FLATVLA: ROBUST VISION-LANGUAGE-ACTION MODEL WITH SHARPNESS-AWARE MINIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language-Action (VLA) models integrate visual observations and language instructions to generate actions, yet remain vulnerable to multimodal distribution shifts and errors arising during sequential execution. Existing robustness methods target predefined variations, making it difficult to cover a broad range of variations across VLA inputs and actions. To address this challenge, we propose FlatVLA, a flatness-aware training framework that adapts Sharpness-Aware Minimization (SAM) to the sequential structure of VLA action prediction. FlatVLA applies larger weights to positions near the beginning of each action chunk, as errors at these positions can affect a larger portion of the remaining prediction. Our theoretical analysis connects local smoothness to policy sensitivity and compounding error, motivating this sequential weighting. Across recent models, FlatVLA shows improvements on performance and robustness across diverse evaluation factors and settings, with further analysis of model behaviors. These gains also extend from simulation to real-world tasks, as demonstrated via hardware experiments. Overall, FlatVLA optimizes parameter space smoothness while accounting for sequential error accumulation in action prediction, improving robustness without explicitly tailoring training to specific deployment conditions.

1 INTRODUCTION

Deep learning has increasingly moved toward foundation models in Computer Vision (CV) and Natural Language Processing (NLP), where transformer-based architectures with attention (Bahdanau et al., 2015; Vaswani et al., 2017; Dosovitskiy et al., 2021) and large-scale pretraining (Devlin et al., 2019; Liu et al., 2019) have enabled transfer across a broad range of tasks and modalities. This trend has led to general-purpose models such as GPT (Brown et al., 2020; OpenAI, 2023) and Gemini (Gemini Team et al., 2023), and has recently extended to multimodal foundation models such as Vision-Language Models (VLMs) (Radford et al., 2021). These models learn shared representations across visual and linguistic modalities, enabling strong transfer to a wide range of downstream tasks.

As multimodal foundation models have evolved into more complex systems, such as VLA models, their outputs are no longer limited to visual or language understanding but can directly influence downstream decision-making and system behavior. Such systems can become more sensitive to errors or distribution shifts, as their effects may accumulate through repeated interactions with the environment. Small variations in visual observations (*e.g.*, camera viewpoint) or disturbances during execution (*e.g.*, action noise) can alter policy outputs and accumulate via closed-loop control, resulting in unstable trajectories or task failures (Ross & Bagnell, 2010). Several studies and benchmarks also have shown substantial performance degradation under physical and multimodal distribution shifts, highlighting a gap between standard evaluation and real-world deployment. (Fei et al., 2026).

Recent efforts have therefore focused on under such variations, prior work has largely focused on broadening the range of components encountered during training. A prominent approach is to pretrain generalist robot policies on large-scale datasets covering diverse tasks, environments, and robot embodiments (Collaboration et al., 2024), following the broader trend of scaling robot data and model capacity (Yu et al., 2023). Alongside this paradigm, recent studies have explored a range of action modeling and training strategies, including discretized action tokens with behavior cloning, diffusion-based policies, flow matching, and reinforcement learning-based post-training (Kim et al.,

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

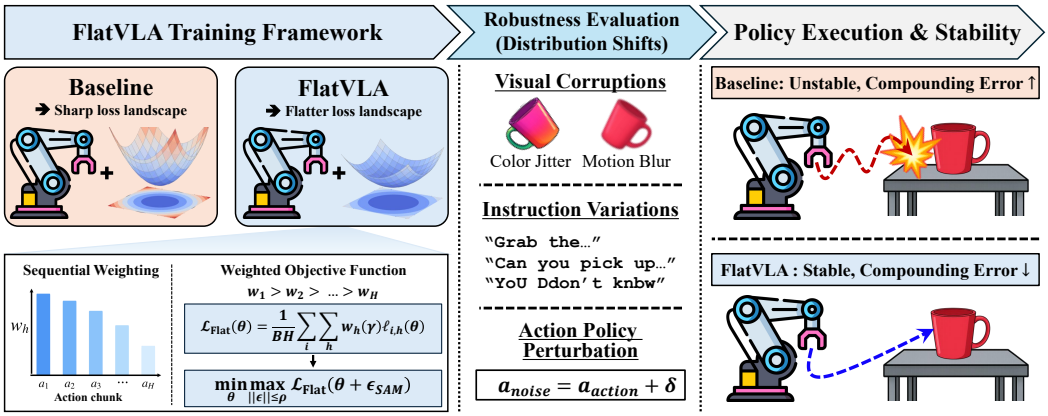


Figure 1: Overview of **FlatVLA**. Baseline often leads to unstable policies and increase compounding errors, while FlatVLA leads to stable policy behavior that mitigates compounding errors.

2025b; Ghosh et al., 2024; Black et al., 2025b; Zhang et al., 2026). Together, these advances have broadened the range of tasks and conditions that pretrained robot policies can handle.

Despite their effectiveness, these approaches remain tied to the variations specified during training. This dependence poses a specific challenge for VLA models, as potential deployment shifts can arise across visual observations, language instructions, and action spaces. For example, adversarial approaches applying to a VLA model (Guo et al., 2026) require specifying representative "worst-case" perturbations for multimodal inputs, which may not capture the diverse and unforeseen shifts encountered in real-world settings (Lechner et al., 2021). Consequently, anticipating and covering all relevant variations across these heterogeneous spaces and various circumstances remains difficult.

To address this challenge, we shift our perspective from explicitly accounting for individual factors to the parameter space, focusing on *flatness optimization* of the learned solution. We hypothesize that optimizing such parameter-space solution could provide a system-level approach to improve the overall VLA policy. There have been prior efforts to apply flatness-aware optimization to VLA models (Zhang & Bisk, 2026); however, its broader role in system-level robustness and policy behavior remains underexplored. Motivated by this gap, we thus seek a principled understanding of VLA models through parameter-space flatness, examining how it relates to their intrinsic policy behavior.

Accordingly, as illustrated in Figure 1, we propose FlatVLA, a flatness-aware VLA framework that connects *parameter-space flatness* to *policy sensitivity and sequential error propagation*. Specifically, we establish a theoretical connection between local loss smoothness and policy sensitivity, showing that smoother local loss landscapes provide a tighter upper bound on compounding error under the same conditions and one-step imitation-error budget. Guided by this analysis, we then propose FlatVLA that employs Sharpness-Aware Minimization (SAM) (Foret et al., 2021) together with a theory-guided sequential weighting strategy that places greater SAM pertrubation on earlier predictions, as errors at earlier steps can influence more later action behaviors and decisions. We evaluate FlatVLA across VLA benchmarks and robustness settings, and further analyze the resulting policy behaviors. Thus, our contributions are threefold:

1. We propose FlatVLA that promotes flatter minima by using a theory-inspired approach that applies sequential weighting to emphasize earlier stages of each action horizon in VLA tasks.
2. We establish a theoretical link between local loss smoothness and compounding error, showing that smoother landscapes yield no larger compounding-error upper bound.
3. We evaluate FlatVLA under multimodal distribution shifts and action perturbations, and real-world evaluation to demonstrate consistent robustness gains and more stable policy behavior.

2 Related Works

2.1 Vision-Language-Action and Imitation Learning

Recent research on VLA models has increasingly shifted from task-specific policies toward generalist, instruction-conditioned policies that integrate visual perception, language grounding, and action

generation. Beyond architectural advances, many large-scale VLA models are trained under an imitation learning paradigm using transformer-based sequence modeling (Bjorck et al., 2025). Several VLA backbones formulate robot control as supervised prediction of discretized action tokens or action sequences, providing a scalable and practical framework for downstream fine-tuning (Kim et al., 2025b;a). At the same time, the increasing scale and diversity of datasets have enabled stronger cross-task and -embodiment generalization (Collaboration et al., 2024) via training on trajectories spanning a broad range of tasks and embodiments. In this work, we specifically focus on imitation-learning-based fine-tuning of foundation VLA models with generalist capabilities. This results in evaluating whether the proposed method further improves downstream adaptation and robustness.

2.2 Robustness in VLA

Robustness in Deep Learning Model robustness broadly concerns whether a trained model maintains reliable behavior when inputs deviate from the training distribution, resulting in a generalization gap. Many approaches have been proposed in two forms to reduce this gap: data- and parameter-oriented. For example, Adversarial Training (AT) (Madry et al., 2018) explicitly optimizes for worst-case perturbations within a specified threat model, improving robustness to strong input noise. More recently, Sharpness-Aware Minimization (SAM) (Foret et al., 2021) has emerged as a prominent optimizer that seeks flatter landscapes by minimizing loss within the parameter proximal region, thereby enhancing robustness against distribution shifts, and has been applied across diverse domains such as LLMs (Bahri et al., 2022), and generative modeling (Lee et al., 2025). These two perspectives can be theoretically unified with a duality analysis between AT and SAM (Zhang et al., 2024), further generalizing to data augmentations and flatter minima (Yoo & Yoon, 2026).

Robust Training for VLA Models Recent studies show that VLA policies can degrade under distribution shifts under compounding error during closed-loop execution and alter later states (Ross & Bagnell, 2010). These shifts span visual factors and language variations, with RobustVLA (Guo et al., 2026) improving robustness to visual and language perturbations through AT. Beyond such input space approaches, there have been prior efforts to apply SAM to VLA models from the parameter space perspective, primarily focusing on preserving instruction-following performance (Zhang & Bisk, 2026). However, these approaches leave limitations. For instance, RobustVLA requires an additional search for worst-case inputs with predefined perturbation families. Moreover, Zhang & Bisk (2026) provide limited analyses of its broader effects of smoothness on action behavior and other aspects of policies. This motivates our approach to take a *system-level parameter space perspective*, optimizing the shared policy parameters rather than introducing separate robustness mechanisms for individual sources of variation. Beyond applying SAM, our work, FlatVLA, also *theoretically connects local loss smoothness to policy sensitivity and compounding error* with *detailed analyses of multimodal and action robustness, and action behavior* across simulation and real-world settings.

3 Towards Robust VLA: A Theoretical Analysis of Smoothness

In this section, we provide a theoretical framework to analyze how flatter landscape mitigates compounding errors in VLA policies. Our analysis characterizes how one step imitation errors accumulate during sequential execution and derives an upper bound that depends on local smoothness. Under this local conditions, a smoother landscape leads to a tighter compounding-error upper bound.

3.1 Preliminaries

Notations for Imitation Learning Consider a VLA policy $\pi_\theta : \mathcal{X} \to \mathcal{A}$ parameterized by $\theta \in \mathbb{R}^d$, where $\mathcal{X}$ and $\mathcal{A}$ denote the input and action spaces, respectively. We define the input space as $\mathcal{X} = \mathcal{O} \times \mathcal{T}$, where $\mathcal{O}$ and $\mathcal{T}$ denote the visual observation and language instruction spaces, respectively. Each input $\mathbf{x} = (\mathbf{o}, \mathbf{t})$ comprises a visual observation $\mathbf{o} \in \mathcal{O}$ and a language instruction $\mathbf{t} \in \mathcal{T}$. Given an expert policy $\pi^* : \mathcal{X} \to \mathcal{A}$ and a training dataset $\mathcal{D}_{\text{train}} = \{(\mathbf{x}^{(i)}, \mathbf{a}^{*,(i)})\}_{i=1}^N$, the policy is trained to imitate the expert actions $\mathbf{a}^{*,(i)} = \pi^*(\mathbf{x}^{(i)})$. For a finite-horizon task with $T \geq 2$, let $t \in \{1, \dots, T\}$ denote the timestep of sequential interaction with the environment. At timestep t, let $s_t \in \mathcal{S}$ denote the environment state and $\mathbf{x}_t \in \mathcal{X}$ the corresponding policy input, and we define the sequential variable as $z_t = (s_t, \mathbf{x}_t) \in \mathcal{Z} := \mathcal{S} \times \mathcal{X}$. The expert and learned policies induce the time-indexed distributions $\mathcal{D}_{*,t}$ and $\mathcal{D}_{\theta,t}$ over $\mathcal{Z}$, from which $z_t^* \sim \mathcal{D}_{*,t}$ and $z_t \sim \mathcal{D}_{\theta,t}$, respectively.

Sequential Distribution Deviation and Imitation Error We measure the deviation between the expert and learned-policy distributions at each timestep as $\delta_t(\theta) = W(\mathcal{D}_{*,t}, \mathcal{D}_{\theta,t})$, where W denotes the Wasserstein distance. The compounding error over the task horizon is defined as $C_T(\theta) = \sum_{t=1}^{T} \delta_t(\theta)$. We then define the one-step imitation error as $\epsilon_t(\theta) = \mathbb{E}\left[\|\pi_\theta(\mathbf{x}_t^*) - \mathbf{a}_t^*\|\right]$. While $\delta_t(\theta)$ captures the deviation accumulated up to timestep t, $\epsilon_t(\theta)$ measures the new action error at the expert input $\mathbf{x}_t^*$. For paired expert and learned-policy samples at timestep t, the action discrepancy satisfies

$$\|\mathbf{a}_t - \mathbf{a}_t^*\| \leq \underbrace{\|\pi_\theta(\mathbf{x}_t) - \pi_\theta(\mathbf{x}_t^*)\|}_{\text{error due to propagated input deviation}} + \underbrace{\|\pi_\theta(\mathbf{x}_t^*) - \mathbf{a}_t^*\|}_{\text{new imitation error at timestep } t}. \tag{1}$$

The first term reflects deviations propagated from previous interactions, whereas the second represents the new imitation error at timestep t. This decomposition characterizes how existing deviation and newly introduced imitation error contribute to covariate shift and compounding errors in imitation learning (Ross & Bagnell, 2010; Ross et al., 2011).

Local Smoothness of Sequential Dynamics Let $\mathcal{P}_t(\cdot \mid z, \mathbf{a})$ denote the conditional distribution of z_{t+1} given the current sequential variable z and action $\mathbf{a}$. We introduce the following local smoothness assumption for sequential execution in imitation learning.

Assumption 1 (Local Smoothness (Makdah et al., 2022))**.** *For sequential variables and actions in the local region, the conditional next-step distribution is locally Lipschitz with respect to z and $\mathbf{a}$:*

$$W\left(\mathcal{P}_t(\cdot \mid z, \mathbf{a}), \mathcal{P}_t(\cdot \mid z', \mathbf{a}')\right) \leq L_{z,t} d_{\mathcal{Z}}(z, z') + L_{a,t}\|\mathbf{a} - \mathbf{a}'\|, \tag{2}$$

Since $z = (s, \mathbf{x})$ contains both state and policy input, we assume that the metric on $\mathcal{Z}$ upper bounds as $\|\mathbf{x} - \mathbf{x}'\| \leq d_{\mathcal{Z}}((s, \mathbf{x}), (s', \mathbf{x}'))$. We define the local input sensitivity of the policy as

$$K_t(\theta) = \sup_{\mathbf{x} \in \mathcal{R}_t} \|J_{\mathbf{x}} \pi_\theta(\mathbf{x})\|. \tag{3}$$

Here, $J_{\mathbf{x}}\pi_\theta(\mathbf{x})$ denotes the Jacobian of the policy output with respect to its input $\mathbf{x}$. By the definition of $K_t(\theta)$ and the metric condition above, this discrepancy induced by the existing input deviation can be bounded as $\|\pi_\theta(\mathbf{x}) - \pi_\theta(\mathbf{x}')\| \leq K_t(\theta) d_{\mathcal{Z}}(z, z')$. Thus, the existing deviation affects the next-step distribution directly through $L_{z,t}$ and indirectly through the resulting action discrepancy, scaled by $L_{a,t}K_t(\theta)$. We then define the sequential error amplification factor as $\alpha_t(\theta) = L_{z,t} + L_{a,t}K_t(\theta)$.

Lemma 1 (Sequential Error Bound)**.** *Under Assumption 1, the distribution deviation satisfies*

$$\delta_{t+1}(\theta) \leq \alpha_t(\theta)\delta_t(\theta) + L_{a,t}\epsilon_t(\theta). \tag{4}$$

Proof. See Appendix A.2. □

According to Lemma 1, the deviation at each timestep is controlled by two terms: the existing distribution deviation propagated to the next timestep through $\alpha_t(\theta)$ and the newly introduced imitation error $\epsilon_t(\theta)$. Consequently, even when two policies have comparable one-step imitation errors, their long-horizon errors can differ substantially due to the error amplification factor $\alpha_t(\theta)$.

3.2 Local Smoothness and Policy Sensitivity

Local Smoothness of SAM Let $\mathcal{L}_{\mathrm{IL}}(\theta) = \mathbb{E}_{(\mathbf{x},\mathbf{a}^*)\sim\mathcal{D}_{\text{train}}}[\ell(\pi_\theta(\mathbf{x}), \mathbf{a}^*)]$ denote the imitation-learning objective. For a radius $\rho > 0$, we quantify its local smoothness as

$$S_\rho(\theta) = \max_{\|\varepsilon\|_2 \leq \rho} \left[\mathcal{L}_{\mathrm{IL}}(\theta + \varepsilon) - \mathcal{L}_{\mathrm{IL}}(\theta)\right]. \tag{5}$$

This measures the local worst-case loss increase considered by Sharpness-Aware Minimization (SAM). Through the Generalized Gauss–Newton (GGN) structure, local loss curvature can be related to policy-output sensitivity via model-output Jacobians (Martens, 2020; Dangel et al., 2020), motivating a smoothness-dependent upper bound on $K_t(\theta)$ derived in Appendix A.3.

Proposition 1 (Local Smoothness and Error Amplification)**.** *Under Assumption 3, 4, and 5 in Appendix A.3, let $L_H \geq 0$ denote the local Hessian-smoothness constant, and let $A_t, B_t \geq 0$ denote*

the local sensitivity terms. Then, the input sensitivity satisfies $K_t(\theta) \le A_t\sqrt{\frac{2S_\rho(\theta)}{\rho^2} + \frac{L_H\rho}{3}} + B_t$. *Consequently, the error amplification factor is bounded as*

$$\alpha_t(\theta) \le \bar{\alpha}_t(\theta) = L_{z,t} + L_{a,t}\left[A_t\sqrt{\frac{2S_\rho(\theta)}{\rho^2} + \frac{L_H\rho}{3}} + B_t\right]. \tag{6}$$

Proof. See Appendix A.3. □

3.3 Main Theorem: Compounding-Error Bound under Local Smoothness

Compounding-Error Upper Bound Let $\bar{\alpha}(\theta) = \max_{t<T}\bar{\alpha}_t(\theta)$ and $\beta = \max_{t<T} L_{a,t}$ denote the *maximum error-amplification factor* and *action-sensitivity constant over the horizon*, respectively, and assume $\max_{t<T}\epsilon_t(\theta) \le \epsilon$. Then, Lemma 1 and Proposition 1 yield the upper-bound recurrence $\bar{\delta}_1 = 0$ and $\bar{\delta}_{t+1} = \bar{\alpha}(\theta)\bar{\delta}_t + \beta\epsilon$, where $\delta_t(\theta) \le \bar{\delta}_t$. Unrolling this recurrence gives the finite-horizon upper bound, with the detailed derivation provided in Appendix A.4:

$$C_T(\theta) \le \bar{C}_T(\theta;\epsilon) = \beta\epsilon\sum_{j=0}^{T-2}(T-1-j)\bar{\alpha}(\theta)^j. \tag{7}$$

Theorem 1 (Compounding-Error Bound under Local Smoothness)**.** *Under the shared local conditions in Appendix A.5, let* π_θ *and* $\pi_{\theta'}$ *have the same one step imitation-error bound* ϵ. *If the* $\pi_{\theta'}$ *has a smoother landscape, it leads to a tighter compounding-error upper bound:*

$$S_\rho(\theta') \le S_\rho(\theta) \quad \Longrightarrow \quad \bar{C}_T(\theta';\epsilon) \le \bar{C}_T(\theta;\epsilon). \tag{8}$$

Proof. See Appendix A.5. □

Remark 1.1 (Why Local Smoothness Matters)**.** *Under the same one step imitation-error budget, the policy with a smoother (local) loss landscape has a compounding-error upper bound that is at least same or smaller. Intuitively, a one step imitation error affects not only the current action but also the subsequent inputs encountered by the policy, allowing the error to be repeatedly amplified over time. A smoother local objective constrains the policy sensitivity and reduces this amplification, thereby limiting how rapidly errors accumulate during sequential execution. Therefore, Theorem 1 explains why the local smoothness encouraged by SAM can mitigate compounding errors in VLA policies.*

4 FlatVLA: Sequentially Weighted Sharpness-Aware Minimization

In this section, we introduce FlatVLA, which adapts SAM to the sequential structure of VLA action prediction. While SAM promotes a smoother local objective, it treats all positions within a predicted action chunk equally. Our analysis in Section 3 suggests that errors occurring earlier in the action horizon have more opportunities to influence later decisions. Thus, FlatVLA incorporates this structure by assigning sequential weights to the imitation loss, emphasizing on earlier action positions. The detailed learning procedure is provided in Appendix B.2.

Compounding-Error Perspective on Action Horizons As shown by Eq. (4), one step imitation error can affect later decisions as the policy proceeds through a task. In principle, constructing a training weight that captures this effect would require estimating these amplification factors along the policy rollout, together with the remaining episode horizon for each training sample. These quantities are unrealistic in stochastic offline training, where VLA models are typically trained on randomly sampled action chunks without their absolute timesteps within an episode.

We apply this connection within each predicted action horizon. Since the factors $\alpha_t(\theta)$ are not directly applicable during training, we approximate their local effect within an action chunk using a constant coefficient $\gamma \ge 0$ as a surrogate . Under this approximation, an imitation error introduced at position h can influence the remaining action positions through factors, yielding the weights

$$\widetilde{w}_h(\gamma) = \sum_{j=0}^{H-h}\gamma^j, \qquad w_h(\gamma) = \frac{H\widetilde{w}_h(\gamma)}{\sum_{r=1}^{H}\widetilde{w}_r(\gamma)}. \tag{9}$$

Sequentially Weighted SAM for VLA Consider a minibatch of size B, and let $\ell_{i,h}(\theta)$ denote the native imitation-learning loss for sample i at action position h. FlatVLA leaves this underlying loss unchanged and instead reweights how losses are aggregated across the action horizon. Specifically, we define the sequentially weighted imitation objective as

$$\mathcal{L}_{\text{Flat}}(\theta) = \frac{1}{BH} \sum_{i=1}^{B} \sum_{h=1}^{H} w_h(\gamma) \ell_{i,h}(\theta). \tag{10}$$

This normalization preserves the overall loss scale while placing greater emphasis on action positions with more opportunities to influence later actions. FlatVLA then applies standard SAM (Foret et al., 2021) to $\mathcal{L}_{\text{Flat}}$, using the same horizon-weighted objective for both the perturbation and parameter-update steps.

Role of γ As discussed above, γ is motivated by the recurrence in Eq. (4), where $\alpha_t(\theta)$ determines how strongly a one-step error propagates to the next timestep. Because this timestep-specific factor is unavailable for each offline training sample, γ serves as a shared local proxy that controls how strongly the weighting reflects sequential error amplification within the action horizon. The role of γ can be understood through several limiting cases. For example, setting $\gamma = 0$ gives uniform weights and reduces FlatVLA to standard SAM, and the same reduction holds for $H = 1$ regardless of γ. As γ increases, the weights increasingly favor earlier positions, with a moderate preference for $0 < \gamma < 1$. When $\gamma = 1$, the normalized weights decrease linearly with the action-horizon position, yielding $w_h(1) = 2(H - h + 1)/(H + 1)$. Finally, for $\gamma > 1$, the emphasis on earlier positions becomes stronger. Since $w_1/w_H = \sum_{j=0}^{H-1} \gamma^j$, this effect also depends on the action horizon H. Accordingly, γ *controls how the imitation loss is distributed across the action horizon*, whereas ρ controls the range of the parameter-space local neighborhood considered by SAM-related method.

5 Experimental Results

In this section, we evaluate FlatVLA, our SAM-based training method, across several VLA architectures for robotic manipulation. For each architecture, FlatVLA denotes the model trained with the proposed sequential flatness-aware objective. We examine how flatter minima in VLA objective affect robustness, and the stability of policy behavior. Additional implementation details, experimental settings, benchmark descriptions and evaluation protocols are provided in Appendix B.

5.1 Experimental Settings

VLA Baseline Models We consider four widely used VLA models for general-purpose robotic manipulation: OpenVLA (Kim et al., 2025b), OpenVLA-OFT (Kim et al., 2025a), π_0 (Black et al., 2025b), and $\pi_{0.5}$ (Black et al., 2025a). Following the original fine-tuning protocol, we apply LoRA (Hu et al., 2022) with rank $r = 32$ to OpenVLA and OpenVLA-OFT. All models are initialized from checkpoints pretrained on the Open-X-Embodiment dataset (Collaboration et al., 2024), which contains approximately 970K robot manipulation trajectories spanning diverse tasks and embodiments. Since OpenVLA predicts one action at a time ($H = 1$), FlatVLA applies standard SAM. Otherwise, we apply horizon-dependent weighting to account for their sequential structure.

Benchmark and Evaluation We evaluate task performance on LIBERO (Liu et al., 2023), a widely used simulation benchmark for lifelong robot learning with four suites: Spatial, Object, Goal, and 10. These suites cover spatial relations, object interactions, goal-conditioned behavior, and diverse long-horizon tasks, allowing FlatVLA to be evaluated across different forms of vision-and-language-conditioned manipulation. We further conduct hardware experiments to examine whether the improvements observed in simulation extend to real-world experiments.

5.2 LIBERO Benchmark Evaluation

In this section, we provide conducted experiments, reported on Table 1, whether FlatVLA consistently improves the corresponding VLA baseline across four LIBERO suites. Especially, FlatVLA improves all four VLA baselines on every suite, showing that the gains are consistent across different model architectures and task settings. Averaged across the suites, FlatVLA improves success

Table 1: Success rates on the LIBERO benchmark suites.

Suite	OpenVLA		OpenVLA-OFT		π_0		$\pi_{0.5}$	
	Baseline	**FlatVLA**	Baseline	**FlatVLA**	Baseline	**FlatVLA**	Baseline	**FlatVLA**
SPATIAL	85.8 ± 0.6	$\mathbf{87.2 \pm 0.4}$	97.9 ± 0.4	$\mathbf{98.3 \pm 0.5}$	96.5 ± 0.2	$\mathbf{97.2 \pm 0.5}$	97.7 ± 0.6	$\mathbf{98.5 \pm 0.2}$
OBJECT	87.6 ± 1.4	$\mathbf{91.2 \pm 0.2}$	98.3 ± 0.1	$\mathbf{99.0 \pm 0.0}$	96.7 ± 0.3	$\mathbf{97.6 \pm 0.5}$	97.5 ± 0.3	$\mathbf{99.2 \pm 0.3}$
GOAL	78.0 ± 1.2	$\mathbf{80.6 \pm 0.6}$	96.5 ± 0.9	$\mathbf{97.5 \pm 0.4}$	95.3 ± 1.0	$\mathbf{96.1 \pm 0.3}$	95.9 ± 0.4	$\mathbf{97.5 \pm 0.2}$
10	51.8 ± 1.8	$\mathbf{54.8 \pm 1.0}$	94.3 ± 0.4	$\mathbf{94.9 \pm 0.7}$	84.6 ± 1.6	$\mathbf{86.1 \pm 1.6}$	92.2 ± 1.0	$\mathbf{94.3 \pm 0.8}$
Average	75.80	**78.45** (+2.65)	96.75	**97.43** (+0.68)	93.28	**94.25** (+0.98)	95.83	**97.39** (+1.56)

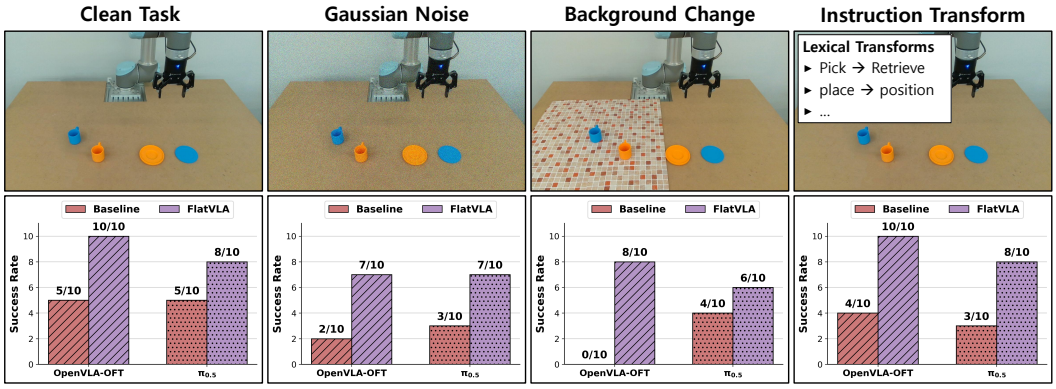


Figure 2: Real-world robotic setup and evaluation results under clean and perturbed settings, including visual Gaussian noise, background changes, and lexical instruction transformations.

rates by 2.65%, percentage points, for OpenVLA, 0.68% for OpenVLA-OFT, 0.98% for π_0, and 1.56% for $\pi_{0.5}$. At the suite level, the improvements range from $+0.4\% \rightarrow +3.6\%$, with the largest gain achieved by OpenVLA on LIBERO-OBJECT. Moreover, even though OpenVLA-OFT and $\pi_{0.5}$ already achieve high success rates on standard LIBERO, FlatVLA further improves their performance. Overall, these results show that flatness-aware training improves task performance across a range of VLA architectures, including both single-action and horizon policies.

5.3 REAL-WORLD EVALUATION

Hardware and Evaluation Setup To examine whether the improvements observed in simulation extend to real-world manipulation, we conduct additional experiments in a real-world robotic setting. We deploy the baseline and FlatVLA variants of OpenVLA-OFT and $\pi_{0.5}$ on a 6-DoF UR5 robotic arm equipped with a Robotiq 2F-85 parallel gripper. Visual observations are captured from a third-person view using an Intel RealSense D435i camera. The evaluation considers the task instruction *"pick up the [color] cup and place it on the [color] plate"*, where the colors are either blue or orange. In addition to the clean setting, we evaluate robustness under Gaussian noise, background changes, and lexical transformations of the instruction (Guo et al., 2026). These perturbations reflect practical variations that can arise in real-world deployment, including sensor noise, environmental changes, and different expressions of the same instruction. Gaussian noise and background changes perturb the visual observations, while lexical transformations perturb the language instructions. Detailed descriptions of the evaluation setup and robustness settings are provided in Appendix B.4.

Real-World Performance and Robustness As shown in Figure 2, FlatVLA achieves higher success rates than the baseline for both OpenVLA-OFT and $\pi_{0.5}$ across the clean and perturbed settings. For each setting, we conduct 10 trials, with five trials for each color configuration (blue and orange). In the clean setting, FlatVLA achieves 10/10 successes with OpenVLA-OFT and 8/10 with $\pi_{0.5}$, compared with 5/10 for both baselines. Under Gaussian action noise, the success rate increases from 2/10 $\rightarrow$ 7/10 for OpenVLA-OFT and from 3/10 $\rightarrow$ 7/10 for $\pi_{0.5}$. With background changes, OpenVLA-OFT improves from 0/10 $\rightarrow$ 8/10, while $\pi_{0.5}$ improves from 4/10 $\rightarrow$ 6/10. Under lexical transformations of the instruction, the success rate increases from 4/10 $\rightarrow$ 10/10 for OpenVLA-OFT and from 3/10 $\rightarrow$ 8/10 for $\pi_{0.5}$. These results show that the improvements in simulations are consistently observed in real robotics evaluations, with even more pronounced gains. Thus, our flatness-aware optimization, FlatVLA, improves both clean-task performance and robustness to practical variations in visual observations and language instructions.

Table 2: Success rates on the LIBERO-PLUS benchmark. The all success rate including 'Average' is weighted by the number of tasks in each variation. Detail information are provided in Appendix B.

Method	Optim.	Camera	Robot	Language	Light	Background	Noise	Layout	Average
OpenVLA	Baseline	0.98	3.82	28.77	4.39	25.74	20.11	31.98	16.31
	FlatVLA	**1.55**	**4.85**	**38.26**	**6.39**	**29.59**	**22.41**	**33.62**	**19.05**
OpenVLA-OFT	Baseline	54.40	30.51	71.24	**92.29**	91.82	66.52	79.86	67.43
	FlatVLA	**55.15**	**36.32**	**74.11**	91.86	**92.12**	**73.01**	**81.39**	**69.98**
π_0	Baseline	13.75	6.65	62.20	86.42	82.24	**20.24**	72.52	45.67
	FlatVLA	**16.89**	**6.78**	**66.81**	**88.17**	**82.62**	19.66	**73.90**	**47.26**
$\pi_{0.5}$	Baseline	64.35	72.77	81.13	93.61	91.45	80.45	85.31	80.52
	FlatVLA	**65.29**	**78.06**	**84.52**	**94.92**	**94.80**	**81.45**	**86.56**	**82.56**

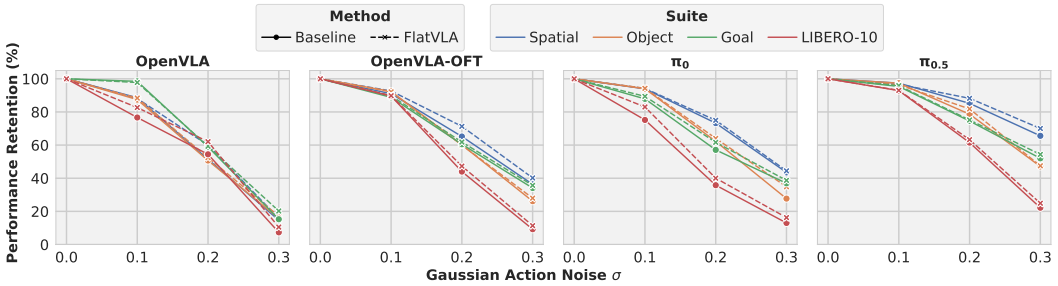


Figure 3: Performance retention under Gaussian action noise across all LIBERO suites. Retention is normalized by the corresponding clean success rate at $\sigma = 0$, with higher values indicating greater robustness. Solid lines with circles are the baseline, while dashed lines with crosses are FlatVLA.

6 Empirical Analysis

6.1 Multimodal Input Variation Robustness

We evaluate robustness to multimodal input variations on LIBERO-PLUS (Fei et al., 2026), which introduces changes to the camera, robot, language instruction, lighting, background, visual noise, and object layout. As shown in Table 2, FlatVLA heightens the success rate for all four VLA models, leading to the improvements from 16.31% to 19.05% for OpenVLA, 67.43% to 69.98% for OpenVLA-OFT, 45.67% to 47.26% for π_0, and 80.52% to 82.56% for $\pi_{0.5}$. The gains also span different variation types, including language changes for OpenVLA ($28.77\% \rightarrow 38.26\%$), visual noise for OpenVLA-OFT ($66.52\% \rightarrow 73.01\%$), and robot appearance for $\pi_{0.5}$ ($72.77\% \rightarrow 78.06\%$). Consequently, although marginal gains or minor reductions occurs in certain settings, the overall results, computed as a weighted average based on the number of tasks in each variation, indicate performance improvements across all models; OpenVLA, OpenVLA-OFT π_0, and $\pi_{0.5}$ achieve performance gains of $+2.74\%$, $+2.55\%$, $+1.59\%$, and $+2.04\%$ percentage points, respectively.

6.2 Action Noise Perturbation Robustness

We assess sensitivity to action perturbations by injecting zero-mean Gaussian noise into the executed continuous actions during evaluation. Specifically, each continuous action vector $\mathbf{a}_{\text{action}}$ is perturbed as $\mathbf{a}_{\text{noise}} = \mathbf{a}_{\text{action}} + \delta_{\text{noise}}$, where $\delta_{\text{noise}} \sim \mathcal{N}(0, \sigma^2 I)$ and σ controls the perturbation magnitude. We evaluate $\sigma \in \{0.1, 0.2, 0.3\}$, selected according to the typical scale of the continuous action values. We exclude the gripper dimension because it represents a binary grasping decision. For each model and LIBERO suite, we compute performance retention separately for the baseline and FlatVLA as $R(\sigma) = 100 \times \text{SR}(\sigma)/\text{SR}(0)$, where $\text{SR}(\sigma)$ denotes the success rate at noise level σ and $\text{SR}(0)$ denotes the corresponding clean success rate.

As shown in Figure 3, performance retention decreases for both methods as σ increases, while FlatVLA generally retains a larger proportion of its clean performance. The gains are more evident under stronger perturbations and appear across both single-action and horizon-based policies, although some differences remain small under mild noise. Since retention is normalized by each method's clean success rate, these curves do not directly show FlatVLA's higher absolute success rates under noise. This trend is consistent with the analysis in Section 3, where an action error can

Table 3: Decision-level uncertainty averaged across the four LIBERO suites. OpenVLA uses logit entropy $H(p)$, while π-series models use sampling-based action uncertainty U_t. Gap Ratio denotes the relative uncertainty gap, $(\text{Failure} - \text{Success})/\text{Success} \times 100$, expressed as a percentage.

Case	**OpenVLA** (Metric: $H(p)$)		$\boldsymbol{\pi_0}$ (Metric: U_t)		$\boldsymbol{\pi_{0.5}}$ (Metric: U_t)	
	Baseline	FlatVLA	Baseline	FlatVLA	Baseline	FlatVLA
Success	0.353	0.344	0.026	0.026	0.073	0.056
Failure	0.392	0.457	0.059	0.071	0.112	0.125
Gap Ratio	11.0%	**32.8%**	126.9%	**173.1%**	53.4%	**123.2%**

affect subsequent states and decisions, linking robustness to action noise with reduced compounding error. The absolute success rates for each model, suite, and noise level are provided in Appendix C.2.

6.3 Decision-Level Analysis of the Action Policy

Predictive Uncertainty of VLA Policies Task success alone provides limited insight into how a VLA policy behaves as execution becomes difficult or eventually fails. We complement the rollout evaluation with a decision-level analysis, showing how predictive uncertainty differs between successful and failed executions. Since VLA models employ different action parameterizations, we use an uncertainty measure suited to each policy architecture rather than enforcing a single estimator.

For OpenVLA, which predicts discretized action tokens through categorical logits p, we measure the entropy $H(p)$ of the output distribution as a direct proxy for predictive uncertainty. For the generative $\boldsymbol{\pi}$-series, which does not provide categorical logits, we repeatedly sample real-valued action chunks from the same observation and measure their dispersion directly in the action space. Given K sampled action chunks $A_t^{(k)} \in \mathbb{R}^{H\times D}$, we compute the mean action at each horizon and action dimension as $\bar{A}_{t,h,d} = \frac{1}{K}\sum_{k=1}^{K} A_{t,h,d}^{(k)}$. We then define the uncertainty at timestep t as $U_t = \frac{1}{KHD}\sum_{k=1}^{K}\sum_{h=1}^{H}\sum_{d=1}^{D}\left(A_{t,h,d}^{(k)} - \bar{A}_{t,h,d}\right)^2$. A larger U_t indicates greater disagreement among action predictions generated from the same observation. We compare the baseline and FlatVLA using five randomly sampled successful and five failed trajectories from each LIBERO suite, avoiding imbalance between the two cases. We exclude OpenVLA-OFT, as its L1-regression action head does not naturally expose a predictive distribution suitable for uncertainty estimation.

Results Table 3 compares the uncertainty gap relative to the uncertainty during successful executions, accounting for differences in this reference value between methods. The gap ratio increases from 11.0% to 32.8% for OpenVLA, from 126.9% to 173.1% for π_0, and from 53.4% to 123.2% for $\pi_{0.5}$ under FlatVLA. Across all three models, FlatVLA shows higher predictive uncertainty during failed executions, while uncertainty during successful executions remains similar or lower. For π_0, uncertainty during successful executions remains at 0.026, whereas failure-case uncertainty increases from 0.059 to 0.071, showing that the larger ratio is not solely due to a lower reference value. Together, these results indicate a more pronounced relative increase in predictive uncertainty from successful to failed executions under FlatVLA, rather than an overall increase across both outcomes.

7 Conclusion

In this paper, we propose FlatVLA, a training framework for Vision-Language-Action (VLA) models that leverages Sharpness-Aware Minimization (SAM) to promote flatter minima, thereby mitigating compounding errors over sequential decision-making. Through simulations, we demonstrate that FlatVLA improves both task performance and robustness under distribution shifts without relying on modality-specific augmentations, also validated in robotics demonstrations. Beyond such gains, our behavioral analysis shows that FlatVLA exhibits uncertainty that is relatively lower than the baseline in successful cases, while showing higher uncertainty in failure cases. This contrast suggests that flatter minima may enable the policy to maintain a broader set of plausible actions in difficult states, providing greater potential for recovery from failures. Given that SAM-based approaches have demonstrated improved stability across diverse learning problems, we propose that FlatVLA provides a promising foundation for improving next-generation embodied AI systems.

AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on mathematical claims, assist in writing proofs, and polish the writing by checking grammatical errors. Additionally, we used generative AI tools to propose a title or keywords for the research paper. We reviewed all AI-assisted content and carefully double checked the revised sentences or paragraph to ensure that hallucinated content is involved during the polishing process. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This paper does not contain any content that raises ethical concerns. The work mainly focuses on theoretical analysis and experiments that validate our approach, including real-world robot manipulation experiments conducted with a self-collected dataset. These real-world experiments are limited to a standard "Pick & Place" manipulation task and do not involve human subjects, personal data, or other ethically sensitive content. Accordingly, we do not identify any ethical concerns associated with the experiments presented in this work.

REPRODUCIBILITY STATEMENT

We provide detailed experimental settings, including the selected models, a pseudo-algorithm for our proposed approach, hyperparameter settings, and reproduced prior works in the main paper and appendix. Moreover, we provide detail informationabout the hardware resources, including GPU types and their VRAM sizes, as well as other environmental setups, such as robot arm and camera type, for the real-world experiments at appendix. We also provide detailed mathematical proofs for our theoretical framework to support the validity of our approach and intuition. In last, We provide the main implementation file (e.g., `main.py`) containing the core approach described in this paper, without the remaining other modules, to aid understanding and reproducibility. We plan to release the complete codebase, including all modules, upon acceptance.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *The 3rd International Conference on Learning Representations (ICLR)*, 2015.

Dara Bahri, Hossein Mobahi, and Yi Tay. Sharpness-aware minimization improves language model generalization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics Volume 1: Long Papers)*, pp. 7360–7371, Dublin, Ireland, May 2022. Association for Computational Linguistics (ACL). doi: 10.18653/v1/2022.acl-long.508.

Robert Bain, Mikhail Tokarev, Harsh Kothari, and Rahul Damineni. Lossplot: A better way to visualize loss landscapes. *arXiv preprint arXiv:2111.15133*, 2021.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

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

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In Joseph Lim, Shuran Song, and Hae-Won Park (eds.), *Proceedings of The 9th Conference on Robot Learning (CoRL)*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 27–30 Sep 2025a. URL https://proceedings.mlr.press/v305/black25a.html.

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. In *Proceedings of Robotics: Science and Systems (RSS)*, Los Angeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.010.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chenfeng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao,

Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booher, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minho Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Muhammad Zubair Irshad, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitrano, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Mart'in-Mart'in, Rohan Baijal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Haldar, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Vitor Guizilini, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024. doi: 10.1109/ICRA57147.2024.10611477.

Felix Dangel, Stefan Harmeling, and Philipp Hennig. Modular block-diagonal curvature approximations for feedforward architectures. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108 of *Proceedings of Machine Learning Research*, pp. 799–808. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/dangel20a.html.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar" Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL): Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1019–1028. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/dinh17b.html.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *The Ninth International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 38574–38583, June 2026.

Tamar Flash and Neville Hogan. The coordination of arm movements: an experimentally confirmed mathematical model. *The Journal of Neuroscience*, 5(7):1688–1703, July 1985. ISSN 0270-6474. doi: 10.1523/JNEUROSCI.05-07-01688.1985.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *The Ninth International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=6Tm1mposlrM.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: A family of highly capable multimodal models. Technical report, Google DeepMind, 2023.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024. doi: 10.15607/RSS.2024.XX.090. URL https://www.roboticsproceedings.org/rss20/p090.html.

Jianing Guo, Zhenhong Wu, Chang Tu, Yiyao Ma, Xiangqi Kong, Zhiqian Liu, Jiaming Ji, Shuning Zhang, Yuanpei Chen, Kai Chen, Qi Dou, Yaodong Yang, Xianglong Liu, Huijie Zhao, Weifeng Lv, and Simin Li. On robustness of vision-language-action model against multi-modal perturbations. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations (ICLR)*, volume 2026, pp. 70248–70272, 2026.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Proceedings of Robotics: Science and Systems (RSS)*, Los Angeles, CA, USA, 06–09 Nov 2025a. doi: 10.15607/RSS.2025.XXI.017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning (CoRL)*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Mathias Lechner, Ramin Hasani, Radu Grosu, Daniela Rus, and Thomas A Henzinger. Adversarial training is not ready for robot learning. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4140–4147. IEEE, 2021. doi: 10.1109/ICRA48506.2021.9561036.

HyunKyu Lee and Sung Whan Yoon. Flat reward in policy parameter space implies robust reinforcement learning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations (ICLR)*, volume 2025, pp. 20408–20430, 2025.

Taehwan Lee, Kyeongkook Seo, Jaejun Yoo, and Sung Whan Yoon. Understanding flatness in generative models: Its role and benefits. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4908–4917, October 2025.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31. Curran Associates, Inc., 2018.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023. doi: 10.52202/075280-1939.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized bert pretraining approach, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *The Seventh International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Haocheng Luo, Zehang Deng, Thanh-Toan Do, Mehrtash Harandi, Dinh Phung, and Trung Le. Sharpness-aware minimization in logit space efficiently enhances direct preference optimization. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations (ICLR)*, volume 2026, pp. 96557–96581, 2026.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *The 6th International Conference on Learning Representations (ICLR)*, 2018. URL https://openreview.net/forum?id=rJzIBfZAb.

Abed AlRahman Al Makdah, Vishaal Krishnan, and Fabio Pasqualetti. Learning lipschitz feedback policies from expert demonstrations: Closed-loop guarantees, robustness, and generalization. *IEEE Open Journal of Control Systems*, 1:85–99, 2022. doi: 10.1109/OJCSYS.2022.3181584.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research (JMLR)*, 21(146):1–76, 2020. URL https://jmlr.org/papers/v21/17-678.html.

Yurii Nesterov and Boris Teodorovich Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006. doi: 10.1007/s10107-006-0706-8.

OpenAI. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774*, 2023.

Minho Park, Kinam Kim, Junha Hyung, Hyojin Jang, Hoiyeong Jin, Jooyeol Yun, Hojoon Lee, and Jaegul Choo. ACG: Action coherence guidance for flow-based vla models. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, Vienna, Austria, June 2026. IEEE.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 9 of *Proceedings of Machine Learning Research*, pp. 661–668, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/ross10a.html.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík (eds.), *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/ross11a.html.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. LLaMA 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems (NIPS)*, volume 30. Curran Associates, Inc., 2017.

Weebum Yoo and Sung Whan Yoon. A flat minima perspective on understanding augmentations and model robustness. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence*, AAAI'26. AAAI Press, 2026. doi: 10.1609/aaai.v40i33.40011.

Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Jodilyn Peralta, Brian Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

Haochen Zhang and Yonatan Bisk. Flatness preserves instruction following in vision-language-action models. *arXiv preprint arXiv:2606.23641*, 2026.

Hongyin Zhang, Shiyuan Zhang, Junxi Jin, Qixin Zeng, Yifan Qiao, Hongchao Lu, and Donglin Wang. Balancing signal and variance: adaptive offline rl post-training for vla flow models. In *Proceedings of the Fortieth AAAI Conference on Artificial Intelligence*, AAAI'26. AAAI Press, 2026. doi: 10.1609/aaai.v40i22.38944.

Yihao Zhang, Hangzhou He, Jingyu Zhu, Huanran Chen, Yifei Wang, and Zeming Wei. On the duality between sharpness-aware minimization and adversarial training. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the Forty-First International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 59024–59041. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/zhang24w.html.

A Theoretical Proofs for Smoothness and Compounding Error

A.1 Detailed Assumptions and Notations

Sequential Variables and Distributions Let $(\mathcal{Z}, d_{\mathcal{Z}})$ be the metric space of sequential variables and let $\mathcal{A} \subseteq \mathbb{R}^m$ be the action space with norm $\|\cdot\|$. Throughout this appendix, W denotes the 1-Wasserstein distance. We assume that all distributions used below are well defined and have finite first moments, ensuring that every Wasserstein distance is finite. For compactness, we write the policy input $\mathbf{x}$ as x throughout this appendix. The sequential variable is $z = (s, x)$, and the metric $d_{\mathcal{Z}}$ is assumed to dominate policy-input displacement in the local region, such that $\|x - x'\| \le d_{\mathcal{Z}}(z, z')$.

The expert and learned-policy distributions at timestep t are $\mathcal{D}_{*,t}$ and $\mathcal{D}_{\theta,t}$, and $\mathcal{P}_t(\cdot \mid z, \mathbf{a})$ denotes the conditional distribution of z_{t+1} given the z and $\mathbf{a}$. The corresponding expert and learned actions are $\mathbf{a}_t^* = \pi^*(x_t^*)$ and $\mathbf{a}_t = \pi_\theta(x_t)$, respectively. The one step expert imitation error is $\epsilon_t(\theta) = \mathbb{E}_{z_t^* \sim \mathcal{D}_{*,t}}[\|\pi_\theta(x_t^*) - \mathbf{a}_t^*\|]$, and the sequential distribution deviation is $\delta_t(\theta) = W(\mathcal{D}_{*,t}, \mathcal{D}_{\theta,t})$.

Assumption 2 (Common Initialization)**.** *The expert and learned-policy executions begin from the same task-initial distribution.*

Under Assumption 2, $\delta_1(\theta) = 0$. This initialization is used to obtain the finite-horizon bound in Eq. (7); if the initial distributions differ, the recurrence remains valid but includes an additional term determined by the initial deviation.

Local Smoothness Conditions The local smoothness condition in Assumption 1 is

$$W(\mathcal{P}_t(\cdot \mid z, \mathbf{a}), \mathcal{P}_t(\cdot \mid z', \mathbf{a}')) \le L_{z,t} d_{\mathcal{Z}}(z, z') + L_{a,t} \|\mathbf{a} - \mathbf{a}'\|. \tag{11}$$

The local policy input sensitivity is $K_t(\theta) = \sup_{x \in \mathcal{R}_t} \|J_x \pi_\theta(x)\|$, where $\mathcal{R}_t$ contains the coupled expert and learner inputs and the line segments joining each paired input. We assume π_θ is differentiable on an open neighborhood of $\mathcal{R}_t$.

A.2 Proof of Lemma 1

Proof. The expert and learned-policy next-step distributions admit the following mixture representations:

$$\mathcal{D}_{*,t+1} = \mathbb{E}_{z_t^* \sim \mathcal{D}_{*,t}} \left[\mathcal{P}_t(\cdot \mid z_t^*, \mathbf{a}_t^*)\right], \tag{12}$$

$$\mathcal{D}_{\theta,t+1} = \mathbb{E}_{z_t \sim \mathcal{D}_{\theta,t}} \left[\mathcal{P}_t(\cdot \mid z_t, \mathbf{a}_t)\right]. \tag{13}$$

For any $\varepsilon > 0$, let $\gamma_t \in \Pi(\mathcal{D}_{*,t}, \mathcal{D}_{\theta,t})$ be an ε-optimal coupling satisfying $\mathbb{E}_{\gamma_t}[d_{\mathcal{Z}}(z_t^*, z_t)] \le \delta_t(\theta) + \varepsilon$. The corresponding coupling of the two mixture distributions yields

$$\delta_{t+1}(\theta) \le \mathbb{E}_{(z_t^*, z_t) \sim \gamma_t} \Big[W(\mathcal{P}_t(\cdot \mid z_t^*, \mathbf{a}_t^*), \mathcal{P}_t(\cdot \mid z_t, \mathbf{a}_t)) \Big]. \tag{14}$$

Applying Eq. (11),

$$\delta_{t+1}(\theta) \le L_{z,t} \mathbb{E}_{\gamma_t}[d_{\mathcal{Z}}(z_t^*, z_t)] + L_{a,t} \mathbb{E}_{\gamma_t}\left[\|\mathbf{a}_t - \mathbf{a}_t^*\|\right]. \tag{15}$$

Applying the triangle inequality after adding and subtracting $\pi_\theta(x_t^*)$ gives

$$\|\mathbf{a}_t - \mathbf{a}_t^*\| \le \|\pi_\theta(x_t) - \pi_\theta(x_t^*)\| + \|\pi_\theta(x_t^*) - \mathbf{a}_t^*\| \tag{16}$$

$$\le K_t(\theta) d_{\mathcal{Z}}(z_t, z_t^*) + \|\pi_\theta(x_t^*) - \mathbf{a}_t^*\|. \tag{17}$$

Because the first marginal of γ_t is $\mathcal{D}_{*,t}$,

$$\mathbb{E}_{\gamma_t}\left[\|\pi_\theta(x_t^*) - \mathbf{a}_t^*\|\right] = \epsilon_t(\theta). \tag{18}$$

Taking $\varepsilon \downarrow 0$,

$$\delta_{t+1}(\theta) \le \left(L_{z,t} + L_{a,t} K_t(\theta)\right) \delta_t(\theta) + L_{a,t} \epsilon_t(\theta). \tag{19}$$

Substituting $\alpha_t(\theta) = L_{z,t} + L_{a,t} K_t(\theta)$ proves Lemma 1. □

A.3 Derivation and Proof of Proposition 1

Local Smoothness and Positive Curvature Let $\mathcal{L}_{\mathrm{IL}}(\theta)$ be the imitation-learning objective and $H_{\mathrm{IL}}(\theta) = \nabla^2 \mathcal{L}_{\mathrm{IL}}(\theta)$ its Hessian. Since the finite-radius measure $S_\rho(\theta)$ depends on the parameterization, all smoothness comparisons in this analysis use the same parameterization and perturbation radius ρ (Dinh et al., 2017).

Assumption 3 (Local Loss Smoothness)**.** *The Hessian is locally Lipschitz in the ρ-ball:*

$$\|H_{\mathrm{IL}}(\theta + u) - H_{\mathrm{IL}}(\theta)\| \le L_H \|u\|_2, \qquad \|u\|_2 \le \rho. \tag{20}$$

This standard condition in second-order optimization controls the remainder of the local quadratic approximation (Nesterov & Polyak, 2006). Under Assumption 3, Taylor's theorem yields

$$\left| \mathcal{L}_{\mathrm{IL}}(\theta + u) - \mathcal{L}_{\mathrm{IL}}(\theta) - \nabla \mathcal{L}_{\mathrm{IL}}(\theta)^\top u - \frac{1}{2} u^\top H_{\mathrm{IL}}(\theta) u \right| \le \frac{L_H}{6} \|u\|_2^3. \tag{21}$$

Let $\lambda = \lambda_{\max}(H_{\mathrm{IL}}(\theta))$ and let v be a corresponding unit eigenvector. Applying the lower Taylor bound in Eq. (21) with $u = \rho v$ and $u = -\rho v$ gives

$$\mathcal{L}_{\mathrm{IL}}(\theta + \rho v) - \mathcal{L}_{\mathrm{IL}}(\theta) \ge \rho \nabla \mathcal{L}_{\mathrm{IL}}(\theta)^\top v + \frac{1}{2}\rho^2 \lambda - \frac{L_H}{6}\rho^3, \tag{22}$$

$$\mathcal{L}_{\mathrm{IL}}(\theta - \rho v) - \mathcal{L}_{\mathrm{IL}}(\theta) \ge -\rho \nabla \mathcal{L}_{\mathrm{IL}}(\theta)^\top v + \frac{1}{2}\rho^2 \lambda - \frac{L_H}{6}\rho^3. \tag{23}$$

Since both perturbations are feasible in Eq. (5), it follows that

$$S_\rho(\theta) \ge \max\{\mathcal{L}_{\mathrm{IL}}(\theta + \rho v) - \mathcal{L}_{\mathrm{IL}}(\theta), \mathcal{L}_{\mathrm{IL}}(\theta - \rho v) - \mathcal{L}_{\mathrm{IL}}(\theta)\} \tag{24}$$

$$\ge |\rho \nabla \mathcal{L}_{\mathrm{IL}}(\theta)^\top v| + \frac{1}{2}\rho^2 \lambda - \frac{L_H}{6}\rho^3 \tag{25}$$

$$\ge \frac{1}{2}\rho^2 \lambda - \frac{L_H}{6}\rho^3. \tag{26}$$

Because $u = 0$ is feasible in Eq. (5), $S_\rho(\theta) \ge 0$, and therefore

$$[\lambda_{\max}(H_{\mathrm{IL}}(\theta))]_+ \le \frac{2S_\rho(\theta)}{\rho^2} + \frac{L_H \rho}{3}. \tag{27}$$

Thus, Eq. (27) controls the positive part of the largest Hessian eigenvalue rather than the Hessian spectral norm. No stationarity condition is required because the symmetric perturbations $+\rho v$ and $-\rho v$ convert the first-order contribution into an absolute value.

Curvature and Policy Sensitivity We next connect the curvature of the imitation-learning objective to the policy input sensitivity $K_t(\theta)$ through a local parameter-sensitivity condition and an input-parameter sensitivity decomposition. Let the action dimension be m, the policy-input dimension be p, and the parameter dimension be d, so that $J_\theta \pi_\theta(x) \in \mathbb{R}^{m \times d}$ and $J_x \pi_\theta(x) \in \mathbb{R}^{m \times p}$.

Assumption 4 (Curvature–Parameter Sensitivity)**.** *The parameter sensitivity of the policy over $\mathcal{R}_t$ is controlled by the curvature of the imitation-learning objective with a residual term:*

$$\sup_{x \in \mathcal{R}_t} \|J_\theta \pi_\theta(x)\|^2 \le c_t \, [\lambda_{\max}(H_{\mathrm{IL}}(\theta))]_+ + \xi_t, \qquad c_t, \xi_t \ge 0. \tag{28}$$

The condition in Eq. (28) is motivated by Generalized Gauss–Newton (GGN) analyses, where model-output Jacobians naturally appear in positive-semidefinite curvature matrices of the form $J_\theta^\top H_\ell J_\theta$, with H_ℓ denoting the curvature of the loss with respect to the model output (Martens, 2020; Dangel et al., 2020). This structure motivates connecting positive loss curvature to local parameter sensitivity, while ξ_t allows for sensitivity that is not fully captured by the curvature term.

Input–Parameter Sensitivity Decomposition The sequential bound depends on sensitivity to input changes rather than parameter perturbations. Then, we decompose the input Jacobian as

$$J_x \pi_\theta(x) = J_\theta \pi_\theta(x) M_t(x) + R_t(x), \tag{29}$$

where $M_t(x) \in \mathbb{R}^{d \times p}$ represents a local mapping from input perturbations to parameter-space directions, and $R_t(x) := J_x \pi_\theta(x) - J_\theta \pi_\theta(x) M_t(x)$ captures the remaining input-sensitive component. The corresponding local bounds are $M_t := \sup_{x \in \mathcal{R}_t} \|M_t(x)\|$ and $\eta_t := \sup_{x \in \mathcal{R}_t} \|R_t(x)\|$.

Assumption 5 (Bounded Local Input–Parameter Sensitivity)**.** *The quantities $\bar{M}_t$ and η_t are finite over $\mathcal{R}_t$. For policy comparisons, they are assumed to admit common finite upper bounds over the corresponding local regions.*

The decomposition in Eq. (29) bridges parameter sensitivity and the input sensitivity required by the sequential analysis. It separates the input Jacobian into a component represented through parameter-sensitive directions and a residual component. Together with the curvature–parameter sensitivity condition, this allows us to bound $K_t(\theta)$ and, in turn, the sequential error amplification factor $\alpha_t(\theta)$.

Combining Eqs. (27) and (28) yields

$$\sup_{x\in\mathcal{R}_t} \|J_\theta \pi_\theta(x)\| \le \sqrt{c_t\left(\frac{2S_\rho(\theta)}{\rho^2} + \frac{L_H\rho}{3}\right) + \xi_t}. \tag{30}$$

Applying Eq. (29) and the standard norm inequality $\|AB\| \le \|A\|\|B\|$ yields

$$K_t(\theta) = \sup_{x\in\mathcal{R}_t} \|J_x \pi_\theta(x)\| \tag{31}$$

$$\le \bar{M}_t \sup_{x\in\mathcal{R}_t} \|J_\theta \pi_\theta(x)\| + \eta_t \tag{32}$$

$$\le \bar{M}_t \sqrt{c_t\left(\frac{2S_\rho(\theta)}{\rho^2} + \frac{L_H\rho}{3}\right) + \xi_t} + \eta_t. \tag{33}$$

Applying $\sqrt{a+b} \le \sqrt{a} + \sqrt{b}$ for $a, b \ge 0$, set $A_t = \bar{M}_t\sqrt{c_t}$ and $B_t = \bar{M}_t\sqrt{\xi_t} + \eta_t$. The previous bound then simplifies to

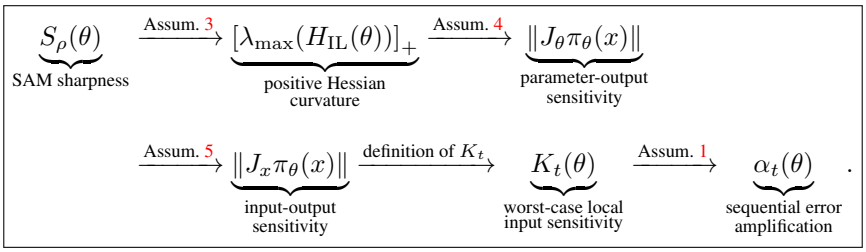
$$K_t(\theta) \le A_t \sqrt{\frac{2S_\rho(\theta)}{\rho^2} + \frac{L_H\rho}{3}} + B_t. \tag{34}$$

Substituting the bound on $K_t(\theta)$ into $\alpha_t(\theta) = L_{z,t} + L_{a,t}K_t(\theta)$ gives

$$\alpha_t(\theta) \le L_{z,t} + L_{a,t}\left[A_t\sqrt{\frac{2S_\rho(\theta)}{\rho^2} + \frac{L_H\rho}{3}} + B_t\right] = \bar{\alpha}_t(\theta). \tag{35}$$

For fixed nonnegative constants $L_{z,t}, L_{a,t}, A_t, B_t, L_H, \rho$, the right-hand side is non-decreasing in $S_\rho(\theta)$. For comparing two policies, these constants must be common dominating constants over the relevant local neighborhoods.

The overall proof proceeds through a sequence of local bounds that progressively connect SAM sharpness to sequential error amplification. Assumption 3 relates SAM sharpness to positive Hessian curvature through the two-sided Taylor bound, while Assumption 4 connects this curvature to parameter-output sensitivity. The decomposition in Eq. (29), together with Assumption 5, then connects parameter sensitivity to input-output sensitivity, whose local supremum defines $K_t(\theta)$. Finally, Assumption 1 relates $K_t(\theta)$ to the sequential error amplification factor $\alpha_t(\theta) = L_{z,t} + L_{a,t}K_t(\theta)$. For clarity, the overall flow of Proposition 1 is summarized below, showing how the local bounds progressively connect SAM sharpness to sequential error amplification:



This establishes the expected bound and completes the proof of Proposition 1.

A.4 Derivation of the Compounding-Error Bound

Recurrence Unrolling The uniform quantities defined in the main paper satisfy $\bar{\alpha}_t(\theta) \le \bar{\alpha}(\theta)$, $L_{a,t} \le \beta$, and $\epsilon_t(\theta) \le \epsilon$ for every $t < T$. These inequalities yield the upper-bound recurrence

$$\bar{\delta}_1 = 0, \qquad \bar{\delta}_{t+1} = \bar{\alpha}(\theta)\bar{\delta}_t + \beta\epsilon. \tag{36}$$

Induction on t yields

$$\bar{\delta}_t = \beta\epsilon \sum_{j=0}^{t-2} \bar{\alpha}(\theta)^j, \qquad t \geq 2. \tag{37}$$

Summing over t and exchanging the order of the finite sums gives

$$C_T(\theta) \leq \bar{C}_T(\theta;\epsilon) \tag{38}$$

$$= \sum_{t=1}^{T} \bar{\delta}_t = \beta\epsilon \sum_{t=2}^{T} \sum_{j=0}^{t-2} \bar{\alpha}(\theta)^j \tag{39}$$

$$= \beta\epsilon \sum_{j=0}^{T-2} (T-1-j)\bar{\alpha}(\theta)^j. \tag{40}$$

This is the finite-horizon upper bound in Eq. (7). For $\bar{\alpha}(\theta) \neq 1$, evaluating the weighted geometric sum yields

$$\bar{C}_T(\theta;\epsilon) = \beta\epsilon \frac{(T-1) - T\bar{\alpha}(\theta) + \bar{\alpha}(\theta)^T}{(1-\bar{\alpha}(\theta))^2}. \tag{41}$$

For $\bar{\alpha}(\theta) = 1$, direct summation gives

$$\bar{C}_T(\theta;\epsilon) = \beta\epsilon \frac{T(T-1)}{2}. \tag{42}$$

Stable, Marginal, and Expansive Regimes Let $\bar{\alpha} = \bar{\alpha}(\theta)$. The three error-accumulation regimes follow directly from the finite-sum and closed-form expressions. If $\bar{\alpha} < 1$, then

$$\bar{C}_T(\theta;\epsilon) \leq \frac{\beta\epsilon(T-1)}{1-\bar{\alpha}} = O(T\epsilon). \tag{43}$$

If $\bar{\alpha} = 1$, Eq. (42) gives $O(T^2\epsilon)$. If $\bar{\alpha} > 1$, Eq. (41) gives the geometric upper bound

$$\bar{C}_T(\theta;\epsilon) \leq \frac{\beta\epsilon\, \bar{\alpha}^T}{(\bar{\alpha}-1)^2}. \tag{44}$$

Monotonicity of the Upper Bound For fixed $T \geq 2$ and $\beta, \epsilon \geq 0$, the finite-sum form is non-decreasing in $\bar{\alpha} \geq 0$ because

$$\frac{\partial \bar{C}_T}{\partial \bar{\alpha}} = \beta\epsilon \sum_{j=1}^{T-2} j(T-1-j)\bar{\alpha}^{j-1} \geq 0. \tag{45}$$

The upper bound is also non-decreasing in ϵ. Consequently, $\bar{C}_T$ is non-decreasing in both $\bar{\alpha}$ and ϵ.

A.5 Proof of Theorem 1

Proof. Consider two policies π_θ and $\pi_{\theta'}$ under the same perturbation radius ρ and parameterization. Let $L_{z,t}, L_{a,t}, A_t, B_t, L_H$, and β be common constants that upper bound the corresponding quantities for policies, and let ϵ be a common upper bound on their one-step imitation errors. Suppose

$$S_\rho(\theta') \leq S_\rho(\theta). \tag{46}$$

Applying Proposition 1 under the same local constants yields as follows:

$$\bar{\alpha}_t(\theta') \leq \bar{\alpha}_t(\theta) \quad \Rightarrow \quad \bar{\alpha}(\theta') \leq \bar{\alpha}(\theta). \tag{47}$$

Since the two policies share the same error bound ϵ and $\bar{C}_T$ is non-decreasing in $\bar{\alpha}$, we obtain

$$\bar{C}_T(\theta';\epsilon) \leq \bar{C}_T(\theta;\epsilon). \tag{48}$$

This proves Theorem 1. □

Remark 1.2 (Policy-specific imitation-error bounds)**.** *If the two policies have different one-step imitation-error bounds, denoted by $\epsilon_{\theta'}$ and ϵ_θ, the comparison additionally requires $\epsilon_{\theta'} \leq \epsilon_\theta$. Since $\bar{C}_T$ is non-decreasing in both $\bar{\alpha}$ and ϵ, combining this condition with $\bar{\alpha}(\theta') \leq \bar{\alpha}(\theta)$ yields $\bar{C}_T(\theta';\epsilon_{\theta'}) \leq \bar{C}_T(\theta;\epsilon_\theta)$. Here, θ' can be interpreted as the smoother policy obtained by our approach, while θ denotes its corresponding baseline. Therefore, when the smoother policy does not incur a larger one-step imitation-error bound, the improvement in local smoothness directly translates into a no-worse compounding-error upper bound.*

B Implementation and Experimental Details

B.1 Dataset Description

LIBERO Benchmark The LIBERO benchmark (Liu et al., 2023) is widely used to evaluate VLA policies across diverse robotic manipulation tasks. It contains four suites that vary spatial relations, object identities, language-conditioned goals, and long-horizon task structure. Together, these suites evaluate task performance across controlled forms of visual, semantic, and sequential variation:

- LIBERO-Spatial: A suite of 10 tasks that share the same object set and manipulation goal but differ in the *spatial relationship* of the target bowl (*e.g.*, "between" or "next to"). This suite emphasizes spatial reasoning and sensitivity to layout changes; instructions typically follow the template `"Pick up the black bowl [spatial location] and place it on the plate."`
- LIBERO-Object: A suite of 10 tasks with a largely fixed layout, where each task requires manipulating a *different target object*. This suite evaluates object-centric robustness to changes in object identity and appearance; instructions follow `"Pick up the [object] and put it in the basket."`
- LIBERO-Goal: A suite of 10 tasks that share the same objects and spatial arrangement but differ in the *task goal* (*i.e.*, the desired outcome and behavior). This suite evaluates instruction following by requiring the policy to map different linguistic commands to distinct action sequences. Instructions generally follow `"[goal] the [object] on the [object]"`, where `goal` includes actions such as `Put`, `Push`, and `Open`.
- LIBERO-10 (Long Horizon): A suite of 10 long-horizon tasks sampled from LIBERO-100, with each task combining multiple manipulation skills. Compared with the suites above, LIBERO-10 contains more diverse objects, longer temporal dependencies, and more complex task sequences. A sample instruction is `"Put the [object] in the basket and [goal] the [object]."`

LIBERO-Plus Benchmark (Robustness) The LIBERO-Plus benchmark (Fei et al., 2026) extends LIBERO to evaluate VLA robustness under visual, linguistic, and physical variations. It contains 10,030 evaluation instances derived from the 40 original tasks across seven perturbation conditions. Each condition modifies the task environment or policy input while preserving the original manipulation goal. The conditions cover the following forms of variation:

- **Object Layout**: A condition that introduces unseen distractor objects or changes the *initial position and orientation* of the target object. This condition evaluates object selection and spatial adaptation while preserving the relations specified in the instruction (*e.g.*, "next to" the cookie box).
- **Camera Viewpoint**: A condition that varies the camera's *distance, position, and orientation* relative to the workspace. Camera distance ranges from 1.01 to 2.00 times its original value, while position and orientation changes are sampled within $15°$–$75°$ cones and $2°$–$10°$, respectively. These variations evaluate visual understanding of the same task from different viewpoints.
- **Robot Initial State**: A condition that changes the robot arm's *initial joint configuration* while retaining the same manipulation goal. This condition evaluates whether the policy can adapt its actions to different starting poses.
- **Language Instruction**: A condition that expresses the same task through *different linguistic descriptions*, including irrelevant context, commonsense object descriptions, and changes in reasoning complexity. This condition evaluates instruction following when the wording changes but the intended action remains the same.
- **Lighting**: A condition that varies *diffuse color, light direction, specular highlights, and shadows* within the scene. These changes evaluate sensitivity to illumination while retaining the same objects and spatial arrangement.
- **Background Texture**: A condition that changes the *scene theme and surface appearance*, including wall, tabletop, and floor textures. This condition evaluates robustness to background appearance beyond the original task environment.
- **Sensor Noise**: A condition that degrades visual observations through *motion blur, Gaussian blur, zoom blur, fog, and glass blur*. Stronger perturbations reduce image detail or introduce local distortions, evaluating robustness to changes in observation quality.

Table 4: Number of evaluations in LIBERO-PLUS across perturbation conditions and suites.

Condition	Spatial	Object	Goal	10	Total
Object layout	312	425	403	385	1,525
Camera viewpoint	376	396	408	419	1,599
Robot initial state	350	398	409	393	1,550
Language instruction	354	390	410	383	1,537
Lighting	292	297	279	274	1,142
Background texture	258	248	281	289	1,076
Sensor noise	351	422	379	449	1,601
Total	2,293	2,576	2,569	2,592	10,030

The benchmark further divides the evaluation instances into five difficulty levels based on the success of four reference models: OpenVLA-OFT, π_0, π_0-FAST, and UniVLA. Specifically, L1–L5 correspond to instances solved by four, three, two, one, and none of these models, respectively, allowing robustness to be examined across increasing empirical difficulty. Table 4 summarizes the number of evaluation instances for each condition across the four LIBERO suites, including all sub-dimensions and difficulty levels.

B.2 BASELINE ARCHITECTURE AND TRAINING PROCEDURE

OpenVLA OpenVLA (Kim et al., 2025b) is a 7B-parameter open-source VLA model for robotic manipulation. It combines a fused DINOv2 and SigLIP vision encoder with the LLaMA-2 (Touvron et al., 2023) language-model backbone. The projected visual features are concatenated with tokenized language instructions and processed as a unified multimodal sequence. Continuous actions are normalized and discretized into 256 bins per dimension, then predicted autoregressively as action tokens. Each prediction represents seven control dimensions: 3-DoF translation, 3-DoF rotation, and one gripper. The model is fine-tuned using cross-entropy loss over discretized tokens.

OpenVLA-OFT OpenVLA-OFT (Kim et al., 2025a) retains the pretrained OpenVLA backbone but changes how robot actions are produced during fine-tuning. It replaces sequential discrete-token decoding with parallel prediction of continuous action chunks, allowing all actions in a chunk to be generated in one forward pass. The final decoder features are mapped through an MLP action head and trained with an L1 regression objective. For LIBERO, the model predicts eight-step action chunks, providing faster inference and multi-step control than autoregressive OpenVLA.

π_0 π_0 (Black et al., 2025b) is a flow-based VLA model that combines a PaliGemma vision-language backbone (Beyer et al., 2024) with a smaller action expert for robot states and actions. The VLM processes images and language instructions, while robotics-specific inputs and outputs are handled by the action expert within the same transformer. Instead of discretizing actions, π_0 uses conditional flow matching (Lipman et al., 2023) to model distributions over continuous action chunks. The model is pretrained on a large mixture of cross-embodiment robot data and then adapted to downstream tasks through post-training.

$\pi_{0.5}$ $\pi_{0.5}$ (Black et al., 2025a) extends π_0 through co-training on heterogeneous robot data, high-level semantic supervision, verbal instructions, and web data. It retains the PaliGemma backbone (Beyer et al., 2024) and flow matching (Lipman et al., 2023) action expert while supporting both tokenized text and continuous action chunk outputs. During pretraining, robot actions are represented as discrete FAST tokens; post-training adds the action expert for efficient continuous action generation. At inference, the model first predict a high-level subtask and then generate the corresponding low-level action chunk, combining semantic planning and control within a single policy.

Pseudo Code Algorithm 1 summarizes the training procedure of FlatVLA for a VLA policy and its task loss. For each mini-batch, FlatVLA retains the task loss at every action-horizon position and modifies only its aggregation using the normalized weights in Equation (9). The resulting objective $\mathcal{L}_{\text{Flat}}$ is used in both SAM passes. The first gradient defines the parameter perturbation, while the second is evaluated at the perturbed parameters to update the original policy. The procedure applies

Algorithm 1 Training Procedure of FlatVLA

Require: Training dataset $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, \mathbf{a}_i^*)\}_{i=1}^N$ where $\mathbf{x}_i = (\mathbf{o}_i, \mathbf{t}_i)$
Require: VLA policy π_θ, per-horizon task loss ℓ (Cross Entropy, Flow Matching, etc), action horizon H
Require: Base optimizer Optim, horizon-weighting coefficient γ, and SAM perturbation radius ρ

while not converged **do**
  Sample a mini-batch $\mathcal{B} = \{(\mathbf{x}_i, \mathbf{a}_i^*)\}_{i=1}^B \subset \mathcal{D}_{\text{train}}$
  $\{\ell_{i,h}(\theta)\}_{i,h} \leftarrow \text{TaskLoss}(\pi_\theta, \mathcal{B})$ ▷ Compute the VLA task loss
  $\mathcal{L}_{\text{VLA}}(\theta) \leftarrow \frac{1}{BH} \sum_{i=1}^B \sum_{h=1}^H \ell_{i,h}(\theta)$ ▷ Uniform loss over the action horizon
  $g \leftarrow \nabla_\theta \mathcal{L}_{\text{VLA}}(\theta)$ ▷ Compute the baseline gradient
  $\theta \leftarrow \text{Optim}(\theta, g)$ ▷ Standard VLA update (replaced by FlatVLA)

  ▷ **Additional Steps for FlatVLA**

  $\widetilde{w}_h(\gamma) \leftarrow \sum_{j=0}^{H-h} \gamma^j,\ h = 1, \ldots, H$ ▷ Compute horizon-dependent weights
  $w_h(\gamma) \leftarrow \frac{H \widetilde{w}_h(\gamma)}{\sum_{r=1}^H \widetilde{w}_r(\gamma)}$ ▷ Normalize the weights to unit mean
  $\mathcal{L}_{\text{Flat}}(\theta) \leftarrow \frac{1}{BH} \sum_{i=1}^B \sum_{h=1}^H w_h(\gamma) \ell_{i,h}(\theta)$ ▷ Aggregate the weighted task loss
  $g_1 \leftarrow \nabla_\theta \mathcal{L}_{\text{Flat}}(\theta)$ ▷ Compute the first-pass gradient
  $\epsilon_{\text{SAM}} \leftarrow \rho \frac{g_1}{\|g_1\|_2}$ ▷ Compute the SAM perturbation
  $\theta' \leftarrow \theta + \epsilon_{\text{SAM}}$ ▷ Temporarily perturb the parameters
  $\{\ell_{i,h}(\theta')\}_{i,h} \leftarrow \text{TaskLoss}(\pi_{\theta'}, \mathcal{B})$ ▷ Recompute the VLA task loss
  $\mathcal{L}_{\text{Flat}}(\theta') \leftarrow \frac{1}{BH} \sum_{i=1}^B \sum_{h=1}^H w_h(\gamma) \ell_{i,h}(\theta')$ ▷ Use the same weighted objective
  $g_2 \leftarrow \nabla_\theta \mathcal{L}_{\text{Flat}}(\theta')$ ▷ Compute the second-pass gradient
  $\theta \leftarrow \text{Optim}(\theta, g_2)$ ▷ Update the original policy parameters
end while

Legend: Red Highlight: Original / Baseline Blue Highlight: FlatVLA

to different VLA training formulations, including token prediction, continuous-action regression, diffusion, and flow matching, without changing their loss definitions. For clarity, the baseline update and the additional steps introduced by FlatVLA are highlighted separately in Algorithm 1.

B.3 Hyperparameter Settings

All four VLA models are fine-tuned on the LIBERO training data across all suites under a common dataset protocol. We follow the standard model-specific hyperparameter settings and modify only the hardware allocation to accommodate resource constraints. Detailed configurations are provided in Table 5; γ denotes the coefficient used for sequential horizon weighting in Equation (9). Lastly, we use AdamW (Loshchilov & Hutter, 2019), a widely used optimizer for fine-tuning models.

As mentioned previously, for a chunk of length H, γ controls the relative weight assigned to position h through Equation (9). OpenVLA has $H = 1$, so $\gamma = 0$ yields a uniform single-position weight; $\gamma = 0.5$ for $\boldsymbol{\pi}_0$ ($H = 50$) mildly favors earlier positions, while $\gamma = 0.7$ for OpenVLA-OFT ($H = 8$) and $\boldsymbol{\pi}_{0.5}$ ($H = 10$) give them stronger emphasis. For example, with $\boldsymbol{\pi}_{0.5}$ and $H = 10$, $\gamma = 0.7$ gives $(w_1, w_2, w_3, \ldots, w_8, w_9, w_{10}) \approx (1.257, 1.241, 1.219, \ldots, 0.850, 0.660, 0.388)$, so the normalized weights place progressively less emphasis on later positions while retaining unit mean across the chunk. For $\boldsymbol{\pi}_0$ with $H = 50$ and $\gamma = 0.5$, the corresponding profile is $(w_1, w_2, w_3, \ldots, w_{48}, w_{49}, w_{50}) \approx (1.020, 1.020, 1.020, \ldots, 0.893, 0.765, 0.510)$.

B.4 Detailed Evaluation Setup and Robustness Settings

Evaluation Setup The real-world manipulation task requires the policy to identify the cup specified by the language instruction (`orange` or `blue`), grasp it, and place it on the plate of the corre-

Table 5: Fine-tuning configurations for the VLA models on LIBERO. Here, **C.A** stands for **C**osine **A**nnealing (Decay) scheduler.

Configuration	**OpenVLA**	**OpenVLA-OFT**	π_0	$\pi_{0.5}$
GPUs	4× A6000 (48 GB)	4× A100 (80 GB)	2× A100 (80 GB)	2× A100 (80 GB)
Batch size (Total)	64	32	32	32
Learning rate	5×10^{-4}	5×10^{-4}	2.5×10^{-5}	5×10^{-5}
Scheduler	None	Step Decay (100k)	Warmup + **C.A**	Warmup
radius ρ	0.05	0.05	0.01	0.01
γ	0	0.7	0.5	0.7
Action Chunk H	1	8	50	10
Iterations	120k	150k	30k	30k

Table 6: Fine-tuning hyperparameters for VLA models on the real-world manipulation task.

Configuration	**OpenVLA-OFT**		$\pi_{0.5}$	
	Baseline	FlatVLA	Baseline	FlatVLA
GPUs	1× A100 (80 GB)	1× A100 (80 GB)	1× A100 (80 GB)	1× A100 (80 GB)
Batch size (Total)	16	16	8	8
Learning rate	5×10^{-4}	5×10^{-4}	5×10^{-5}	5×10^{-5}
Scheduler	Step Decay (4k)	Step Decay (4k)	Warmup + **C.A**	Warmup + **C.A**
radius ρ	None	0.05	None	0.05
γ	None	0.7	None	0.7
Action Chunk H	5	5	5	5
Iterations	4.5k	4.5k	20k	20k

sponding color. We use the canonical instruction template "`Pick up the [color] cup and place it on the [color] plate`", where `[color]` denotes either `orange` or `blue`. The training set contains 100 expert demonstrations, evenly balanced across the two target colors with 50 demonstrations per color. For each model variant and evaluation condition, we perform 10 trials in total, with five trials assigned to each target color to ensure a balanced evaluation. Each episode is executed for up to 100 timesteps, corresponding to a maximum duration of 20 s. We report the overall task success rate aggregated across both target colors. Detailed fine-tuning configurations and hyperparameters for each model are provided in Table 6.

Robustness Settings Beyond the clean evaluation setting, we assess model robustness under three types of perturbations: **Gaussian visual noise**, **background changes**, and **lexical transformations** of the language instruction. For **Gaussian visual noise**, each input image is first resized to 224×224 pixels, after which zero-mean Gaussian noise with a standard deviation of $\sigma = 25$ is applied. For this perturbation, we use a fixed random seed of 42 to ensure consistent noise across model evaluations. For **background changes**, we place grid-patterned panels around the target cups to alter their local visual context while leaving the task-relevant objects unchanged. The structured patterns introduce additional visual clutter and potentially ambiguous spatial cues around the objects, making object-background separation and spatial perception more challenging. For **lexical transformations**, we replace the original instruction with *"Retrieve the [color] mug and position it onto the [color] dish"*, where `[color]` is either `orange` or `blue`. This transformation changes the lexical form of the instruction while preserving the underlying task semantics and manipulation objective.

C Additional Experimental Analysis

C.1 Trajectory and Action Smoothness Analysis in Real-World Manipulation

In this section, we further analyze the real-world evaluation rollouts to better understand the behavioral differences between the baseline and FlatVLA policies. Specifically, we examine their task execution trajectories together with Action Total Variation (ATV) (Flash & Hogan, 1985), which provides a measure of action smoothness. This analysis allows us to characterize how each policy progresses toward task completion and to identify where unsuccessful rollouts begin to deviate in terms of both task execution and action dynamics.

Table 7: Action total variation in success cases. Lower values indicate smoother actions.

Models & Approaches		Action Total Variation (↓)	
		Translation (X, Y, Z)	Rotation (Roll, Pitch, Yaw)
OpenVLA-OFT	Baseline	0.92 ± 0.05	0.72 ± 0.01
	FlatVLA	$\mathbf{0.66 \pm 0.07}$	$\mathbf{0.69 \pm 0.02}$
$\pi_{0.5}$	Baseline	2.85 ± 0.07	1.25 ± 0.02
	FlatVLA	$\mathbf{2.64 \pm 0.07}$	$\mathbf{1.21 \pm 0.02}$

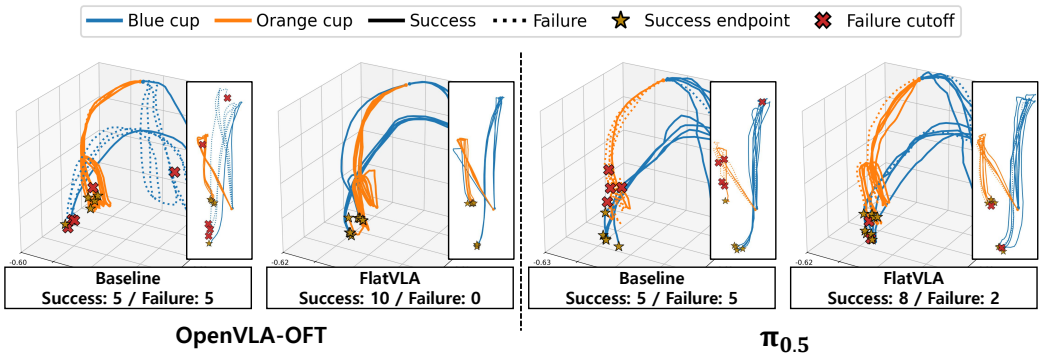


Figure 4: Representative real-world manipulation trajectories of the baseline and FlatVLA policies, illustrating their behavior throughout task execution.

Action Smoothness Analysis FlatVLA is designed to mitigate compounding error by promoting local smoothness in the parameter space at each decision step. To examine how this property affects the resulting policy behavior, we compare the action sequences produced by FlatVLA and the corresponding baseline policies in the real-world setting. Following prior work (Park et al., 2026), we measure action smoothness using Action Total Variation (ATV) (Flash & Hogan, 1985), defined as

$$\mathrm{ATV} = \frac{1}{D(T-1)} \sum_{t=1}^{T-1} \sum_{d=1}^{D} \left| \mathbf{a}_{t+1}^{d} - \mathbf{a}_{t}^{d} \right|. \tag{49}$$

Here, T denotes the episode length and $\mathbf{a}_t \in \mathbb{R}^D$ represents the policy action at timestep t. The action consists of 6-DoF end-effector commands, including 3D translation (X, Y, Z) and 3D rotation (Roll, Pitch, Yaw), while the gripper action is excluded. We compute ATV separately for the each components, using $D = 3$ for each. For a consistent comparison across rollouts, we compute ATV over the approach phase, corresponding to the first 64 timesteps of each episode (Park et al., 2026).

As shown in Table 7, FlatVLA yields lower ATV than the baseline across both translation and rotation for both VLA models. For OpenVLA-OFT, the translational ATV decreases from $0.92 \rightarrow 0.66$, corresponding to a reduction of approximately 28.3%, while the rotational ATV decreases from $0.72 \rightarrow 0.69$. A similar trend is observed for $\pi_{0.5}$, where the translational and rotational ATV decrease from $2.85 \rightarrow 2.64$ and from $1.25 \rightarrow 1.21$, respectively. Overall, these results indicate that FlatVLA produces smoother consecutive action updates during the approach phase, consistent with its objective of promoting locally smooth policy behavior.

Trajectory Analysis As shown in Figure 4, the baseline and FlatVLA exhibit qualitatively different failure behaviors. For both OpenVLA-OFT and $\pi_{0.5}$, baseline failures occur at multiple intermediate stages, whereas FlatVLA failures tend to occur near the task endpoint, where a minor corrective action could have resulted in success. For $\pi_{0.5}$, this pattern is consistent with the decision-level uncertainty analysis in Table 3. Specifically, the larger success–failure uncertainty gap observed with FlatVLA suggests that, in difficult states, the policy may preserve multiple plausible action choices rather than committing early to a suboptimal decision. This trajectory-level observation is also consistent with the higher success rates achieved by FlatVLA under both clean and perturbed hardware settings in Figure 2. Overall, these results suggest that FlatVLA not only improves aggregate task success but also produces failure trajectories that remain closer to successful task completion.

Table 8: Success rates(SR; %) under Gaussian action noise across LIBERO suites. The clean setting is denoted by $\sigma = 0$. Bold values indicate the higher SR for each model, suite, and noise level.

Models	σ	Spatial		Object		Goal		LIBERO-10	
		Baseline	FlatVLA	Baseline	FlatVLA	Baseline	FlatVLA	Baseline	FlatVLA
OpenVLA	0.0	85.80	**87.20**	87.60	**91.20**	78.00	**80.60**	51.80	**54.80**
	0.1	76.00	**76.80**	76.60	**80.60**	76.80	**78.80**	39.70	**45.30**
	0.2	44.10	**52.27**	44.40	**47.80**	46.00	**47.80**	28.20	**34.00**
	0.3	12.60	**13.80**	13.78	**15.60**	11.90	**16.30**	3.80	**5.77**
OpenVLA-OFT	0.0	97.90	**98.30**	98.30	**99.00**	96.50	**97.50**	94.30	**94.90**
	0.1	89.10	**91.10**	90.80	**91.20**	85.70	**87.50**	84.50	**85.20**
	0.2	63.80	**70.10**	59.10	**59.40**	58.40	**60.30**	41.50	**44.90**
	0.3	35.60	**39.50**	25.70	**27.70**	32.90	**34.80**	8.70	**10.80**
π_0	0.0	96.50	**97.20**	96.73	**97.60**	95.27	**96.10**	84.60	**86.10**
	0.1	90.67	**91.53**	90.93	**92.00**	83.73	**85.93**	63.60	**71.47**
	0.2	70.70	**72.87**	60.20	**62.53**	54.40	**59.27**	30.29	**34.47**
	0.3	42.00	**43.20**	26.80	**34.00**	35.20	**37.20**	11.00	**14.00**
$\pi_{0.5}$	0.0	97.68	**98.53**	97.50	**99.23**	95.89	**97.50**	92.24	**94.30**
	0.1	95.13	**95.27**	94.93	**95.67**	91.47	**93.47**	85.67	**87.73**
	0.2	83.20	**86.93**	76.67	**81.20**	71.47	**73.27**	56.73	**59.67**
	0.3	64.07	**69.00**	45.93	**47.13**	49.87	**53.00**	20.47	**23.50**

C.2 Success Rates under Action Noise Perturbations

In this section, we provide the absolute success rates in Table 8, from which the performance retention shown in Figure 3 is calculated. These results include the clean setting ($\sigma = 0$) and three Gaussian action noise levels, $\sigma \in \{0.1, 0.2, 0.3\}$, across the four VLA models and LIBERO suites. For each model, method, and suite, the success rate at each noise level is normalized by its corresponding clean success rate to obtain the retention reported in the main paper.

As expected, success rates gradually decrease as the magnitude of action noise increases. Nevertheless, FlatVLA consistently outperforms the corresponding baseline across all models, suites, and tested noise levels, indicating improved robustness to perturbations in executed actions. The largest improvement is observed for OpenVLA on LIBERO-Spatial at $\sigma = 0.2$, where FlatVLA retains 59.94% of its clean performance compared with 51.40% for the baseline, corresponding to an 8.54 percentage-point improvement in retention. The benefit also persists under stronger action noise. For example, on LIBERO-Object at $\sigma = 0.3$, π_0 improves from 26.80% with the baseline to 34.00% with FlatVLA. While the magnitude of improvement varies across models and suites, FlatVLA achieves higher success rates at every tested noise level. Together with the normalized retention results reported in the main paper, these results suggest that FlatVLA not only improves performance under noisy actions but also better preserves the policy's clean-performance level as action noise becomes stronger.

C.3 Robustness Analysis in Parameter-Space Perspective

Parameter-Space Analysis Existing robustness evaluations for vision-language-action (VLA) tasks have primarily focused on perturbations in the input space, such as visual corruptions and linguistic variations, and in the output space, through action disturbances. However, these assessments do not fully capture the inherent robustness from a learning perspective. In deep learning, the primary objective is to learn an optimal hypothesis $h_\theta \in \mathcal{H}$ parameterized by θ. From this viewpoint, we claim that robustness should be considered not only in the input or action space, but also in the parameter space, since the model behavior is fundamentally determined by the parameters.

This aspect is especially relevant to our approach FlatVLA, since it is trained with a SAM-based optimization scheme that explicitly encourages convergence to flatter solutions in parameter space. To better understand the robustness improvements observed in the main paper, it is necessary to analyze whether FlatVLA indeed exhibits enhanced robustness with respect to parameter space (Yoo & Yoon, 2026). In other words, if the improved robustness of FlatVLA is truly related to flatness-aware optimization, analyses at parameter space is mandatory. Therefore, we provide additional analyses from the perspective of parameter-space robustness, **loss landscape visualization** (Li et al., 2018) to directly examine whether FlatVLA converges to flatter regions in the parameter space.

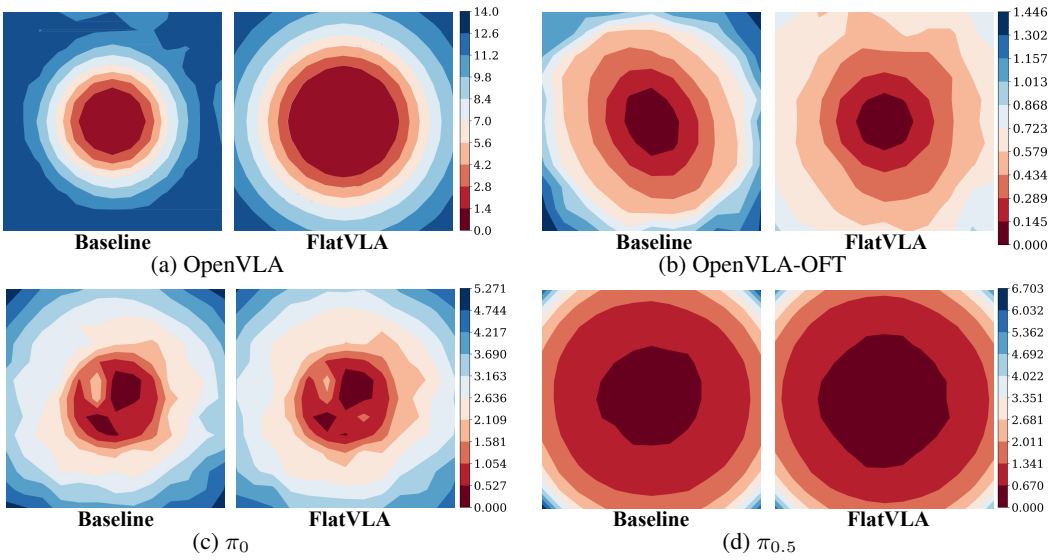


Figure 5: **Loss landscape visualization** across LIBERO benchmark. Compared to the baseline, FlatVLA consistently forms wider and flatter basins around the converged solution.

Loss Landscape Visualization Although SAM has been widely shown to promote flatter minima in various domains, whether the same geometric behavior emerges in VLA policies remains unclear. We thus examine the local landscape around the converged solutions to determine whether FlatVLA indeed learns flatter solutions than baseline. Specifically, we visualize the loss landscape on the training datasets for each LIBERO benchmark suite in Figure 5. Following common practice in loss landscape analysis, we evaluate the loss along two random directions in parameter space using filter-wise normalization (Li et al., 2018). Because evaluating the landscape over the full training set is computationally prohibitive, we randomly sample 10% of the training data from overall LIBERO benchmark, including all 4 suites, providing a computationally tractable approximation of the local loss surface (Bain et al., 2021).

As shown in Figure 5, FlatVLA consistently exhibits flatter loss surfaces than the baseline across models. This flatness is clearly observed for OpenVLA ($H = 1$), OpenVLA-OFT ($H = 7$), and $\pi_{0.5}$ ($H = 10$), covering single-step to mid-length prediction action chunk horizons. For π_0 with a substantially longer prediction horizon ($H = 50$), although the difference becomes less pronounced, FlatVLA still exhibits an overall flatter loss landscape across the evaluated region. Thus, these results indicate that FlatVLA consistently converges toward flatter regions in parameter space, even as the action prediction horizon increases.

D Limitations and Future Work

While FlatVLA improves robustness without modifying the model architecture or inference procedure, its sharpness-aware update requires an additional gradient computation at each training iteration. To quantify this overhead, we calculate the training cost in GPU-hours from the measured wall-clock time under each model's hardware configuration. Relative to their respective baselines, FlatVLA requires $1.95\times$ the GPU-hours for OpenVLA, $2.01\times$ for OpenVLA-OFT, $1.93\times$ for π_0, and $1.99\times$ for $\pi_{0.5}$. This nearly twofold increase in training cost remains a limitation when adapting large VLA models under constrained computational resources.

Improving the training efficiency of FlatVLA is therefore an important direction for future work. Beyond this computational aspect, we emphasize the potential of FlatVLA for broader VLA training paradigms beyond imitation learning. This direction is motivated by prior studies demonstrating the benefits of flatness-aware optimization for robust reinforcement learning through flatter solutions in policy parameter space (Lee & Yoon, 2025), language-model fine-tuning (Bahri et al., 2022), and preference-based post-training with Direct Preference Optimization (DPO) (Luo et al., 2026). These results motivate examining whether the connection between local smoothness and sequential error also extends to VLA policies optimized with reward or preference signals.