimanual robot with a 23-DoF embodiment (two 6-DoF arms, a 3-DoF torso for workspace extension, a 6-DoF vector-drive omnidirectional base up to 1.5 m/s, and two 1-DoF grippers). The platform is equipped with a head stereo RGB camera for scene-level context and dual Intel RealSense D405 RGB-D wrist cameras for close-range manipulation.

A.2. Real-robot evaluation details

For each real-robot task, we use a simple rubric that measures whether the policy can complete the task within the fixed time budget used in Sec. 4. We report binary success/failure over 50 trials.

Store the dolls. The robot must pick up **three** plush dolls placed on a tabletop and put them into a designated storage container. Success is defined as all three dolls being fully inside the container at the end of the episode.

Arrange eggs. The robot must pick up **one** egg and place it into a designated slot of an egg carton. Success is defined as the egg being seated in the target slot without being dropped.

Put the flowers in the vase. The robot must grasp a flower and insert it into the opening of a vase. Success is defined as the flower being inside the vase and remaining stably placed at the end of the episode.

Put the pen into the pen holder. The robot must pick up **two** pens and insert them into a pen holder. Success is defined as both pens being inside the holder (stably contained) at the end of the episode.

Wipe the desktop. The robot must wipe a visible stain/dirty region on a desktop using a **cloth rag**. Success is defined as the target region being visibly cleaned (stain disappears or is substantially reduced) within the time budget.

Fold the clothes. The robot must fold a shirt starting from a flat or mildly wrinkled state on the table. Success is defined as completing the target fold (folding in sleeves and making a main fold) with the garment remaining on the working surface.

Pick up garbage on the ground. The robot must pick up **two** plastic bottles from the ground and dispose them into a trash bin. Success is defined as both bottles ending inside the bin.

Open the drawer. The robot must grasp the drawer handle and pull to open the drawer to a specified extent. Success is defined as the drawer being opened beyond a threshold without damaging the mechanism.

A.3. Training hyperparameters

To facilitate reproducibility, we summarize the key hyperparameters used in our simulation training (LIBERO) and real-robot training (Galaxea-500h). All simulation runs are executed on **4**×**H100** GPUs and are initialized from a pretrained VLM backbone (without any VLA data pretraining). For comprehensive implementation details, please refer to our codebase, which is provided in the Supplementary Material.

Simulation. Table 7 reports the configuration.

Table 7. **Hyperparameters for simulation datasets training.**

Configuration	Libero	WidowX	Google Robot
Batch size (per GPU)	64	80	80
Global batch size	64×4 = 256	80×4 = 320	80×4 = 320
Action head	Large (1024,24,16)	Large (1024,24,16)	Large (1024,24,16)
Action chunk horizon H	10	30	30
Image resize	128×128	224×224	224×224
Action normalization	on	on	on
Data shuffling	on	on	on
Optimizer	AdamW	AdamW	AdamW
Betas	(0.9, 0.95)	(0.9, 0.95)	(0.9, 0.95)
Weight decay	0.0	0.0	0.0
Learning rate	2e-4	1e-4	1e-4
VLM LR multiplier	0.1	0.1	0.1
Warm-up steps	0	0	0
Scheduler	none	none	none
Training steps	150K	50K	150K
Precision	bfloat16	bfloat16	bfloat16

Real-robot training (Galaxea-500h). Table 8 summarizes the training setup on the Galaxea Open-World Dataset. For a fair comparison, both our method and $\pi_{0.5}$ are trained using the same Galaxea data and the same hyperparameters, with the same compute budget: **64×H100** GPUs for **150K** training steps.

Table 8. **Hyperparameters for Galaxea-500h training.**

Configuration	Value
Batch size (per GPU)	32
Global batch size	32×64=2048
Action head	Large (1024,24,16)
Action chunk horizon H	30
Image resize	224×224
Action normalization	on
Data shuffling	on
Optimizer	AdamW
Betas	(0.9, 0.95)
Weight decay	0.0
Learning rate	1e-4
VLM LR multiplier	0.1
Warm-up steps	1000
Scheduler	none
Training steps	150K
Precision	bfloat16