STRUCTRL: STRUCTURED ACTION-SPACE EXPLORATION FOR FLOW-BASED VLAS

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-based Vision-Language-Action (VLA) models are now widely used for continuous robotic manipulation, and online reinforcement learning (RL) is emerging as a key technique for adapting them to new tasks. Existing RL methods typically inject stochasticity inside the denoising chain, often through isotropic or temporally independent noise. However, effective robot exploration calls for structured noise: temporally smooth and scaled differently across action groups. We show that simply switching the in-chain noise to a structured form does not suffice: noise added at an intermediate flow time can be weakened by the remaining denoising steps before execution, a phenomenon we call *Structured Noise Dilution*. We propose **StructRL**, which avoids dilution by relocating policy stochasticity to the action space via three coupled choices: (i) a deterministic ODE decoder, (ii) structured noise injected directly in the action space, and (iii) terminal replay, which avoids assigning likelihoods to intermediate denoising states while shortening the policy-gradient path. This keeps structured exploration tied to the executed action while providing a tractable training signal for the flow decoder. Across three flow-based VLA models on multiple simulated manipulation benchmarks and two real-world tasks, StructRL improves exploration efficiency and OOD performance over prior in-chain baselines, demonstrating the effectiveness of structured action-space exploration for adapting flow-based VLA with RL.

1 INTRODUCTION

Vision-Language-Action (VLA) models cast robot control as instruction-conditioned action prediction, conditioning on visual observations and language instructions while typically pairing pretrained VLM backbones with continuous action heads trained on heterogeneous robot data (Kim et al., 2025b; Black et al., 2024a; Intelligence et al., 2025; Bjorck et al., 2025). Among continuous action-head designs, flow matching (Lipman et al., 2022) has become a common choice in recent VLA policies: it learns a velocity field and supports efficient few-step inference over action chunks. Recent flow-based VLAs such as π_0 (Black et al., 2024a), $\pi_{0.5}$ (Intelligence et al., 2025), and GR00T (Bjorck et al., 2025) are trained by large-scale behavior cloning. This imitation objective, however, is fundamentally bounded by the quality and coverage of the offline demonstrations. Improving these pretrained policies beyond the offline data therefore requires feedback from environment interaction, motivating recent RL approaches for flow-based VLAs (Chen et al., 2025; Wang et al., 2026).

In these RL approaches, the exploration distribution determines which actions are evaluated during interaction and therefore shapes what the policy can learn. For continuous robot control, exploratory actions should reflect the structure of robot actions (Fig. 1a). First, actions sampled within a chunk should be temporally smooth, since jittery actions can make execution unstable or unsafe. Second, noise should be scaled differently across action groups, since *position*, *rotation*, and *gripper* actions vary in range and safety sensitivity. Existing RL methods for flow-based VLAs (Chen et al., 2025; Wang & Yu, 2025; Wang et al., 2026) instead inject random noise inside the denoising chain, without explicitly controlling temporal correlation or action-group scale at the action level.

We find that this in-chain design makes structured exploration difficult to realize (Fig. 1b). When structured noise is injected at an intermediate flow time, the remaining denoising steps map the perturbed sample back toward the SFT action manifold. As a result, the temporal and action-group

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.

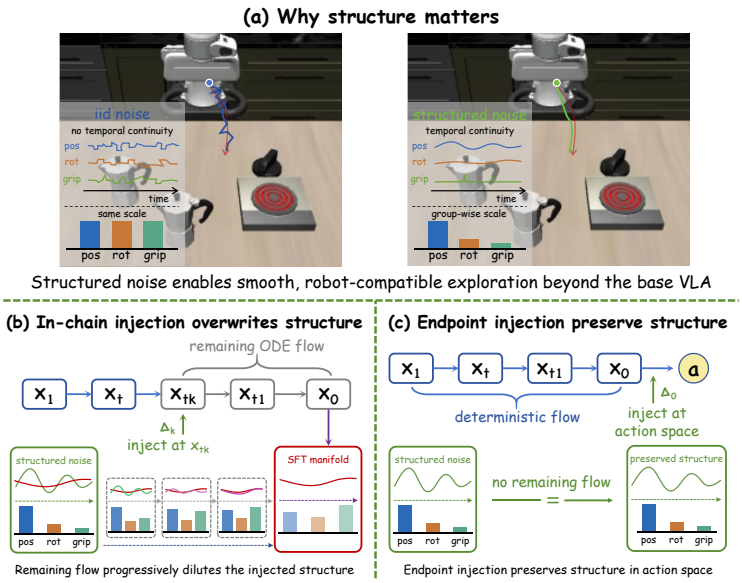


Figure 1: **Structured exploration for flow-based VLA-RL. (a)** Robot exploration benefits from smooth action chunks and group-aware noise scales. **(b)** In-chain noise can lose its intended structure during the remaining denoising steps. **(c)** StructRL keeps denoising deterministic and applies structured noise directly in the action space.

structure specified by the injected noise can be weakened or overwritten before the action is executed. We refer to this phenomenon as *Structured Noise Dilution* (Sec. 3).

This observation suggests a simple alternative: keep the flow decoder deterministic and place the exploration distribution directly in the action space (Fig. 1c). We propose **StructRL** based on this idea. Given the clean action chunk produced by ODE denoising, StructRL samples structured action-space noise with fixed AR(1) temporal correlation and state-conditioned scales constrained by group-specific bounds. During training, the policy likelihood is evaluated on the sampled action-space perturbation, so structured exploration remains tied to the executed action. StructRL then uses terminal replay to propagate policy-gradient updates through the final denoising steps without assigning likelihoods to intermediate denoising transitions. Across three flow-based VLA backbones (π_0, $\pi_{0.5}$, GR00T N1.5) on multiple manipulation benchmarks, StructRL improves exploration efficiency and consistently outperforms prior in-chain baselines on out-of-distribution and long-horizon manipulation tasks. We further deploy $\pi_{0.5}$ on a Franka robot and instantiate the same action-space likelihood in an asynchronous AWAC (Nair et al., 2020) learner. The comparisons between Gaussian exploration and StructRL on *pick-banana* and *plug-charger-in* tasks show faster and more effective online adaptation from sparse binary rewards. These results suggest that preserving structure in action-space exploration is important for efficient and robust VLA adaptation.

The main contributions can be summarized as follows:

- We identify *Structured Noise Dilution*: structured noise injected inside the flow chain can be weakened by the remaining denoising steps before it reaches the executed action.
- We build a latent-conditioned action-space RL framework for flow-based VLAs and introduce terminal replay as an efficient truncated surrogate that propagates gradients via only final denoising steps, without assigning likelihoods to intermediate denoising transitions.
- Based on this framework, we propose **StructRL**, which combines fixed AR(1) temporal correlation with state-conditioned exploration scales constrained by group-specific bounds for position, rotation, and gripper actions.
- Across multiple simulated manipulation benchmarks and two real-world tasks, we show that StructRL improves exploration efficiency and OOD performance over in-chain baselines and endpoint Gaussian exploration.

2 Related Work

Vision-Language-Action Models. Early VLAs such as RT-1 (Brohan et al., 2023), RT-2 (Zitkovich et al., 2023), OpenVLA (Kim et al., 2025b), and π_0-FAST (Pertsch et al., 2025) represent robot actions as language tokens and train the model autoregressively. Another line keeps the vision–language backbone fixed and trains an attached continuous action head (Team et al., 2024; Kim et al., 2025a). Recent VLA policies often use flow-matching (Lipman et al., 2022; Liu et al., 2023b) or diffusion (Chi et al., 2025; Ze et al., 2024) decoders to model continuous action chunks with few-step inference. Our work focuses on RL adaptation of representative flow-based VLAs, including π_0 (Black et al., 2024a), $\pi_{0.5}$ (Intelligence et al., 2025), and GR00T N1.5 (Bjorck et al., 2025), which are trained by large-scale imitation learning on heterogeneous robot data.

RL for Generative Robot Policies. RL has been used to optimize generative policies. DDPO (Black et al., 2024b) treats the denoising process of a diffusion model as a multi-step stochastic policy and applies policy-gradient updates to the denoising chain. In offline continuous-control RL, Diffusion-QL (Wang et al., 2022), IDQL (Hansen-Estruch et al., 2023), CPQL (Chen et al., 2024), and QAM (Li & Levine, 2026) combine generative policy classes with value-based objectives. For flow-based VLAs, π_{RL} (Chen et al., 2025) introduces two methods: Flow-Noise models denoising as a discrete-time MDP with a learnable noise network, and Flow-SDE converts ODE denoising into an SDE for exploration. Flow-CPS (Wang & Yu, 2025) proposes a coefficient-preserving stochastic sampler for RL with flow matching, while π-StepNFT (Wang et al., 2026) introduces a critic and likelihood-free step-wise negative-aware fine-tuning objective for flow-based VLAs. These works make RL feasible for denoising policies, but their exploration signal is not defined as a structured distribution over executed actions.

Structured Exploration in Continuous-Control RL. Continuous-control RL has explored several ways to structure exploration, including temporally correlated noise across environment steps (OU noise in DDPG (Lillicrap et al., 2016), colored noise (Eberhard et al., 2023; Hollenstein et al., 2024)), learned action-dimension coupling (Chiappa et al., 2023; Yang et al., 2026), state-dependent linear noise (Raffin et al., 2022), parameter-space perturbation (Plappert et al., 2018), and on-manifold exploration (Jin et al., 2025). These methods mainly define structure on policy actions or parameters. Flow-based VLAs complicate this design because action samples are produced through a denoising chain, while existing RL methods usually place stochasticity inside the chain. StructRL moves the structured perturbation to the action space and uses terminal replay to propagate policy-gradient updates through the final denoising steps.

3 Structured Noise Dilution

This section studies why structure injected inside the denoising chain may not survive to the executed action. We first explain how the remaining denoising steps can filter an intermediate perturbation, then test this effect on a GR00T N1.5 SFT model by injecting noise with controlled temporal correlation and per-group scale at different locations. We finally discuss why the structure induced by the SFT model is not a substitute for task-specific exploration.

Mechanism. In-chain exploration specifies noise before the decoder has finished predicting an action. The remaining denoising steps are still those of the SFT model, so the perturbed state is processed toward demonstration-like action regions. The action-level perturbation is therefore not the injected noise itself: its temporal correlation and action-group scale can be changed by the rest of the decoder, which we call *Structured Noise Dilution*.

Empirical Verification. We test whether Structured Noise Dilution appears in flow-based VLAs, and Table 1 provides direct empirical evidence. Specifically, we use a GR00T N1.5 SFT model on LIBERO-Long (Liu et al., 2023a), and inject structured perturbations Δ with either a prescribed AR(1) coefficient ρ_{inj} along the chunk axis or prescribed per-group magnitudes across position, rotation, and gripper actions. The perturbation is inserted at one of two sites: *in-chain*, before the remaining denoising steps, or *action-space*, after denoising completes. We then measure the structure of the resulting executed action. The table reports one representative temporal-correlation setting and one representative action-group scale setting.

For strong temporal correlation ($\rho_{\text{inj}} = 0.95$), action-space injection changes the measured output correlation, while in-chain injection stays close to the SFT baseline. For the position-heavy scale ratio $R_{\text{approach}} = (1.0, 0.5, 0.05)$ over (position, rotation, gripper), action-space injection increases the position standard deviation from 0.10 to 0.23, whereas in-chain injection changes it only from 0.10 to 0.11. Additional experimental details are provided in App. C.

Table 1: **Structured noise dilution.** Output statistics after injecting structured noise into a GR00T N1.5 SFT model on LIBERO-Long. We compare action-space injection with in-chain injection under two requested structures: strong temporal correlation ($\rho_{\text{inj}} = 0.95$) and a position-heavy scale ratio $R_{\text{approach}} = (1.0, 0.5, 0.05)$, both using base noise level $\sigma = 0.2$. Action-space injection better follows the requested structure, while in-chain injection stays close to the SFT baseline.

	Action-space			In-chain		
	pos	rot	grip	pos	rot	grip
$\hat{\rho}_{\text{SFT}}$	0.78	0.72	0.57	0.78	0.72	0.57
$\rho_{\text{inj}} = 0.95$	0.70	0.64	0.63	0.79	0.72	0.58
$\hat{\sigma}_{\text{SFT}}$	0.10	0.08	0.21	0.10	0.08	0.21
R_{approach}	0.23	0.13	0.19	0.11	0.08	0.19

SFT Structure Constrains Exploration. The structure that survives in-chain injection comes from the SFT model. The SFT-induced correlation $\rho_{\text{SFT}}(o)$ reflects the behavior distribution in the offline data (Ross et al., 2011; Atkeson & Schaal, 1997), and may be useful as an imitation prior. However, if exploration is mainly shaped by this prior, the policy remains biased toward actions that the SFT model already favors. This makes it harder to reach useful actions outside the SFT prior and can reduce exploration efficiency during RL process. These limitations motivate us to move structured exploration to the action space, where temporal correlation and action-group scale can be specified directly.

4 Method: StructRL

We propose StructRL, an action-space RL framework for flow-based VLAs, illustrated in Fig. 2. StructRL keeps the flow decoder deterministic, injects structured noise at action output, and uses terminal replay for efficient policy optimization. In this section, we first review flow-based VLA policies and RL likelihood problem, then describe the action-space RL framework, and finally instantiate the noise distribution with temporal correlation and state-conditioned scaling under group-specific bounds.

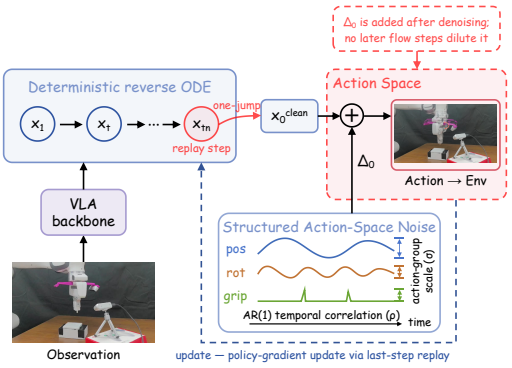


Figure 2: **The StructRL framework.** The reverse flow runs as a deterministic ODE to predict a clean action x_0^{clean}. StructRL samples structured noise Δ_0 directly in the action space, combining fixed AR(1) temporal correlation with state-conditioned scales constrained by group-specific bounds. During RL, the likelihood is evaluated on this action-space perturbation, and terminal replay propagates gradients through the final denoising steps.

4.1 Preliminaries

Flow-Matching VLAs. A flow-based VLA predicts an action chunk by integrating a learned velocity field $v_\theta(x_t, t, o)$, where o is the multimodal observation and $x_t \in \mathbb{R}^{C\times D}$ is a noised action chunk. Starting from $x_1 \sim \mathcal{N}(0, I)$, the reverse flow is integrated with T Euler steps:

$$x_{t_k - \delta} = x_{t_k} - \delta\, v_\theta(x_{t_k}, t_k, o), \quad k = T, T-1, \ldots, 1. \tag{1}$$

The final state x_0 is used as the clean action. We denote by $F^{\theta}_{k\to 0}$ the deterministic map from x_{t_k} to x_0 obtained by the remaining k denoising steps.

Policy Likelihood for Flow-based RL. Likelihood-based RL objectives require a tractable policy density, such as policy ratios in PPO (Schulman et al., 2017) or entropy terms in SAC (Haarnoja et al., 2018). For a flow-matching VLA, the executed action is obtained after the multi-step denoising process in Eq. 1, so its likelihood is not directly exposed. Existing flow-based VLA RL methods attach likelihoods to denoising-chain transitions (Chen et al., 2025). Such chain-level likelihoods are tied to intermediate denoising states, while action-space structured noise is defined on the perturbation added to the executed output action.

4.2 ACTION-SPACE RL WITH LATENT-CONDITIONED REPLAY

Action-space exploration avoids modification by subsequent denoising steps. We therefore keep the flow decoder deterministic and place policy stochasticity directly on its output. Starting from $x_1 \sim \mathcal{N}(0, I)$, the reverse ODE predicts a clean action chunk

$$x_0^{\text{clean}} = F_{T\to 0}^{\theta}(x_1; o), \tag{2}$$

after which an action-space perturbation Δ_0 is sampled from an exploration distribution $p(\Delta_0)$ and added to the clean output:

$$a = x_0^{\text{clean}} + \Delta_0. \tag{3}$$

For the Gaussian baseline, we use $p(\Delta_0) = \mathcal{N}(0, \sigma^2 I)$. Sec. 4.3 replaces this isotropic distribution with the structured distribution used by StructRL.

Conditioned on the sampled initial latent x_1, the action has a tractable density

$$\pi_\theta(a \mid o, x_1) = p\big(a - F_{T\to 0}^{\theta}(x_1; o)\big). \tag{4}$$

Because x_1 is drawn from a fixed, parameter-independent Gaussian prior, it can be retained as an auxiliary policy variable during optimization. Likelihood-based RL can therefore operate on the latent-conditioned density in Eq. 4 without assigning stochastic likelihoods to intermediate denoising transitions. A formal derivation is provided in App. A.

Evaluating the exact current likelihood in Eq. 4 after the policy parameters are updated requires recomputing $F_{T\to 0}^{\theta}(x_1; o)$ from the stored initial latent x_1. Optimizing through this exact likelihood further requires backpropagating through all T denoising steps, increasing both computation and the gradient-path length. To obtain a cheaper local surrogate, we use *terminal replay*.

During rollout, we cache a late denoising state x_{t_n}, where n denotes the number of terminal denoising steps retained for replay. During optimization, this cached upstream state is treated as fixed and only the final n denoising steps are recomputed:

$$\hat{x}_0^{\theta,n} = F_{n\to 0}^{\theta}(x_{t_n}; o), \qquad \Delta_0^{\theta,n} = a - \hat{x}_0^{\theta,n}. \tag{5}$$

The corresponding replay-conditioned log-likelihood is

$$\tilde{\ell}_n(\theta) = \log p\Big(\Delta_0^{\theta,n}\Big). \tag{6}$$

For $n < T$, Eq. 6 forms a truncated surrogate by keeping cached upstream state fixed and propagating gradients only through the final n denoising steps. We use $n = 1$ by default, which we refer to as *last-step replay*. The relation between full latent replay and terminal replay is detailed in App. A.

In simulation, we use the replay-conditioned likelihood in PPO. For the real-world off-policy setting, we use $T = 1$, where the cached replay latent is exactly the policy-independent initial noise x_1. Terminal replay therefore reduces to full latent replay, and the likelihood is evaluated using Eq. 4.

4.3 STRUCTURED NOISE INSTANTIATION

We now instantiate the generic action-space exploration distribution $p(\Delta_0)$ in Sec. 4.2 as a structured distribution $p_\phi(\Delta_0 \mid h_\theta)$, where $h_\theta \in \mathbb{R}^{C\times d_h}$ denotes the terminal hidden representation produced by the action decoder, conditioned on the observation-language prefix encoded by the frozen VLM. Before being passed to the noise modulator, h_θ is detached from the action-decoder computation graph. The distribution combines a fixed AR(1) process that controls temporal correlation within the action chunk with state-conditioned action scales predicted from h_θ.

State-Conditioned Action Scaling. Robot manipulation policies commonly represent actions as relative end-effector motion, typically covering position, rotation, and gripper control. These action groups have different units, ranges, and safety sensitivity, so a single isotropic noise scale is poorly matched to this action representation. We partition the D action dimensions into three groups, $\mathcal{G} = \{\mathcal{G}_{\text{pos}}, \mathcal{G}_{\text{rot}}, \mathcal{G}_{\text{grip}}\}$. A global base noise level σ^{base} sets the overall noise magnitude, while group-specific bounds control the admissible scale range. A lightweight MLP head f_ϕ predicts state-conditioned element-wise multipliers from h_θ:

$$\sigma_{c,d} = \sigma^{\text{base}} \cdot \text{clip}\left(f_\phi(h_\theta)_{c,d}, [\alpha_{g(d)}, \beta_{g(d)}]\right), \tag{7}$$

where $g(d)$ is the group of dimension d. The group-specific bounds $[\alpha_g, \beta_g]$ keep the learned exploration scale within a controlled range while allowing it to vary with the current policy state.

AR(1) Temporal Correlation. Within an action chunk, adjacent actions are executed sequentially, so independent perturbations may introduce unnecessary jitter. Therefore, we sample an AR(1) noise sequence for each action dimension:

$$\varepsilon_{1,d} \sim \mathcal{N}(0,1), \qquad \varepsilon_{c+1,d} = \rho_{g(d)}\, \varepsilon_{c,d} + \sqrt{1-\rho_{g(d)}^2}\, \zeta_{c,d}, \qquad \zeta_{c,d} \sim \mathcal{N}(0,1), \tag{8}$$

where $\rho_{g(d)} \in [0,1)$ is a fixed group-specific coefficient controlling temporal correlation. The final action-space perturbation combines the state-conditioned scale and the AR(1) noise:

$$\Delta_{0,c,d} = \sigma_{c,d}\varepsilon_{c,d}. \tag{9}$$

This construction preserves explicit temporal structure through the fixed AR(1) process while adapting the exploration magnitude to the current state.

Structured Log-Probability. Under terminal replay, Eq. 5 gives the replayed action-space perturbation $\Delta_0^{\theta,n}$. We evaluate it under the structured distribution in Eq. 9. The AR(1) density is applied to the corresponding normalized noise coordinates, together with the scale change-of-variables term:

$$\log p_\phi\left(\Delta_0^{\theta,n} \mid h_\theta\right) = \sum_{d=1}^{D}\left[\log \mathcal{N}\left(\varepsilon_{1,d}^{\theta,n}; 0, 1\right) + \sum_{c=1}^{C-1} \log \mathcal{N}\left(\varepsilon_{c+1,d}^{\theta,n}; \rho_{g(d)}\varepsilon_{c,d}^{\theta,n}, 1-\rho_{g(d)}^2\right)\right] - \sum_{c,d} \log \sigma_{c,d}, \tag{10}$$

where $\varepsilon_{c,d}^{\theta,n} = \Delta_{0,c,d}^{\theta,n}/\sigma_{c,d}$ is the AR(1) coordinate corresponding to the replayed perturbation under the current state-conditioned scale. This structured density instantiates the generic replay-conditioned likelihood in Eq. 6. The detailed change-of-variables derivation is provided in App. D.

Regularization. The state-conditioned scale σ controls the magnitude of action-space exploration, so unconstrained updates can collapse or over-amplify the perturbations. We therefore regularize σ with three lightweight terms: a dimension-entropy term that discourages scale collapse onto a few action dimensions, a budget term that keeps the overall noise magnitude near the target level, and a temporal smoothness term that avoids abrupt scale changes along the chunk. The full form is given in App. E.

5 Experiments

5.1 Experimental Setup

Benchmarks. We conduct experiments on three standard simulation benchmarks: LIBERO (Liu et al., 2023a), ManiSkill Tao et al. (2025), and CALVIN (Mees et al., 2022). Hyperparameter settings and CALVIN results are provided in App. H and App. G, respectively. All experiments are conducted on 4 NVIDIA A800 GPUs.

Baseline variants. We adopt π_0 (Black et al., 2024a), $\pi_{0.5}$ (Intelligence et al., 2025), and GR00T N1.5 (Bjorck et al., 2025) as the base VLA models throughout the experiments. For simulation, we train the RL variants with PPO, since the environments allow parallel rollout collection. The physical experiments instead use the off-policy AWAC instantiation described in Sec. 5.4 and App. F. StructRL is compared against several baselines: the SFT model, π_{RL} (Chen et al., 2025) (Flow-SDE+PPO), and π-StepNFT (Wang et al., 2026). Flow-SDE injects noise inside the denoising chain and trains with PPO, while π-StepNFT keeps SDE-based in-chain exploration but replaces likelihood-based policy-gradient training with a step-wise fine-tuning objective. We define the version that adds Gaussian noise within our action-space RL framework as *Baseline*. StructRL further replaces this Gaussian noise with structured action-space noise.

5.2 Main Results

Results on LIBERO. Table 2 presents the results across the four LIBERO suites. On GR00T N1.5, where the few-shot SFT model reaches only 52.5 % average, StructRL pushes the average success rate to 99.0 %, outperforming Flow-SDE+PPO by 2.5% with consistent gains on the four suites. On π_0, StructRL yields a modest improvement against Flow-SDE+PPO. Because all RL methods approach saturation on LIBERO, we treat this benchmark primarily as an in-distribution sanity check; ManiSkill OOD and the learning curves in Sec. 5.3 provide the more diagnostic comparisons of injection location and structured exploration.

Table 2: **LIBERO benchmark results.** Average success rates (%) across four task suites, each containing 10 tasks. The few-shot SFT models are directly taken from the π_{RL} checkpoints.

Model	Spatial	Object	Goal	Long	Avg.	Δ Avg.
GR00T N1.5 – Few-shot SFT	41.4	58.6	48.2	61.9	52.5	—
+ π_{RL} (Flow-SDE+PPO)	96.6	**100**	93.8	95.6	96.5	+44.0
+ Baseline	96.2	99.4	91.8	95.6	95.8	+43.3
+ StructRL	**99.2**	**100**	**97.6**	**99.0**	**99.0**	**+46.3**
π_0 – Few-shot SFT	65.3	64.4	49.8	51.2	57.6	—
+ π_{RL} (Flow-SDE+PPO)	98.4	99.4	96.2	90.2	96.0	+38.4
+ π-StepNFT	93.5	98.0	83.7	86.7	90.5	+32.9
+ Baseline	**98.8**	99.0	97.2	90.2	96.3	+38.7
+ StructRL	**98.8**	**99.8**	**98.4**	**91.8**	**97.2**	**+39.6**
$\pi_{0.5}$ – Few-shot SFT	84.6	95.4	84.6	43.9	77.1	—
+ π_{RL} (Flow-SDE+PPO)	99.6	100	98.8	**93.0**	**97.9**	**+20.8**
+ π-StepNFT	97.8	**100**	98.2	79.8	94.0	+16.9
+ Baseline	99.4	**100**	93.8	86.0	94.8	+17.7
+ StructRL	**99.6**	**100**	**98.8**	90.2	97.2	+20.1

Table 3: **ManiSkill benchmark results.** Success rates (%) on in-distribution (IND) tasks and three OOD splits: Vision (5 tasks), Semantic (5 tasks), and Execution (2 tasks). Avg. reports the mean success rate across the OOD splits.

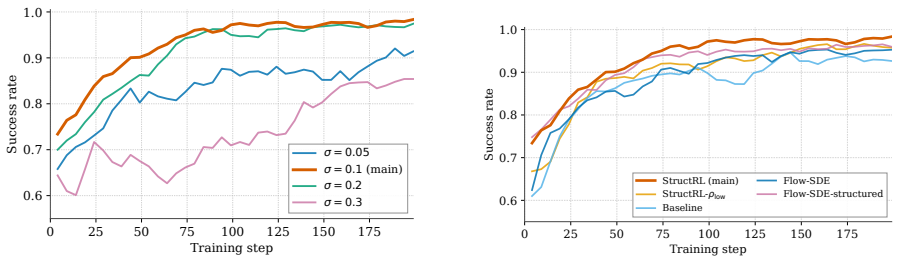
Model	IND	OOD			
		Vision	Semantic	Execution	Avg.
π_0 – SFT	38.4	32.6	8.4	13.2	18.1
+ π_{RL} (Flow-SDE+PPO)	78.8	61.1	25.4	31.5	39.3
+ π-StepNFT	79.2	69.1	49.1	33.1	50.4
+ Baseline	84.1	79.4	63.3	**59.1**	**67.3**
+ StructRL	**87.8**	**80.0**	**64.2**	56.2	66.8
$\pi_{0.5}$ – SFT	40.1	38.8	16.6	22.3	25.9
+ π_{RL} (Flow-SDE+PPO)	**90.9**	68.0	34.5	45.4	49.3
+ π-StepNFT	85.4	76.9	56.6	45.1	59.5
+ Baseline	85.9	76.2	62.2	59.4	65.9
+ StructRL	**90.9**	**78.3**	**68.4**	**64.2**	**70.3**

Results on ManiSkill. Table 3 reports the simulated results on the ManiSkill benchmark. StructRL achieves the highest in-distribution success rate of 87.8% on π_0, and matches the best on $\pi_{0.5}$. For OOD evaluation, the action-space RL framework shows a clear advantage over Flow-SDE's in-chain noise injection. Moving only the injection location from Flow-SDE to the endpoint Gaussian Baseline improves the OOD average by 28.0 points on π_0 (39.3%→67.3%) and 16.6 points on $\pi_{0.5}$ (49.3%→65.9%). This identifies endpoint action-space exploration, rather than AR structure alone, as the main source of the large final OOD gain. StructRL further changes the endpoint covariance and is most consistently beneficial in learning efficiency; its final performance generally matches or improves upon the Gaussian Baseline. Thus, the controlled results support two distinct conclusions: the injection location expands effective exploration beyond the flow-shaped SFT distribution, while temporal and group structure make that exploration more efficient.

5.3 Ablation Studies

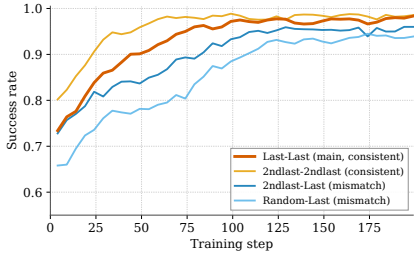
All ablations use GR00T N1.5 with PPO on LIBERO-Long, the LIBERO suite that most directly tests long-horizon execution. For a fair comparison, all runs use the same 200 training iterations and are evaluated every 5 iterations (Fig. 3).

(a) Sensitivity to the Noise Level σ^{base}. We set $\sigma^{\text{base}} \in \{0.05, 0.1, 0.2, 0.3\}$ to test how the magnitude of exploration noise affects training stability. As shown in Fig. 3(a), small noise levels under-explore,

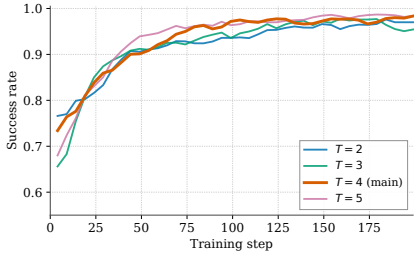


(a) Action-space noise level σ^{base}

(b) Noise injection scheme



(c) Train–inference consistency



(d) Number of denoising steps T

Figure 3: **Four ablations on LIBERO-Long with GR00T N1.5. (a)** Moderate action-space noise gives stable online improvement. **(b)** Structured action-space noise explores more efficiently than Gaussian or in-chain variants. **(c)** Matched train–inference replay improves performance, with larger matched jumps further improving exploration efficiency. **(d)** Default last-step replay remains effective across different denoising horizons.

while large noise levels over-perturb the executed actions and destabilize training. The intermediate setting provides learnable signals over a broader action region, enabling efficient online improvement.

(b) Noise Injection Scheme. We compare five ways of injecting exploration noise: *action-space Gaussian noise* (the Baseline), *StructRL*, *StructRL*$-\rho_{low}$ (reduced temporal correlation), *Flow-SDE* (isotropic in-chain noise), and *Flow-SDE*$-$*structured* (the same structured noise injected in-chain). These controls separate two factors. Flow-SDE versus the Gaussian Baseline changes the injection location, whereas the Gaussian Baseline versus StructRL changes only the endpoint noise structure. Fig. 3(b) shows that StructRL reaches 90% success in about 40 training steps, compared with about 80 steps for the endpoint Gaussian Baseline. The final values become close after saturation, so this ablation supports an efficiency benefit from temporal and group structure rather than a universal final-performance advantage.

(c) Train–Inference Consistency. Terminal replay can cover one or more final denoising steps, while the inference-time ODE step can either match or differ from the replay depth. We therefore compare four configurations. **Last-Last:** last-step replay ($n = 1$) in training and last-step inference. **2ndlast-2ndlast:** second-last-step replay ($n = 2$) in training and second-last-step inference. **2ndlast-Last:** second-last-step replay ($n = 2$) in training but last-step inference. **Random-Last:** randomly selected replay depth in training and last-step inference. The first two settings are consistent between training and inference, while the last two deliberately introduce mismatch. As shown in Fig. 3(c), the train–inference consistent settings substantially outperform the mismatched settings. We also find that a larger replay jump brings a more evident gain in exploration efficiency.

(d) Robustness Evaluation. To evaluate the robustness of the default last-step replay setting ($n = 1$), we vary the number of denoising steps as $T \in \{2, 3, 4, 5\}$. As shown in Fig. 3(d), all settings show similar convergence speeds and achieve comparable final performance. This result indicates that the truncated terminal-replay objective remains effective across different denoising horizons.

5.4 Real-World Experiment

We evaluate StructRL in the real world on *pick-banana*, a pick-and-place (PnP) task, and *plug-charger-in*, a fine-manipulation task (Fig. 4). Because physical rollouts have low throughput and each


Pick-Banana


Plug-Charger-In

Task / metric	Baseline	StructRL
Pick-banana, SR at 60 min	56% (14/25)	**84% (21/25)**
Plug-charger-in, convergence	∼35 min	**∼30 min**

Figure 4: **Real-world tasks and online-RL comparisons.** Both task-specific policies are initialized by SFT on 10 pre-collected demos and achieve 0% success rate (SR) before online RL.

transition is expensive to collect, we combine StructRL with AWAC (Nair et al., 2020) to obtain an off-policy real-world RL algorithm that can repeatedly reuse collected data.

Algorithm: To retain the unlimited transition-reuse advantage of off-policy learning, we set the ODE denoising horizon to a single step ($T = 1$). The cached replay latent is therefore the initial Gaussian noise x_1, whose sampling distribution is independent of the policy parameters. Each chunk-based transition is stored as

$$\mathcal{T}_t = \big(o_t, a_t^b, \mathbf{r}_t, \mathbf{d}_t, o_{t+C}, x_1\big), \tag{11}$$

where o_t and o_{t+C} are the current and next observations; a_t^b is the executed action chunk, either generated by the policy with exploration noise or human intervention; $\mathbf{r}_t$ and $\mathbf{d}_t$ are the rewards and termination/truncation indicators within the chunk. Given the stored x_1, the current actor recomputes the one-step action chunk without noise injection and evaluates a_t^b under the StructRL action-space distribution. Its AWAC objective is

$$\mathcal{L}_{\text{AWAC}}(\theta) = -\mathbb{E}_{\mathcal{T}_t \sim \mathcal{D}} \left[w_t \log \pi_\theta(a_t^b \mid o_t, x_1) \right], \tag{12}$$

where the advantage weight is

$$w_t = \exp\left(\text{clip}\left(\frac{\min_i Q_i'(o_t, a_t^b) - V(o_t)}{\beta}, -3, 3 \right) \right), \qquad \beta = 0.5. \tag{13}$$

Thus, previously collected rollout actions and human corrections can be repeatedly reused through advantage-weighted BC, while higher-advantage actions receive larger likelihood updates.

Setup: We conduct the experiments on a FR3 robot, using $\pi_{0.5}$ as the policy backbone. The policy observes two 256×256 RGB views and a 19-dimensional proprioceptive state. *Pick-banana* uses $C = 10$ and a 7-DoF action while *Plug-charger-in* uses $C = 5$ and a 6-DoF action with fixed gripper dimension. Both tasks use delta end-effector pose actions to represent action space. All experiments use a binary sparse reward, with a reward of 10 for successful task completion and 0 otherwise.

Results: For each task, we first perform SFT on 10 pre-collected demos. This initialization provides only a coarse motion trend and achieves 0% success rate. We then run online RL with few human interventions. In *pick-banana*, the limited demos do not cover the banana's varying positions, so the initial policy usually fails during grasping. After 60 minutes of online RL, StructRL reaches an 84% success rate (21/25), whereas the Gaussian baseline reaches 56% (14/25), as shown in Fig. 4. For *plug-charger-in*, the SFT prior mainly produces a downward insertion motion but cannot make the necessary lateral alignment corrections. StructRL reaches a stable, fully successful regime after approximately 30 minutes, while the Gaussian baseline requires approximately 35 minutes (Fig. 4). These results show that structured exploration also improves exploration efficiency in real-world online RL. Additional implementation, training, and safety details are provided in App. F.

6 Conclusion

We presented StructRL, an action-space RL framework for RL adaptation of flow-based VLAs. Motivated by structured noise dilution, StructRL keeps the flow decoder deterministic, injects temporally correlated exploration with state-conditioned scales directly on the executed action chunk, and uses terminal replay for efficient policy optimization. Controlled simulation results show that moving exploration to the action space accounts for most of the OOD gain, while structuring the action-space noise primarily improves learning efficiency. Real-world experiments further show that StructRL enables more efficient online adaptation than Gaussian exploration.

AI USE STATEMENT

We used generative AI tools to assist with language polishing and grammar checking of the manuscript, including improving the clarity and readability of the text. The authors are responsible for verifying that AI-assisted edits preserve the intended scientific meaning and accuracy of the manuscript. We take full responsibility for the final content of this work, including text revised with the assistance of generative AI.

REFERENCES

Christopher G. Atkeson and Stefan Schaal. Robot learning from demonstration. In *International Conference on Machine Learning (ICML)*, 1997.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

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

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pp. 4965–4987, 2024b.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. In *Robotics: Science and Systems*. Robotics: Science and Systems Foundation, 2023.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, et al. π_{rl}: Online rl fine-tuning for flow-based vision-language-action models. *arXiv preprint arXiv:2510.25889*, 2025.

Yuhui Chen, Haoran Li, and Dongbin Zhao. Boosting continuous control with consistency policy. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, pp. 335–344, 2024.

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

Alberto Silvio Chiappa, Alessandro Marin Vargas, Ann Huang, and Alexander Mathis. Latent exploration for reinforcement learning. *Advances in Neural Information Processing Systems*, 36: 56508–56530, 2023.

Onno Eberhard, Jakob Hollenstein, Cristina Pinneri, and Georg Martius. Pink noise is all you need: Colored noise exploration in deep reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

Jakob Hollenstein, Georg Martius, and Justus Piater. Colored noise in ppo: Improved exploration and performance through correlated action sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 12466–12472, 2024.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Yang Jin, Jun Lv, Han Xue, Wendi Chen, Chuan Wen, and Cewu Lu. Soe: Sample-efficient robot policy self-improvement via on-manifold exploration. *arXiv preprint arXiv:2509.19292*, 2025.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025a.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. Openvla: An open-source vision-language-action model. In *Conference on Robot Learning*, pp. 2679–2713. PMLR, 2025b.

Qiyang Li and Sergey Levine. Q-learning with adjoint matching. *arXiv preprint arXiv:2601.14234*, 2026.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023a.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations (ICLR)*, 2023b.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

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

Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. In *International Conference on Learning Representations*, 2018.

Antonin Raffin, Jens Kober, and Freek Stulp. Smooth exploration for robotic reinforcement learning. In *Conference on robot learning*, pp. 1634–1644. PMLR, 2022.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, et al. Maniskill3: Gpu parallelized robot simulation and rendering for generalizable embodied ai. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

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

Feng Wang and Zihao Yu. Coefficients-preserving sampling for reinforcement learning with flow matching. *arXiv preprint arXiv:2509.05952*, 2025.

Siting Wang, Xiaofeng Wang, Zheng Zhu, Minnan Pei, Xinyu Cui, Cheng Deng, Jian Zhao, Guan Huang, Haifeng Zhang, and Jun Wang. π-stepnft: Wider space needs finer steps in online rl for flow-based vlas. *arXiv preprint arXiv:2603.02083*, 2026.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

Jiarui Yang, Bin Zhu, Jingjing Chen, and Yu-Gang Jiang. Actor-critic for continuous action chunks: A reinforcement learning framework for long-horizon robotic manipulation with sparse reward. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18692–18700, 2026.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. In *Robotics: Science and Systems*. Robotics: Science and Systems Foundation, 2024.

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

A Latent-Conditioned Policy and Terminal Replay

This appendix provides additional details for the latent-conditioned action-space formulation in Sec. 4.2 and clarifies the relation between full latent replay and terminal replay. Throughout this appendix, $p(\Delta_0)$ denotes the action-space exploration distribution, with its internal parameterization and conditioning suppressed for notational simplicity; for StructRL, it corresponds to $p_\phi(\Delta_0 \mid h_\theta)$ defined in Sec. 4.3.

Latent-conditioned policy. The policy samples an initial latent $x_1 \sim p_0(x_1) = \mathcal{N}(0, I)$ and generates

$$a = F^\theta_{T\to 0}(x_1; o) + \Delta_0, \qquad \Delta_0 \sim p(\Delta_0). \tag{14}$$

Conditioned on x_1, the corresponding action density is

$$\pi_\theta(a \mid o, x_1) = p\left(a - F^\theta_{T\to 0}(x_1; o)\right). \tag{15}$$

The joint policy over the auxiliary latent and action can therefore be written as

$$\bar{\pi}_\theta(x_1, a \mid o) = p_0(x_1)\, \pi_\theta(a \mid o, x_1). \tag{16}$$

Let θ_{old} denote the policy parameters used to collect the rollout. Since $p_0(x_1)$ is fixed and independent of the policy parameters, it cancels in the likelihood ratio:

$$\frac{\bar{\pi}_\theta(x_1, a \mid o)}{\bar{\pi}_{\theta_{\text{old}}}(x_1, a \mid o)} = \frac{\pi_\theta(a \mid o, x_1)}{\pi_{\theta_{\text{old}}}(a \mid o, x_1)}. \tag{17}$$

Thus, likelihood-based RL can operate directly on the latent-conditioned action density without evaluating the marginal action likelihood $\pi_\theta(a \mid o)$.

Full latent replay. If the initial latent x_1 is retained during rollout, the latent-conditioned likelihood under the updated policy can be evaluated by recomputing the complete deterministic flow:

$$\hat{x}_0^{\theta,\text{full}} = F^\theta_{T\to 0}(x_1; o), \qquad \Delta_0^{\theta,\text{full}} = a - \hat{x}_0^{\theta,\text{full}}. \tag{18}$$

The corresponding log-likelihood is

$$\ell_{\text{full}}(\theta) = \log p\left(\Delta_0^{\theta,\text{full}}\right) = \log \pi_\theta(a \mid o, x_1). \tag{19}$$

Full latent replay therefore requires recomputing the complete deterministic trajectory and backpropagating through all T denoising steps.

Terminal replay. During rollout, the cached intermediate state is generated as

$$x_{t_n} = F^{\theta_{\text{old}}}_{T\to n}(x_1; o), \tag{20}$$

where n denotes the number of terminal denoising steps retained for replay. During optimization, terminal replay treats x_{t_n} as fixed and recomputes only the final n steps:

$$\hat{x}_0^{\theta,n} = F^{\theta}_{n\to 0}(x_{t_n}; o), \qquad \Delta_0^{\theta,n} = a - \hat{x}_0^{\theta,n}. \tag{21}$$

To see the approximation introduced by terminal replay, the full flow can be written as the composition

$$F^{\theta}_{T\to 0}(x_1; o) = F^{\theta}_{n\to 0}\left(F^{\theta}_{T\to n}(x_1; o); o\right). \tag{22}$$

Its parameter derivative contains both the direct dependence of the terminal flow on θ and the dependence through the upstream denoising state:

$$\nabla_\theta F^{\theta}_{T\to 0} = \frac{\partial F^{\theta}_{n\to 0}}{\partial \theta} + \frac{\partial F^{\theta}_{n\to 0}}{\partial x_{t_n}} \frac{\partial F^{\theta}_{T\to n}}{\partial \theta}. \tag{23}$$

By keeping the cached x_{t_n} fixed, terminal replay retains the direct terminal-path gradient while omitting gradients through the upstream denoising trajectory. It therefore provides a truncated surrogate with a shorter backpropagation path and lower computation per policy update.

Single-step case. For the real-world experiments, we use $T = 1$. In this case, the cached replay latent is the initial Gaussian latent x_1, whose sampling distribution is independent of the policy parameters. Terminal replay therefore coincides with full latent replay, and the likelihood is evaluated directly using Eq. 4.

B StructRL Training Pseudocode

This appendix provides the training pseudocode for the framework introduced in Sec. 4.2. Algorithm 1 summarizes the action-space RL framework used by both the Gaussian baseline and StructRL. During rollout, the flow decoder predicts a clean action through the deterministic ODE and exploration noise is added directly in the action space. During policy updates, terminal replay recomputes only the final denoising steps from a cached intermediate state.

Algorithm 1 StructRL

Require: Flow decoder F^θ initialized from the SFT model; action-space distribution p; likelihood-based RL objective $\mathcal{L}_{\text{RL}}$

1: **for** each rollout iteration **do**
2: Sample $x_1 \sim \mathcal{N}(0, I)$ and run the deterministic reverse ODE.
3: Cache x_{t_n} and obtain $x_0^{\text{clean}} = F^{\theta}_{T\to 0}(x_1; o)$.
4: Sample $\Delta_0 \sim p$ and execute $a = x_0^{\text{clean}} + \Delta_0$.
5: Store the transition together with x_1, x_{t_n}, and the rollout log-likelihood when required by the RL objective.
6: **end for**
7: **for** each policy update **do**
8: Replay the final n denoising steps: $\hat{x}_0^{\theta,n} = F^{\theta}_{n\to 0}(x_{t_n}; o)$.
9: Recover $\Delta_0^{\theta,n} = a - \hat{x}_0^{\theta,n}$.
10: Evaluate the replay-conditioned likelihood using Eq. 6; for StructRL, instantiate p with Eq. 10.
11: Optimize $\mathcal{L}_{\text{RL}}$ using this replay-conditioned likelihood.
12: **end for**

C Full Empirical Verification of Structured Noise Dilution

This appendix expands the empirical verification in Sec. 3. Table 1 in the main paper reports one representative temporal-correlation setting and one representative action-group scale setting. Here we

Table C1: **Full per-condition data at base noise level $\sigma = 0.2$.** GR00T N1.5 SFT model on LIBERO-Long. **(a)** Lag-1 autocorrelation $\hat{\rho}_{\text{out}}$ under different injected ρ_{inj}. **(b)** Per-group standard deviation $\hat{\sigma}_{\text{out}}$ under different magnitude profiles (target $\sigma_{\text{inj}} = 0.2 \cdot R$). Action-space injection tracks the requested structure, while in-chain injection stays close to the SFT baseline.

Injected structure	**Action-space output**			**In-chain output**		
	pos	rot	grip	pos	rot	grip
(a) Lag-1 autocorrelation $\hat{\rho}_{out}$						
SFT baseline (ρ_{SFT}, no injection)	0.784	0.718	0.566	0.784	0.718	0.566
$\rho_{\text{inj}} = 0.00$ (no banding)	0.115	−0.033	0.225	0.785	0.718	0.564
$\rho_{\text{inj}} = 0.30$	0.310	0.203	0.378	0.787	0.721	0.562
$\rho_{\text{inj}} = 0.70$	0.556	0.493	0.560	0.788	0.725	0.577
$\rho_{\text{inj}} = 0.95$ (strong banding)	0.705	0.643	0.634	0.789	0.723	0.577
output range (max−min over $\rho_{\text{inj}} \in [0, 0.95]$)	0.590	0.676	0.409	0.004	0.007	0.015
(b) Per-group standard deviation $\hat{\sigma}_{out}$ (target $\sigma_{inj} = 0.2 \cdot R$)						
SFT baseline (σ_{SFT}, no injection)	0.105	0.077	0.206	0.105	0.077	0.206
$R_{\text{approach}} = (1.0, 0.5, 0.05)$ (pos-heavy)	0.227	0.128	0.192	0.109	0.078	0.193
$R_{\text{pregrasp}} = (0.05, 0.1, 1.0)$ (grip-heavy)	0.106	0.080	0.314	0.106	0.077	0.214
$R_{\text{carry}} = (0.5, 0.5, 0.1)$ (pos+rot)	0.147	0.127	0.205	0.106	0.077	0.190
output range (max−min across the three ratios)	0.121	0.048	0.122	0.003	0.001	0.024

Table C2: **Full per-condition data at base noise level $\sigma = 0.5$.** Same GR00T N1.5 SFT model on LIBERO-Long as Table C1, with a larger base noise level (target $\sigma_{\text{inj}} = 0.5 \cdot R$). The same dilution pattern appears: action-space injection follows the requested structure, while in-chain injection remains close to the SFT baseline.

Injected structure	**Action-space output**			**In-chain output**		
	pos	rot	grip	pos	rot	grip
(a) Lag-1 autocorrelation $\hat{\rho}_{out}$						
SFT baseline (ρ_{SFT}, no injection)	0.780	0.723	0.584	0.780	0.723	0.584
$\rho_{\text{inj}} = 0.00$ (no banding)	−0.038	−0.061	0.275	0.785	0.737	0.589
$\rho_{\text{inj}} = 0.30$	0.192	0.181	0.390	0.797	0.752	0.597
$\rho_{\text{inj}} = 0.70$	0.489	0.484	0.558	0.812	0.772	0.612
$\rho_{\text{inj}} = 0.95$ (strong banding)	0.644	0.635	0.609	0.803	0.759	0.617
output range (max−min over $\rho_{\text{inj}} \in [0, 0.95]$)	0.682	0.696	0.334	0.027	0.035	0.028
(b) Per-group standard deviation $\hat{\sigma}_{out}$ (target $\sigma_{inj} = 0.5 \cdot R$)						
SFT baseline (σ_{SFT}, no injection)	0.105	0.077	0.208	0.105	0.077	0.208
$R_{\text{approach}} = (1.0, 0.5, 0.05)$ (pos-heavy)	0.511	0.262	0.209	0.116	0.080	0.199
$R_{\text{pregrasp}} = (0.05, 0.1, 1.0)$ (grip-heavy)	0.109	0.093	0.572	0.107	0.077	0.249
$R_{\text{carry}} = (0.5, 0.5, 0.1)$ (pos+rot)	0.273	0.262	0.206	0.109	0.081	0.197
output range (max−min across the three ratios)	0.402	0.169	0.366	0.009	0.004	0.052

provide the full sweep at $\sigma = 0.2$ (Table C1) and an additional run at $\sigma = 0.5$ (Table C2). The larger noise level checks that the observed dilution is not an artifact of using a small perturbation.

Setup. We use a GR00T N1.5 SFT model on LIBERO-Long and inject structured perturbations Δ in two ways. For temporal structure, Δ is sampled with a prescribed AR(1) coefficient ρ_{inj} along the chunk axis. For action-group scale, Δ uses target magnitudes $\sigma_{\text{inj}} = \sigma \cdot R$ over the (position, rotation, gripper) groups, where σ is the base noise level and R is a ratio vector. We evaluate three ratios: $R_{\text{approach}} = (1.0, 0.5, 0.05)$ (position-heavy), $R_{\text{pregrasp}} = (0.05, 0.1, 1.0)$ (gripper-heavy), and $R_{\text{carry}} = (0.5, 0.5, 0.1)$ (position+rotation, low gripper). The perturbation is inserted either *in-chain*, before the remaining denoising steps, or in the *action space*, after denoising completes. We then measure the lag-1 autocorrelation $\hat{\rho}_{\text{out}}$ and per-group standard deviation $\hat{\sigma}_{\text{out}}$ of the executed action.[1] The SFT baseline rows report the unperturbed reverse flow.

Note on Monte-Carlo fluctuation. The SFT baseline rows should be the same across the $\sigma = 0.2$ and $\sigma = 0.5$ runs because they involve no injected perturbation. The small numerical differences in the

[1] All measurements average over 256 Monte-Carlo perturbation samples and 16 test-set observations.

tables (e.g., $\hat{\rho}_{\text{SFT,pos}} = 0.7840$ at $\sigma = 0.2$ vs. 0.7799 at $\sigma = 0.5$) come from independent Monte-Carlo estimates. With 256 perturbation samples and 16 observations, the standard error is larger than these small shifts, so they do not affect the conclusions.

Temporal correlation is diluted in-chain. Table C1(a) reports $\hat{\rho}_{\text{out}}$ as ρ_{inj} sweeps from 0 to 0.95. Under in-chain injection, the output correlation stays close to the SFT baseline $\rho_{\text{SFT}} \approx (0.78, 0.72, 0.57)$ for all injected values, changing by less than 0.02 in each group. Under action-space injection, the output correlation changes with the requested value; for example, position correlation increases from 0.11 to 0.70 as ρ_{inj} increases from 0 to 0.95. This shows that temporal structure can be specified directly in the action space, while the remaining denoising steps largely absorb the same structure when it is injected in-chain.

Per-group magnitude follows the same pattern. Table C1(b) repeats the comparison for group-wise magnitude profiles. Action-space injection follows the requested group scale: the position-heavy profile increases $\hat{\sigma}_{\text{pos}}$ from 0.105 to 0.227, while the gripper-heavy profile increases $\hat{\sigma}_{\text{grip}}$ from 0.206 to 0.314. In-chain injection changes the same statistics only slightly. The in-chain output is still anisotropic, but the anisotropy mostly reflects the SFT model rather than the injected profile.

Robustness across base noise levels. Table C2 repeats the protocol at $\sigma = 0.5$, which is $2.5\times$ larger than the setting in Table C1. The qualitative pattern remains the same. Action-space injection still tracks both the requested temporal correlation and the requested group scale. In-chain injection changes the executed action statistics only mildly; for example, the position-heavy profile increases $\hat{\sigma}_{\text{pos}}$ from 0.105 to 0.116, far below the action-space value of 0.511. This confirms that structured noise dilution is not caused only by a small base noise level.

D Log-Probability of Structured Action-Space Noise

This appendix derives Eq. 10. The sampling process first draws an AR(1) noise sequence for each action dimension d:

$$\varepsilon_{1,d} \sim \mathcal{N}(0,1), \quad \varepsilon_{c+1,d} \;=\; \rho_{g(d)}\,\varepsilon_{c,d} \;+\; \sqrt{1-\rho_{g(d)}^2}\,\zeta_{c,d}, \quad \zeta_{c,d} \sim \mathcal{N}(0,1), \quad c = 1, \ldots, C-1. \tag{24}$$

The action-space perturbation is obtained by applying the learned diagonal scale:

$$\Delta_{0,c,d} = \sigma_{c,d}\varepsilon_{c,d}. \tag{25}$$

The density of the AR(1) sequence factorizes as

$$\log p(\varepsilon) = \sum_{d=1}^{D}\left[\log \mathcal{N}(\varepsilon_{1,d}; 0, 1) + \sum_{c=1}^{C-1} \log \mathcal{N}(\varepsilon_{c+1,d}; \rho_{g(d)}\varepsilon_{c,d}, 1-\rho_{g(d)}^2)\right]. \tag{26}$$

During terminal replay, the stored action and replayed clean action define the perturbation $\Delta_0^{\theta,n}$. Under the current state-conditioned scale, its corresponding AR(1) coordinate is

$$\varepsilon_{c,d}^{\theta,n} = \frac{\Delta_{0,c,d}^{\theta,n}}{\sigma_{c,d}}. \tag{27}$$

The transformation from ε to Δ_0 is diagonal, with Jacobian determinant $\prod_{c,d}\sigma_{c,d}$. The change of variables gives

$$\log p_\phi\left(\Delta_0^{\theta,n} \mid h_\theta\right) = \log p(\varepsilon^{\theta,n}) - \sum_{c,d}\log\sigma_{c,d}, \tag{28}$$

which yields Eq. 10. Sampling still follows the forward construction $\varepsilon \to \Delta_0$ in Eq. 9.

E Regularizer for the Structured Noise Module

This appendix details the regularizer for the structured noise module in Sec. 4.3. The state-conditioned scale σ controls the magnitude of action-space exploration. To keep it in a useful range during RL, we use three lightweight penalties:

$$\mathcal{L}_{\text{reg}}(\sigma) \;=\; \lambda_{\text{ent}}\big(\log D - \mathcal{H}(\bar{\sigma})\big) \;+\; \lambda_{\text{b}}\big(\|\sigma\|_F - \sigma^{\text{base}}\sqrt{CD}\big)^2 \;+\; \lambda_{\text{s}}\sum_{c=1}^{C-1}\|\sigma_{c+1} - \sigma_c\|^2, \tag{29}$$

where $\bar{\sigma}_{c,d} = \sigma_{c,d} / \sum_{d'} \sigma_{c,d'}$ and $\mathcal{H}(\bar{\sigma}) = -\sum_d \bar{\sigma}_{c,d} \log \bar{\sigma}_{c,d}$, averaged over batch and chunk positions. The dimension-entropy term discourages scale collapse onto a few dimensions, the magnitude-anchor term keeps the total exploration scale near the target level, and the smoothness term avoids abrupt scale changes along the chunk.

F Real-World Experimental Details

This appendix provides additional details for the real-world experiments in Sec. 5.4.

Scene setup. Experiments are conducted on a Franka Research 3 (FR3) robot with a ROS 2 Cartesian-impedance control stack. The policy observes two 256×256 RGB views and a 19-dimensional proprioceptive state containing the relative TCP pose and velocity, gripper position, force, and torque. In *pick-banana*, the robot must grasp a banana and place it in a basket. The policy uses $C = 10$ and a 7-dimensional action at each step, consisting of 3D translation, a 3D rotation vector, and gripper control. In *plug-charger-in*, the connector is held by the gripper and must be aligned with and inserted into a socket. The policy uses $C = 5$ and a 6-dimensional end-effector action, while the gripper is held fixed. Both tasks use delta end-effector pose actions in the current end-effector frame and execute primitive commands at 10 Hz.

Algorithm. We instantiate StructRL with $\pi_{0.5}$ and an asynchronous off-policy AWAC learner. The implementation reuses replay-buffer and target-network infrastructure from a SAC worker, but optimizes the AWAC objective rather than a maximum-entropy SAC objective. As in Sec. 5.4, we set the ODE denoising horizon to $T = 1$. The rollout policy first predicts a clean action chunk and then applies StructRL exploration noise in normalized action space. For *plug-charger-in*, we use fixed AR(1) parameters $\sigma = 0.1$ and $\rho = 0.8$ for both translation and rotation. Because $T = 1$, the stored latent is the policy-independent initial Gaussian noise x_1. The current policy can therefore recompute the clean action from the same x_1 and evaluate previously executed policy or human-intervention actions under its current action-space distribution.

Every full action chunk forms one transition $\mathcal{T}_t$ as defined in Eq. 11. Executing a_t^b from o_t advances the robot by C primitive steps and produces the next observation o_{t+C}. All transitions enter the online buffer. Transitions containing SpaceMouse corrections and transitions from the ten fastest successful episodes are additionally retained in the demo buffer; when both buffers are available, each minibatch contains equal numbers of online and demo transitions. Human-intervention actions are mapped to the same normalized model space as policy actions before forming a_t^b. If an intervention begins within a chunk, the remaining policy tail is discarded and replaced by continued human input or a zero-delta hold, avoiding actions predicted for a counterfactual state sequence.

The critic consumes the cached 2048-dimensional pooled PaliGemma feature, the raw 19-dimensional proprioceptive state, and the full executed action chunk a_t^b. A shared state trunk feeds one value branch and two action-conditioned Q branches, while a GRU encodes the temporal order within the action chunk. Let R_t and d_t denote the chunk return and terminal indicator computed from $\mathbf{r}_t$ and $\mathbf{d}_t$, respectively. The target for each Q_i is

$$y_t = R_t + (1 - d_t)\gamma^C V'(o_{t+C}). \tag{30}$$

The target value of the replay action is

$$q_t = \min_{i \in \{1,2\}} Q_i'(o_t, a_t^b). \tag{31}$$

Here, primes denote target networks. The twin Q functions are trained by squared regression to y_t, and V is fit to the 0.7 expectile of q_t. The replay advantage is

$$A_t = q_t - V(o_t). \tag{32}$$

The corresponding AWAC weight is

$$w_t = \exp\left(\mathrm{clip}\left(\frac{A_t}{\beta}, -3, 3\right)\right), \qquad \beta = 0.5. \tag{33}$$

Finally, the actor is optimized using the same objective as in Eqs. 12–13:

$$\mathcal{L}_{\mathrm{AWAC}}(\theta) = -\mathbb{E}_{\mathcal{T}_t \sim \mathcal{D}}\left[w_t \log \pi_\theta(a_t^b \mid o_t, x_1)\right]. \tag{34}$$

Successful policy actions and useful human corrections therefore receive larger likelihood updates. Rollout runs on one GPU, while four learner GPUs update the critic and action expert asynchronously. Cached VLM features allow multiple critic updates per physical transition without repeatedly evaluating the visual-language encoder.

Training setup. Both tasks use sparse success feedback, with no dense distance shaping or learned reward model. Each task starts from a task-specific $\pi_{0.5}$ checkpoint obtained by SFT on ten pre-collected demonstrations. The resulting policies exhibit only a coarse task-directed motion trend and achieve 0% success over 25 independent trials before online RL. The critic is warmed up on the initially collected replay before actor updates begin. Physical interaction, replay ingestion, critic learning, actor learning, and policy synchronization then proceed asynchronously.

Control and safety. Normalized delta actions are first clipped to the environment bounds, transformed from the current end-effector frame to the robot base frame, scaled to metric delta poses, and integrated from the latest measured pose. The resulting absolute equilibrium pose is then clipped to a task-specific workspace and orientation window before being sent to the Cartesian-impedance controller. Additional safeguards include SpaceMouse override, removal of an interrupted chunk's stale policy tail, controller target clipping, and the robot's collision thresholds. For *plug-charger-in*, success is detected when every Cartesian position coordinate is within 5 mm of the target; the configured orientation threshold is not used by the current success checker. Safety-stop and timeout transitions remain in the online buffer as zero-reward evidence, but enter the demo buffer only when they contain a human intervention.

G Long-Horizon Chaining on CALVIN

This appendix presents the CALVIN long-horizon evaluation referenced in Sec. 5. CALVIN evaluates long-horizon behavior by chaining up to five language instructions in one episode. This setting stresses the temporal axis of StructRL because jitter or unstable exploration in an early subtask can affect later subtasks. Table G1 reports per-length success on $\pi_{0.5}$. StructRL achieves the highest average sequence length and is the only method above 0.93 at Len4. The gap over the time-independent in-chain baselines becomes larger as the sequence length increases, which is consistent with the role of action-space temporal correlation.

Table G1: **Long-horizon chaining on CALVIN.** Per-length success rate (fraction in $[0, 1]$) and average sequence length on $\pi_{0.5}$. StructRL achieves the highest average and maintains the strongest performance at longer sequence lengths.

Model	Len1	Len2	Len3	Len4	Len5	Avg.
$\boldsymbol{\pi_{0.5}}$ – SFT	0.927	0.843	0.767	0.688	0.613	3.838
+ π_{RL} (Flow-SDE+PPO)	0.997	0.982	0.958	0.910	0.870	4.717
+ π_{RL} (Flow-Noise+PPO)	0.996	0.976	0.939	0.896	0.845	4.652
+ Baseline	0.998	0.990	0.964	0.929	0.868	4.749
+ StructRL	**1.000**	**0.993**	**0.968**	**0.934**	**0.880**	**4.775**

H Training Hyperparameters

Table H1: **Training hyperparameters for LIBERO** across the three VLA backbones and four task suites.

Parameters	LIBERO											
	GR00T N1.5				π_0				$\pi_{0.5}$			
	Spatial	**Object**	**Goal**	**Long**	**Spatial**	**Object**	**Goal**	**Long**	**Spatial**	**Object**	**Goal**	**Long**
Train epochs	200	200	200	200	200	200	400	400	400	400	400	400
Batch size	1024	1024	1024	1024	2048	2048	2048	2048	2048	2048	2048	2048
Update epochs	1	1	1	1	1	1	4	4	1	1	4	4
Actor lr	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	5e-6	5e-6	5e-6	5e-6	5e-6	5e-6
Value lr	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4
Noise lr	2e-5	2e-5	2e-5	2e-5	2e-5	2e-5	1e-4	1e-4	1e-4	1e-4	1e-4	1e-4
Interaction steps	240	240	320	480	240	240	320	480	240	240	320	480
Parallel environments	64	64	64	64	64	64	64	64	64	64	64	64
Rollout epochs	8	8	8	8	8	8	8	8	8	8	8	8
Action chunk C	5	5	5	10	5	5	5	10	5	5	5	10
Denoise steps	4	4	4	4	4	4	4	4	3	5	5	5
Noise level σ	0.2	0.2	0.2	0.2	0.2	0.1	0.1	0.1	0.1	0.05	0.05	0.05
Structured noise	AR(1) ρ (pos, rot, grip): $(0.8, 0.6, 0.7)$ Clip scale (pos/rot/grip): $(0.1, 2.0)/(0.1, 1.5)/(0.1, 3.0)$											

Table H2: **Training hyperparameters for CALVIN and ManiSkill multitask settings.**

Parameters	CALVIN	ManiSkill	
	$\pi_{0.5}$	π_0	$\pi_{0.5}$
Train epochs	300	400	400
Batch size	2048	5120	5120
Update epochs	4	4	5
Actor lr	5e-6	2e-5	2e-5
Value lr	1e-4	2e-5	2e-5
Noise lr	1e-4	2e-5	2e-5
Interaction steps	480	80	80
Parallel environments	64	320	320
Rollout epochs	8	1	1
Action chunk H	5	5	5
Denoise steps	5	4	4
Noise level σ	0.2	0.2	0.2
Structured noise	AR(1) ρ: $(0.8, 0.6, 0.7)$ Clip scale s: $(0.1, 2.0)/(0.1, 1.5)/(0.1, 3.0)$		