EDIT ONLY WHERE IT FAILS: BOUNDARY-GUIDED EDITING VIA LATENT TRANSFORMATIONS FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language-action models provide strong priors for robot manipulation, but their performance can degrade under deployment-specific differences in object geometry, viewpoint, contact conditions, and robot embodiment. Existing adaptation methods often modify the policy globally, even when deployment failures are concentrated in a limited set of states. This creates a tension between adaptation and preservation: sufficiently strong updates may correct new failures but also disturb useful pretrained behaviors. We present **BELT-VLA**, a boundary-guided policy editing framework for online reinforcement learning of vision-language-action models. BELT-VLA identifies failure-prone regions through human interventions and applies state-conditioned orthogonal transformations to the action-flow representation only when an edit is required. The edit policy is optimized with an objective that balances exploration against intervention risk, while the orthogonal editing mechanism limits unnecessary policy deviation. We evaluate the method on four real-world manipulation tasks involving precise insertion, assembly, packing, and liquid transfer. The proposed approach increases the average success rate from 33.5% to 82.0%, while reducing human intervention and limiting policy changes outside failure-critical states. Comparisons with parameter-efficient fine-tuning and unconstrained adaptation show that boundary-guided orthogonal editing provides a favorable balance between online policy improvement and prior preservation.

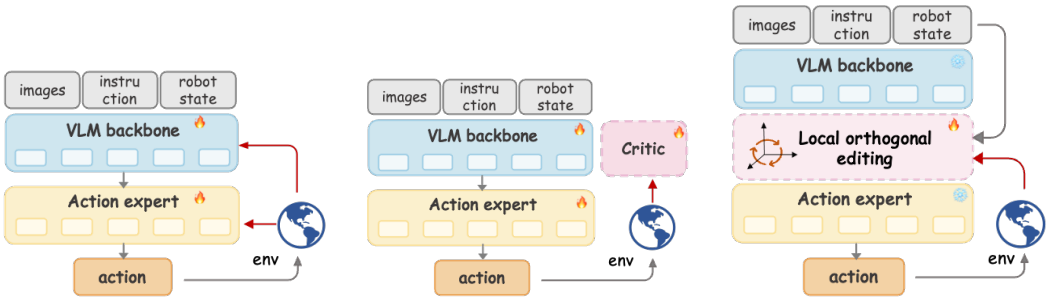
1 INTRODUCTION

Vision-language-action (VLA) models have emerged as a promising foundation for general-purpose robot control (Ma et al., 2026; Zhang et al., 2025a; Guan et al., 2025). By jointly modeling visual observations, language instructions, robot states, and continuous actions, pretrained VLA policies can transfer semantic and behavioral knowledge across tasks, scenes, and embodiments. Recent systems have demonstrated increasingly broad manipulation capabilities, including long-horizon task execution, multi-object interaction, and cross-environment generalization (Kim et al., 2024; Black et al., 2024; Intelligence et al., 2025; Bjorck et al., 2025; Liu et al., 2025). However, strong pretrained performance does not eliminate the need for deployment-time adaptation (Kim et al., 2025; Zhang et al., 2026; Guo et al., 2026a).

During deployment, a pretrained policy may remain reliable over much of a manipulation trajectory yet fail in narrow, contact-sensitive phases such as alignment, insertion, assembly, or transfer (Kim et al., 2025; Zhang et al., 2026; Guo et al., 2026a; Zhang et al., 2025b). In such cases, the desired correction is localized in state space rather than uniformly required throughout the policy. This motivates adaptation mechanisms that concentrate policy changes around failure-prone states while limiting unnecessary deviation elsewhere (Sagar et al., 2025; Xu et al., 2025).

Most existing adaptation methods do not explicitly exploit this locality. Full fine-tuning modifies the pretrained weights, whereas parameter-efficient methods such as LoRA introduce trainable low-rank updates into the model (Kim et al., 2025; 2026; Jiang et al., 2026), as shown in Fig. 1(a). Although both approaches improve downstream performance, their globally shared updates can let a few failure states affect broader inputs. When errors occur only in narrow behavioral phases, such updates

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.



(a) Full/LoRA tuning (b) Critic-based correction (c) Our method

Figure 1: RL adaptation mechanisms for VLA models. (a) Parameter-space tuning modifies shared parameters and may alter behaviors beyond failure regions. (b) Critic-based tuning uses an external critic module to guide adaptation but lacks geometric constraints on latent updates. (c) BELT-VLA performs state-conditioned orthogonal editing of the action-flow representation while keeping the VLM backbone and action expert frozen.

may disrupt reliable behaviors, causing task overfitting and weakening pretrained priors (Guo et al., 2026a; Jiang et al., 2026; Kim et al., 2026). Another line of work uses reinforcement learning with reward models, progress estimators, or other critic-based objectives to refine VLA action generation during post-training (Lu et al., 2025; Tan et al., 2025; Li et al., 2025; Chen et al., 2025a; Zhang et al., 2025b), as illustrated in Fig. 1(b). Compared with directly updating the pretrained model, these methods provide a more modular interface for policy correction. However, they often lack explicit modeling of local failure boundaries, making it difficult to determine when the original policy should remain unchanged. Moreover, unconstrained updates to latent action representations may distort the geometry inherited from pretraining (Qiu et al., 2023; Liu et al., 2024).

Therefore, deployment adaptation must determine when and where to apply the edit, and how to constrain it. Together, these requirements create a fundamental stability–plasticity trade-off: stronger updates can correct new failures but also increase the risk of overwriting previously reliable behaviors.

This paper argues that online VLA adaptation should be formulated as minimal policy editing rather than global policy optimization. The objective is not to replace the pretrained policy, but to use deployment-time human interventions to identify failure-prone regions, localize behavioral corrections around them, and suppress unnecessary policy drift elsewhere. This perspective raises three questions: First, how can online adaptation target local failures without disrupting reliable behaviors? Second, how can sparse human takeovers define continuous failure boundaries and guide exploration and optimization? Third, with the pretrained VLA frozen, how can an editing interface correct actions while limiting policy drift?

We propose BELT-VLA, **B**oundary-Guided **E**diting via **L**atent **T**ransformations for Vision-Language-Action Models. Human interventions supervise a Failure Critic that maps rollout states to continuous intervention risk, characterizing local failure boundaries of the deployed policy. The resulting risk estimates guide both online exploration and the localization of policy edits.

As shown in Fig. 1(c), BELT-VLA keeps the VLM backbone and flow-matching action expert frozen and performs adaptation only in the latent action-flow space. A state-conditioned bottleneck provides a compact representation for the edit policy, whose steering outputs parameterize geometry-preserving orthogonal transformations. During online adaptation, Random Network Distillation (RND) novelty and intervention risk provide the learning signal for PPO, while boundary-conditioned regularization suppresses unnecessary edits in low-risk states (Burda et al., 2018; Schulman et al., 2017).

Our main contributions are as follows:

- A formulation of online VLA adaptation as failure-localized policy editing, which concentrates corrections on failure-related states while preserving reliable behavior and pretrained priors elsewhere.

- A human-intervention-guided, boundary-aware online learning method. A Failure Critic converts pre-takeover states into continuous intervention-risk estimates to guide risk-aware exploration and policy optimization.
- A constrained latent policy editing mechanism. With the VLA policy frozen, BELT-VLA applies state-conditioned orthogonal transformations to action-flow representations through a compact editing interface that constrains representational distortion.
- A real-world evaluation across four manipulation tasks. We validate BELT-VLA in terms of task success rate, human intervention rate, and the locality of policy edits.

2 Related Work

2.1 RL-based Adaptation of VLA Models

Vision-language-action (VLA) models provide reusable policy priors across diverse manipulation tasks, objects, and robot embodiments (Kim et al., 2024; Intelligence et al., 2025; Bjorck et al., 2025; Wu et al., 2026; Guo et al., 2026b). Deployment-specific changes in embodiment, object geometry, viewpoint, and contact conditions can still produce systematic failures, motivating adaptation of pretrained VLA policies to the target environment.

Existing VLA adaptation methods primarily rely on additional demonstrations and update the policy through full fine-tuning, low-rank adaptation, lightweight adapters, or trainable policy heads (Liu et al., 2025; Wang et al., 2026; Kim et al., 2026). While effective for downstream adaptation, these methods typically apply shared policy updates without explicitly accounting for the locality of deployment failures (Chen et al., 2025b). Reinforcement learning further enables VLA adaptation through interaction (Li et al., 2025; Zhang et al., 2025c; Chen et al., 2025a; Lu et al., 2025; Zang et al., 2025). In particular, RIPT-VLA demonstrates that pretrained VLA policies can be substantially improved through interactive post-training with sparse task-level rewards (Tan et al., 2025). These approaches establish the effectiveness of RL for VLA post-training, but their optimization is primarily driven by overall task performance rather than by identifying which states require policy correction. BELT-VLA instead uses human interventions to identify failure-prone regions and concentrates policy edits near states that require correction.

2.2 Human Intervention and Failure-Aware Robot Learning

Real-world robot learning increasingly uses online interaction to refine manipulation policies after deployment (Luo et al., 2024; Chen et al., 2025a; Zhao et al., 2025; Yang et al., 2026). These studies show that robot policies can be refined from real-world experience, but practical deployment still faces high interaction costs, risky exploration, costly resets, and difficult reward specification. Human guidance addresses these challenges through demonstrations, corrective actions, takeovers, and task-level feedback. HIL-SERL combines demonstrations, a learned binary reward classifier, and online interventions for real-world RL (Luo et al., 2024), while DexHiL uses intervention-aware sampling to prioritize corrective segments for VLA post-training (Han et al., 2026). Related approaches incorporate human corrections as supervised data or convert feedback into binary labels, preferences, or learned rewards (Luo et al., 2025; Yang et al., 2025).

Intervention timing further reveals where an autonomous policy begins to become unreliable. States preceding a takeover provide samples near failure-prone regions, whereas human-controlled recovery segments provide corrective behavior (Hu et al., 2025). BELT-VLA uses the former to estimate a continuous failure boundary rather than treating intervention primarily as task feedback or corrective training data. This boundary determines where online policy editing should be concentrated.

2.3 Constrained Latent Policy Editing

Recent robot-learning methods increasingly adapt pretrained policies through constrained latent interfaces. LAE detects undesirable states and selectively edits activations of a frozen policy, while ZPRL freezes a flow-based base policy and learns RL residuals within a compact latent bottleneck (Das et al., 2025; Yu et al., 2026). VLA-Pilot performs inference-time steering without fine-tuning the pretrained VLA, whereas PriorVLA preserves a frozen prior expert while learning a

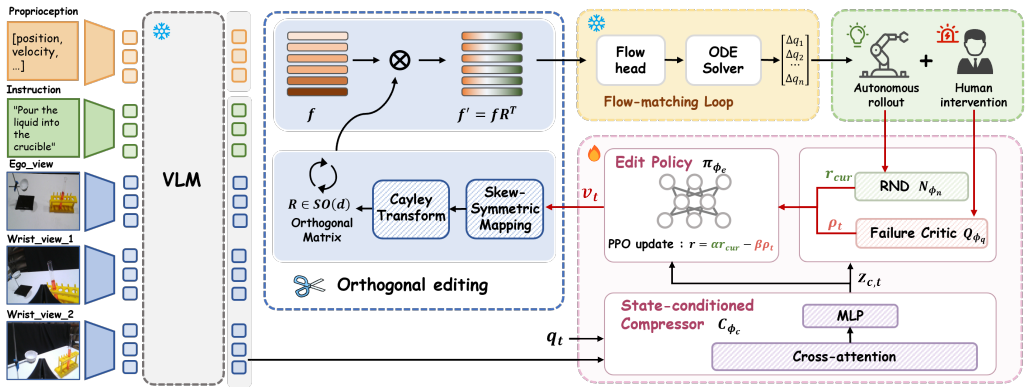


Figure 2: Overview of **BELT-VLA**. A state-conditioned compressor summarizes frozen VLM features for an edit policy, which applies orthogonal transformations to action-flow representations before flow-matching decoding. The edit policy is optimized with PPO using RND novelty and intervention-supervised failure risk. Black arrows denote forward computation, while red arrows indicate edit-control and online-learning signals.

separate adaptation pathway (Li et al., 2026; Guo et al., 2026a). Related model-editing methods constrain updates through sparse localization, orthogonal transformations, null-space projection, or activation steering (Qiu et al., 2023; Liu et al., 2024; Yang et al., 2024; Fang et al., 2025; Lee et al., 2025; Lai et al., 2025).

These works establish selective activation editing, latent RL steering, inference-time policy correction, and prior-preserving adaptation. Unlike these approaches, BELT-VLA uses an intervention-defined failure boundary to regulate the magnitude of latent editing and suppress unnecessary deviations in reliable regions.

3 Method

3.1 Overview

We consider deployment-time adaptation of a flow-matching VLA policy, instantiated with $\pi_{0.5}$ (Intelligence et al., 2025). BELT-VLA keeps the pretrained VLA fixed and introduces a lightweight latent editing module for adaptation. At each policy step t, the multimodal observations are processed by the frozen VLA to produce visual-language representations z_t and latent action-flow features f_t for final continuous action generation.

As shown in Fig. 2, the state-conditioned compressor $\mathcal{C}_{\phi_c}$ aggregates z_t conditioned on proprioceptive state s_t to obtain a compact edit representation $z_{c,t}$. The edit policy π_{ϕ_e} then produces a steering vector v_t, from which an orthogonal transformation R_t is constructed and applied to the latent action-flow features before the frozen flow head. In parallel, a Failure Critic Q_{ϕ_q}, trained from human intervention data, predicts an intervention-risk score $\rho_t = Q_{\phi_q}(z_{c,t}) \in [0, 1]$. The predicted risk identifies regions where the deployed base policy becomes unreliable and provides the boundary signal for localized editing.

BELT-VLA disentangles edit localization from edit realization. During online adaptation, a risk-aware edit policy determines *where* policy modifications are required, while a state-conditioned orthogonal transformation specifies *how* the latent action-flow features are modified. To encourage minimal policy editing during online adaptation, we introduce the following editing regularization:

$$\mathcal{L}_{\text{edit}} = \mathbb{E}_t \left[(1 - \text{sg}[\rho_t]) \, \mathbb{E}_{v_t \sim \pi_{\phi_e}(\cdot|z_{c,t})} \left[\|v_t\|_2^2 \right] \right]. \tag{1}$$

The complete actor objective is

$$\mathcal{L}_{\text{BELT}} = \mathcal{L}_{\text{PPO}} + \lambda_{\text{edit}} \mathcal{L}_{\text{edit}}, \tag{2}$$

where $\mathcal{L}_{\text{PPO}}$ is defined as the negative clipped PPO surrogate objective, and the overall objective is minimized. Here, $\text{sg}[\cdot]$ denotes stop-gradient, and λ_{edit} controls the strength of boundary-

conditioned edit regularization. The surrogate objective is evaluated using the steering action stored during rollout, whereas $\mathcal{L}_{\text{edit}}$ is evaluated under the current edit policy.

3.2 Boundary-Guided Latent Editing

State-conditioned edit representation. Localized policy editing requires identifying when the policy becomes unreliable, but simple pooling of VLM tokens lacks the robot-state conditioning needed to distinguish failure-prone configurations. We therefore construct the edit representation through state-conditioned token aggregation. The proprioceptive state is first encoded into the VLM feature space to form a state-dependent query:

$$q_t = \mathrm{MLP}_s(s_t). \tag{3}$$

This query provides the conditioning signal for determining which parts of the visual-language representation are most relevant under the current robot configuration. Rather than compressing z_t beforehand, the query attends directly to the complete token sequence:

$$\widetilde{z}_t = \mathrm{CrossAttn}\left(Q = q_t,\, K = z_t,\, V = z_t\right), \tag{4}$$

This state-conditioned attention preserves token-level task information while adapting the aggregation to the current robot configuration. The attended representation is then mapped into a low-dimensional bottleneck to retain task-relevant information while reducing the complexity of online adaptation:

$$z_{c,t} = \mathrm{MLP}_c(\widetilde{z}_t) = \mathcal{C}_{\phi_c}(z_t, s_t), \qquad z_{c,t} \in \mathbb{R}^{d_c}. \tag{5}$$

The resulting $z_{c,t}$ provides a compact shared state representation for both failure-boundary estimation and latent policy editing.

Orthogonal action-flow editing. BELT-VLA adopts the Cayley parameterization from Orthogonal Fine-Tuning (OFT) to construct state-conditioned orthogonal edits in the latent action-flow space. Conditioned on $z_{c,t}$, the edit policy samples a low-dimensional steering vector:

$$v_t \sim \pi_{\phi_e}(\cdot \mid z_{c,t}). \tag{6}$$

The sampled vector is mapped to a skew-symmetric matrix:

$$A_t = \Psi(v_t), \qquad A_t^\top = -A_t, \tag{7}$$

where $\Psi(\cdot)$ maps the low-dimensional steering vector to a block-wise skew-symmetric matrix through a lightweight projection and grouped parameterization, enabling the edit policy to control high-dimensional latent features through a compact interface. Details of the projection and grouping configuration are provided in Appendix. Applying the Cayley transform to A_t yields an orthogonal matrix:

$$R_t = (I - A_t)(I + A_t)^{-1}, \qquad R_t^\top R_t = I. \tag{8}$$

A single R_t is generated at each policy inference step and reused across all internal flow-matching steps of the corresponding action chunk. Since $v_t = 0$ yields $A_t = 0$ and $R_t = I$, penalizing $\|v_t\|_2$ encourages near-identity transformations and preserves the original VLA pathway. Let $f_t \in \mathbb{R}^{L_f \times d_f}$ denote the latent action-flow feature matrix of the current action chunk, where L_f is the number of suffix positions and d_f is the feature dimension. BELT-VLA applies the orthogonal transformation along the feature dimension:

$$f_t' = f_t R_t^\top. \tag{9}$$

Because R_t is orthogonal, the transformation preserves the Frobenius norm of the feature matrix. The transformation therefore rotates the latent feature space without introducing arbitrary feature scaling. The frozen output projection maps the edited features to the vector field used by the flow-matching sampler to generate the action chunk.

Intervention-supervised failure boundary. BELT-VLA uses human interventions to identify regions where the base policy becomes unreliable. During online rollout, an operator takes over when execution becomes unsafe or unlikely to recover autonomously. The K autonomous states preceding each intervention are labeled as risky, while states outside all pre-intervention windows are labeled as safe. The value of K and other intervention-labeling settings are provided in the Appendix.

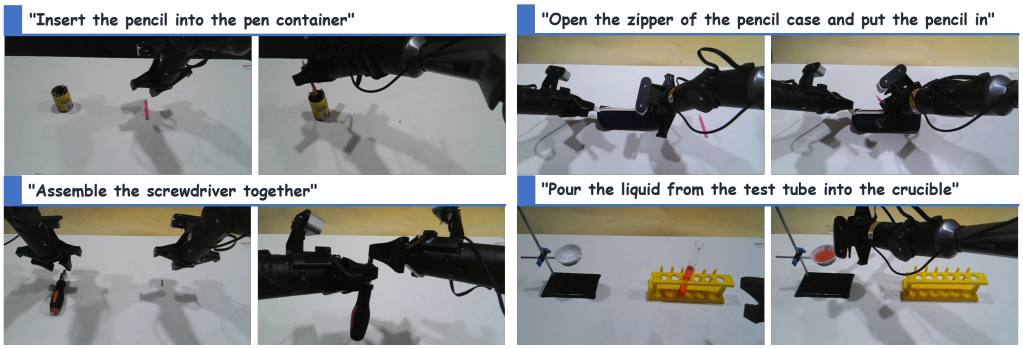


Figure 3: Overview of the real-world tasks, spanning long-horizon execution and fine-grained, contact-rich manipulation.

We collect the labeled rollout tuples in

$$\mathcal{D}_{\text{risk}} = \{(z_{c,t}, s_t, y_t)\}, \qquad y_t \in \{0, 1\}, \tag{10}$$

where $y_t = 1$ denotes a pre-intervention state and $y_t = 0$ denotes an intervention-free state. The Failure Critic Q_{ϕ_q} is optimized with binary cross-entropy:

$$\mathcal{L}_{\text{FC}} = \mathbb{E}_{\mathcal{D}_{\text{risk}}} \left[\text{BCE} \left(Q_{\phi_q}(z_{c,t}), y_t \right) \right]. \tag{11}$$

The Failure Critic produces the state-dependent risk estimate

$$\rho_t = Q_{\phi_q}(z_{c,t}) \in [0, 1], \tag{12}$$

where a larger ρ_t indicates a higher likelihood of human intervention. This continuous risk estimate characterizes the transition from reliable to intervention-prone states and is used both to penalize risky exploration and to modulate the strength of edit regularization. Details of the initialization procedure are provided in the Appendix.

3.3 Online Optimization with Human Guidance

Novelty–risk learning signal. During online adaptation under human supervision, terminal task success signals are sparse and distributionally skewed because intervention truncates many trajectories before their natural outcome. BELT-VLA therefore starts from the frozen task policy and uses novelty and intervention risk to shape online exploration. Novelty favors less familiar states, whereas risk discourages transitions toward intervention-prone regions.

Specifically, a fixed target network g_{tar} and a trainable predictor N_{ϕ_n} map $z_{c,t}$ to the same embedding space. The novelty reward is defined as

$$r_t^{\text{cur}} = \| N_{\phi_n}(z_{c,t}) - g_{\text{tar}}(z_{c,t}) \|_2^2 . \tag{13}$$

A larger prediction error indicates that the current state is less familiar to the predictor and is therefore assigned a higher novelty reward. The target network remains fixed, while the predictor is updated independently by minimizing its prediction error on collected states.

The online reward combines novelty and intervention risk:

$$r_t = \alpha r_t^{\text{cur}} - \beta \rho_t, \tag{14}$$

where α and β balance exploration and intervention avoidance. Reward normalization, coefficient settings, and other details are provided in the Appendix.

Initialization and boundary-regularized online optimization. Before online interaction, BELT-VLA is initialized from task demonstrations while the pretrained VLM backbone and flow-matching action expert remain frozen. The state-conditioned compressor $\mathcal{C}_{\phi_c}$ and edit policy π_{ϕ_e} are trained on demonstrated action chunks through the flow-matching objective, while the RND predictor N_{ϕ_n} is fitted to the compact state representations extracted from the same demonstrations.

Table 1: Task success rate (%) across four real-world manipulation tasks. Results are computed over 50 autonomous trials per task.

Method	Pencil insertion	Screwdriver assembly	Pencil case packing	Chemical pouring	Avg.
$\pi_{0.5}$+LoRA	52.0	16.0	12.0	48.0	32.0
GR00T N1.6	32.0	8.0	10.0	24.0	18.5
BELT-VLA (offline init.)	54.0	14.0	14.0	52.0	33.5
HIL-SERL	92.0	64.0	72.0	82.0	77.5
RL-Token	90.0	58.0	70.0	86.0	76.0
BELT-VLA	**92.0**	**72.0**	**74.0**	**90.0**	**82.0**

During online adaptation, the compressor is fixed. We use $z_{c,t}$ as the PPO state and the steering vector v_t as the actor action. PPO updates only the edit policy π_{ϕ_e} under $\mathcal{L}_{\mathrm{BELT}}$, while a value network is optimized for value estimation and generalized advantage estimation (GAE). The RND predictor N_{ϕ_n} and Failure Critic Q_{ϕ_q} are optimized independently under their respective objectives. The complete PPO objective, actor distribution, initialization procedure, and optimization hyperparameters are provided in the Appendix.

4 Experiments

We structure our experimental evaluation around the following questions: (i) Can BELT-VLA improve real-world manipulation performance through localized policy editing? (ii) Are policy edits concentrated near intervention-defined failure boundaries while remaining limited in reliable states? (iii) How do the main learning and editing components affect task success and human intervention during online adaptation?

4.1 Experimental Setup

Platform and Tasks. Our real-world experiments are conducted on an Agilex ALOHA dual-arm manipulation platform. The robot is equipped with three synchronized RGB cameras, including one head-mounted camera and two wrist-mounted cameras, together with proprioceptive joint-state observations.

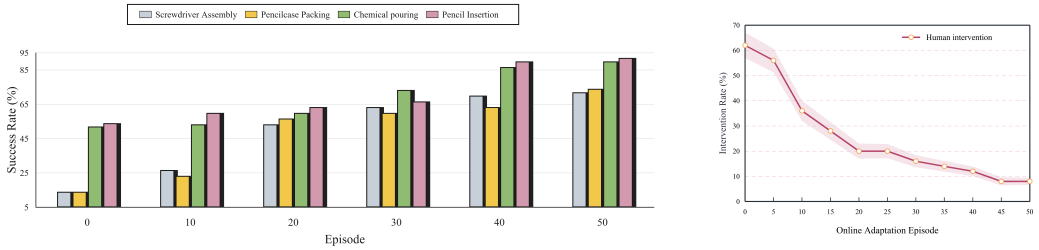
We evaluate BELT-VLA on four challenging tasks: *pencil holder insertion*, *screwdriver bit assembly*, *pencil case packing*, and *chemical liquid transfer*, as illustrated in Fig. 3. Detailed task specifications are provided in Appendix.

Evaluation Protocol and Metrics. For each task, all methods are post-trained using the same 100 task-specific demonstrations. Online baselines use matched interaction budgets and intervention protocols. Full baseline and evaluation settings are provided in the Appendix. For locality analysis, we additionally collect intervention rollouts with the offline policy before online adaptation. These trajectories follow the same intervention protocol but are excluded from training and used only for intervention-aligned evaluation.

We report Success Rate (**SR**) over 50 autonomous trials and Human Intervention Rate (**HIR**) during online adaptation. Policy-edit locality is measured by intervention-aligned latent edit magnitude and action deviation as defined in locality analysis.

4.2 Real-World Online Adaptation

We compare BELT-VLA with representative offline and online adaptation baselines. The offline baselines include LoRA adaptation of $\pi_{0.5}$ and fine-tuning of GR00T N1.6 (Bjorck et al., 2025), while RL-Token (Yang et al., 2026) and HIL-SERL (Luo et al., 2024) serve as real-world online RL baselines.



(a) Autonomous success rate

(b) Human intervention rate

Figure 4: Online adaptation with BELT-VLA. (a) Autonomous success rates across all four tasks show a clear upward trend as online adaptation proceeds. (b) The mean HIR across the four tasks decreases during online adaptation. The shaded region indicates the 95% confidence interval over task-level HIRs.

Table 1 summarizes the final performance on the four real-world tasks. The fine-tuned $\pi_{0.5}$ policy outperforms fine-tuned GR00T N1.6, while its flow-matching architecture provides a continuous latent interface for policy adaptation, motivating our action-flow editing approach. BELT-VLA (offline) achieves an average success rate of **33.5%** while keeping the pretrained policy fixed, comparable to the fine-tuned $\pi_{0.5}$. After online adaptation, BELT-VLA reaches **82.0%** and outperforms RL-Token and HIL-SERL under the same interaction budget.

Figure 4(a) shows how task performance evolves during online adaptation. The gains vary with task difficulty: pencil insertion begins with the strongest offline performance, whereas screwdriver assembly shows the largest improvement and places greater demands on pose accuracy and contact control. Overall, BELT-VLA consistently improves the offline policy through real-world interaction. Figure 4(b) reports the human intervention rate, which decreases from **60%** to **8%** over training. Together with the increase in task success, this reduction suggests that fewer interventions do not come at the expense of task completion.

4.3 Locality of Boundary-Guided Policy Editing

We evaluate whether policy edits concentrate near states where the initialized policy becomes unreliable. Held-out pre-adaptation rollouts are aligned at the onset of human takeover. Pre-intervention states are treated as failure-related, while reliable states are sampled from successful autonomous execution or segments far from intervention, with task progress matched across groups.

We quantify the effect of latent editing at both the representation and action levels. The normalized latent edit magnitude is

$$E_t = \frac{\|f_t R_t^\top - f_t\|_F}{\|f_t\|_F}, \tag{15}$$

where f_t is the latent action-flow representation before transformation. To quantify behavioral deviation, we generate two action chunks for each held-out observation using the initialized policy before online adaptation and the online-adapted policy, with the same initial flow noise. Let $a_{t,k}$ denote the k-th action generated by the initialized policy and $\tilde{a}_{t,k}$ the corresponding action generated by the online-adapted policy. We define

$$D_t = \frac{1}{H} \sum_{k=1}^{H} \|\tilde{a}_{t,k} - a_{t,k}\|_2 , \tag{16}$$

where D_t denotes the action deviation between the initialized and online-adapted policies, and H is the action-chunk horizon. Action dimensions are normalized using demonstration statistics, and D_t is averaged over five shared noise seeds.

Fig. 5 aligns both quantities with human takeover. During autonomous execution, latent edit magnitude and action deviation remain small at states distant from intervention and increase as the trajectory approaches the failure boundary. Values after $t = 0$ are evaluated offline on observations from the recovery segment and do not correspond to robot-executed actions. A group-level

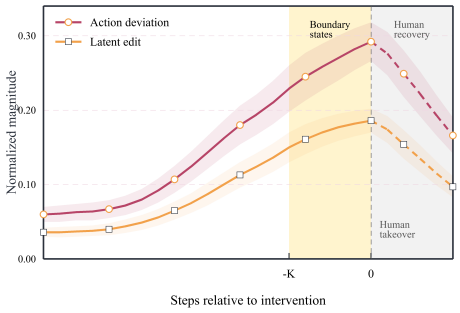


Figure 5: Policy editing around human intervention boundaries. Curves are aligned to human takeover at $t = 0$; shaded regions denote 95% confidence intervals.

Table 2: Ablation of BELT-VLA. Results are averaged across the four real-world tasks.

Method	SR (%) ↑	HIR (%) ↓
BELT-VLA	**82.0**	**8.0**
w/o Novelty	70.5	15.0
w/o Risk Penalty	66.0	28.0
w/o Boundary Reg.	72.0	21.0
Unconstrained Editing	68.5	24.0

analysis in Appendix further shows a boundary-to-reliable action deviation ratio higher than those of the boundary-unregularized and unconstrained editing variants. These results show that BELT-VLA concentrates larger policy changes around intervention-defined failure regions while limiting deviations in states where the offline policy remains reliable.

4.4 Ablation Studies

We ablate four components of BELT-VLA: **w/o Novelty** removes the RND reward, while **w/o Risk Penalty** removes the intervention-risk term from the online objective. **w/o Boundary Regularization** disables the risk-conditioned edit penalty but retains the same online reward. **Unconstrained Editing** replaces the orthogonal transformation with a capacity-matched unconstrained latent transformation.

As shown in Table 2, removing novelty reduces the success rate from 82.0% to 70.5%. This suggests that novelty supports exploration beyond states covered by the demonstrations. Removing the risk penalty produces the highest intervention rate (28%) and the lowest success rate (66.0%), indicating the contribution of intervention risk to online adaptation. Boundary regularization increases SR and reduces HIR; the locality analysis further shows that it suppresses policy deviation away from failure-prone states. Finally, unconstrained editing substantially degrades success and increases intervention, indicating that the orthogonality constraint is beneficial in our setting.

5 Conclusion and Future Work

We presented BELT-VLA, a boundary-guided framework for localized online adaptation of frozen VLA policies. Human interventions identify where the base policy becomes unreliable, while boundary-conditioned regularization suppresses unnecessary edits in reliable states. A state-conditioned orthogonal editor determines how action-flow representations are modified, providing a constrained latent interface for policy correction. Across real-world manipulation tasks, BELT-VLA improves task success, reduces human intervention, and concentrates editing-induced deviations near failure-prone states. These results show that targeted latent editing provides an effective alternative to global adaptation when deployment failures are localized and the base policy remains reliable elsewhere.

BELT-VLA still relies on human takeovers for boundary supervision, and its intervention-risk estimates may shift as the deployed policy or task changes. In addition, the current editor is tailored to the latent action space of flow-matching VLAs. Future work will explore autonomous boundary supervision, transferable risk modeling, and risk-triggered human assistance, where the Failure Critic actively requests intervention before the policy enters a failure-prone region, as well as more general latent editing interfaces.

AI USE STATEMENT

In this work, generative AI tools were used to aid language editing and improve the clarity and readability of the manuscript. They were also used for literature retrieval and discovery of potentially relevant related work. All AI-assisted content was reviewed and revised by the authors, and all references identified with AI assistance were independently verified against the original sources. We take full responsibility for the final content of this work.

ETHICS STATEMENT

This work uses human intervention during robot learning experiments to prevent unsafe or undesirable robot behaviors and to provide supervision for policy adaptation. No sensitive personal information is collected or analyzed. All experiments were conducted in controlled laboratory environments with appropriate safety precautions. We are not aware of additional ethical concerns associated with this work.

REPRODUCIBILITY STATEMENT

The Appendix provides detailed implementation settings, task specifications, baseline configurations, and evaluation protocols. Supplementary materials include the source code used to train and evaluate BELT-VLA.

REFERENCES

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 et al. A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

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

Yuhui Chen, Shuai Tian, Shugao Liu, Yingting Zhou, Haoran Li, and Dongbin Zhao. Conrft: A reinforced fine-tuning method for vla models via consistency policy. *arXiv preprint arXiv:2502.05450*, 2025b.

Satyajeet Das, Darren Chiu, Zhehui Huang, Lars Lindemann, and Gaurav S. Sukhatme. Latent activation editing: Inference-time refinement of learned policies for safer multirobot navigation. *arXiv preprint arXiv:2509.20623*, 2025.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained knowledge editing for language models. In *International Conference on Learning Representations*, 2025.

Weifan Guan, Qinghao Hu, Aosheng Li, and Jian Cheng. Efficient vision-language-action models for embodied manipulation: A systematic survey. *arXiv preprint arXiv:2510.17111*, 2025.

Xinyu Guo, Bin Xie, Wei Chai, Xianchi Deng, Tiancai Wang, Zhengxing Wu, and Xingyu Chen. Priorvla: Prior-preserving adaptation for vision-language-action models. *arXiv preprint arXiv:2605.10925*, 2026a.

Yanjiang Guo, Tony Lee, Lucy Xiaoyang Shi, Jianyu Chen, Percy Liang, and Chelsea Finn. Vlaw: Iterative co-improvement of vision-language-action policy and world model. *arXiv preprint arXiv:2602.12063*, 2026b.

Yifan Han, Zhongxi Chen, Yuxuan Zhao, Congsheng Xu, Yanming Shao, Yichuan Peng, Yao Mu, and Wenzhao Lian. Dexhil: A human-in-the-loop framework for vision-language-action model post-training in dexterous manipulation. *arXiv preprint arXiv:2603.09121*, 2026.

Zheyuan Hu, Robyn Wu, Naveen Enock, Jasmine Li, Riya Kadakia, Zackory Erickson, and Aviral Kumar. Rac: Robot learning for long-horizon tasks by scaling recovery and correction, 2025. URL https://arxiv.org/abs/2509.07953.

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

Yuhua Jiang, Junjie Lu, Xinyao Qin, Xiaoyu Chen, Kaixin Wang, Feifei Gao, and Li Zhao. Vla-gse: Boosting parameter-efficient fine-tuning in vla with generalized and specialized experts. *arXiv preprint arXiv:2605.06175*, 2026.

Donghoon Kim, Minji Bae, Unghui Nam, Gyeonghun Kim, Suyun Lee, Kyuhong Shim, and Byonghyo Shim. Adaptive capacity allocation for vision language action fine-tuning. *arXiv preprint arXiv:2603.07404*, 2026.

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

Moo Jin Kim et al. Fine-tuning vision-language-action models. *arXiv preprint arXiv:2502.19645*, 2025.

Wen Lai, Alexander Fraser, and Ivan Titov. Joint localization and activation editing for low-resource fine-tuning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 32206–32227. PMLR, 2025.

Bruce W. Lee, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Erik Miehling, Pierre Dognin, Manish Nagireddy, and Amit Dhurandhar. Programming refusal with conditional activation steering. In *International Conference on Learning Representations*, 2025.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, et al. Simplevla-rl: Scaling vla training via reinforcement learning. *arXiv preprint arXiv:2509.09674*, 2025.

Zhuo Li, Junjia Liu, Zhipeng Dong, Tao Teng, Quentin Rouxel, Darwin Caldwell, and Fei Chen. Towards deploying vla without fine-tuning: Plug-and-play inference-time vla policy steering via embodied evolutionary diffusion. *IEEE Robotics and Automation Letters*, 11(5):6234–6241, 2026. doi: 10.1109/LRA.2026.3678455.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations*, volume 2025, pp. 29982–30009, 2025.

Weiyang Liu, Zeju Qiu, Yao Feng, Yuliang Xiu, Yuxuan Xue, Longhui Yu, Haiwen Feng, Zhen Liu, Juyeon Heo, Songyou Peng, Yandong Wen, Michael J. Black, Adrian Weller, and Bernhard Schölkopf. Parameter-efficient orthogonal finetuning via butterfly factorization. In *International Conference on Learning Representations*, 2024.

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. Vla-rl: Towards masterful and general robotic manipulation with scalable reinforcement learning. *arXiv preprint arXiv:2505.18719*, 2025.

Jianlan Luo, Zheyuan Hu, Charles Xu, You Liang Tan, Jacob Berg, Archit Sharma, Stefan Schaal, Chelsea Finn, Abhishek Gupta, and Sergey Levine. Serl: A software suite for sample-efficient robotic reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16961–16969. IEEE, 2024.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision–language–action models for embodied ai. *IEEE Transactions on Neural Networks and Learning Systems*, 2026.

Zeju Qiu, Weiyang Liu, Haiwen Feng, Yuxuan Xue, Yao Feng, Zhen Liu, Dan Zhang, Adrian Weller, and Bernhard Schölkopf. Controlling text-to-image diffusion by orthogonal finetuning. *arXiv preprint arXiv:2306.07280*, 2023.

Som Sagar, Jiafei Duan, Sreevishakh Vasudevan, Yifan Zhou, Heni Ben Amor, Dieter Fox, and Ransalu Senanayake. From mystery to mastery: Failure diagnosis for improving manipulation policies. In *Second Workshop on Out-of-Distribution Generalization in Robotics at RSS 2025*, 2025.

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

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, et al. Vla-adapter: An effective paradigm for tiny-scale vision-language-action model. In *Proceedings of the AAAI conference on artificial intelligence*, volume 40, pp. 18638–18646, 2026.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

Feng Xu, Guangyao Zhai, Xin Kong, Tingzhong Fu, Daniel FN Gordon, Xueli An, and Benjamin Busam. Stare-vla: Progressive stage-aware reinforcement for fine-tuning vision-language-action models. *arXiv preprint arXiv:2512.05107*, 2025.

Yi Yang, Huaihang Zheng, Kai Ma, Tian Xie, Guozheng Li, Shenglin Xu, Xiangyu Wang, Yiren Ma, and Baoxu Liu. openpi-rlt: Real-robot rlt reproduction on openpi, 2026. Open-source real-robot reproduction of RL Token (RLT) built on openpi.

Yichu Yang, Hongtao Wu, Xiao Ma, Tao Kong, Di Hu, et al. Robotic policy learning via human-assisted action preference optimization. *arXiv e-prints*, pp. arXiv–2506, 2025.

Yunqiao Yang, Long-Kai Huang, Shengzhuang Chen, Kede Ma, and Ying Wei. Learning where to edit vision transformers. In *Advances in Neural Information Processing Systems*, volume 37, pp. 134459–134487, 2024.

Dongjie Yu, Kun Lei, Zhennan Jiang, Jia Pan, and Huazhe Xu. Beyond action residuals: Real-world robot policy steering via bottleneck latent reinforcement learning. *arXiv preprint arXiv:2605.19919*, 2026.

Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Liangzhi Shi, Yuqing Xie, Zhexuan Xu, et al. Rlinf-vla: A unified and efficient framework for vla+ rl training. *arXiv preprint arXiv:2510.06710*, 2025.

Dapeng Zhang, Jing Sun, Chenghui Hu, Xiaoyan Wu, Zhenlong Yuan, Rui Zhou, Fei Shen, and Qingguo Zhou. Pure vision language action (vla) models: A comprehensive survey. *arXiv preprint arXiv:2509.19012*, 2025a.

Hongyin Zhang, Shuo Zhang, Junxi Jin, Qixin Zeng, Runze Li, and Donglin Wang. Robustvla: Robustness-aware reinforcement post-training for vision-language-action models. *arXiv preprint arXiv:2511.01331*, 2025b.

Hongyin Zhang, Zifeng Zhuang, Han Zhao, Pengxiang Ding, Hongchao Lu, and Donglin Wang. Reinbot: Amplifying robot visual-language manipulation with reinforcement learning. *arXiv preprint arXiv:2505.07395*, 2025c.

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

Yinuo Zhao, Huiqian Jin, Lechun Jiang, Xinyi Zhang, Kun Wu, Pei Ren, Zhiyuan Xu, Zhengping Che, Lei Sun, Dapeng Wu, et al. Real-world reinforcement learning from suboptimal interventions. *arXiv preprint arXiv:2512.24288*, 2025.

A Appendix

A.1 Task and Platform Specifications

We evaluate BELT-VLA on four real-world bimanual manipulation tasks performed on the AgileX ALOHA platform, as shown in Fig.6. These tasks require coordinated dual-arm control and involve long-horizon sequential interactions.

Task descriptions.

- Pencil Insertion. The robot must grasp a pencil from the tabletop, align it with the opening of a pencil holder, and insert it successfully. The task requires accurate grasping, pose alignment, and contact-rich insertion.
- Pencil Case Packing. The robot must unzip a pencil case, pick up a pen from the workspace, place it into the opened case, and maintain the case opening throughout the manipulation sequence. This task involves long-horizon coordination and object transportation.
- Screwdriver Assembly. The robot is required to remove the existing screwdriver head and attach a replacement head. Successful completion requires precise object alignment and stable bimanual coordination during the insertion process.
- Chemical Pouring. The robot must grasp a test tube containing colored liquid and pour the liquid into a crucible without excessive spilling. The task requires stable grasping, continuous motion control, and careful manipulation under contact and fluid-transfer constraints.

A.2 Network Architectures and Editing Location

This section summarizes the trainable components of BELT-VLA and the location at which latent editing is applied. The pretrained VLM backbone, flow-matching action expert, and output projection remain frozen throughout training. The state-conditioned compressor $\mathcal{C}_{\phi_c}$ is trained only during offline initialization and then fixed during online adaptation. Given an observation at policy step t, the frozen VLM backbone produces the token representation z_t. The state-conditioned compressor maps the visual-language tokens and proprioceptive state to a compact representation,

$$z_{c,t} = \mathcal{C}_{\phi_c}(z_t, s_t). \tag{17}$$

The edit policy is a lightweight MLP that maps the compact state representation to the mean steering output,

$$\mu_t = \mu_{\phi_e}(z_{c,t}), \qquad \mu_t \in \mathbb{R}^{d_v}. \tag{18}$$

During offline initialization, the deterministic steering vector is set to $\bar{v}_t = \mu_t$. During online adaptation, μ_t parameterizes the diagonal Gaussian PPO actor described in Appendix A.5. The steering vector parameterizes the grouped orthogonal transformation applied to the latent action-flow representation immediately before the frozen output projection. For each policy step, the same transformation R_t is reused across all flow-matching steps used to generate the corresponding action chunk. The grouped Cayley parameterization is detailed in Appendix A.6.

The architectures of the lightweight BELT-VLA modules are summarized in Table 3.

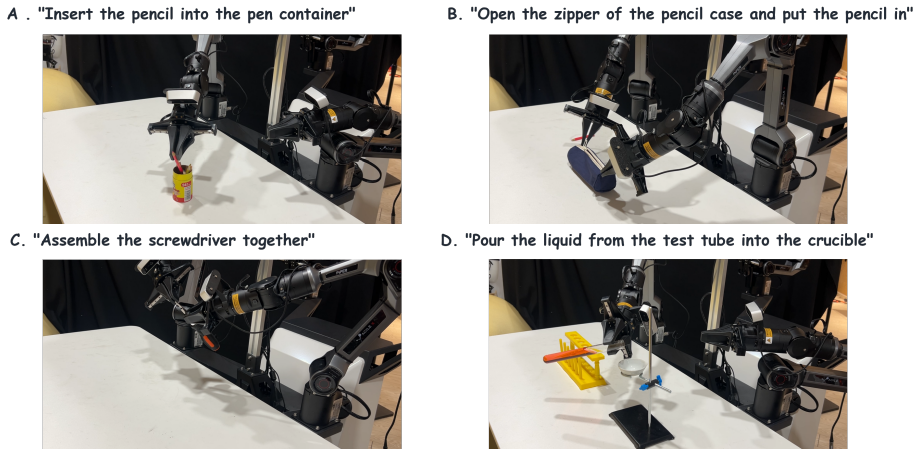


Figure 6: Four real-world bimanual manipulation tasks.

Table 3: Architectures of the lightweight BELT-VLA modules.

Module	**Architecture**	**Output**
State-conditioned compressor	State projection $d_s \to d_z$; one-query, 8-head cross-attention over z_t; MLP $d_z \to 512 \to 256$ with ReLU	$z_{c,t}$
Edit policy	MLP $256 \to 256 \to 64$ with ReLU and $\tanh$	μ_t
PPO value network	MLP $256 \to 256 \to 1$ with ReLU	$V_\psi(z_{c,t})$
Failure Critic	MLP $256 \to 256 \to 1$ with ReLU and sigmoid	$\rho_t \in [0,1]$
RND target	MLP $256 \to 256 \to 256$ with ReLU; fixed after random initialization	$g_{\text{tar}}(z_{c,t})$
RND predictor	MLP $256 \to 256 \to 256$ with ReLU	$N_{\phi_n}(z_{c,t})$

A.3 Offline Initialization

Demonstration collection. For each task, we collect 100 teleoperated demonstrations using the same robot platform, observations, proprioceptive inputs, and action representation as in autonomous execution. Each demonstration covers the complete task from a randomized initial state to successful completion. Let

$$\mathcal{D}_{\text{demo}} = \{(o_t, s_t, a_t^*)\} \tag{19}$$

denote the resulting observation-state-action trajectories, where o_t is the visual-language observation, s_t is the proprioceptive state, and a_t^* is the demonstrated action chunk. The dataset contains no task rewards, intervention labels, or failure labels, and is used only to initialize the state-conditioned compressor, edit policy, and RND predictor.

Edit policy initialization. The state-conditioned compressor $\mathcal{C}_{\phi_c}$ and the actor mean network μ_{ϕ_e} are jointly initialized through the frozen flow-matching action expert. For each demonstration sample, the deterministic steering vector is $\bar{v}_t = \mu_{\phi_e}(z_{c,t})$. The initialization objective is

$$\mathcal{L}_{\text{init}} = \mathcal{L}_{\text{FM}}^{\text{demo}} + \lambda_0 \mathbb{E}_{\mathcal{D}_{\text{demo}}} \left[\|\mu_{\phi_e}(z_{c,t})\|_2^2 \right], \tag{20}$$

where $\mathcal{L}_{\text{FM}}^{\text{demo}}$ is the original flow-matching objective evaluated on the demonstrated action chunks. Only ϕ_c and ϕ_e are updated by this objective; the pretrained VLM, action expert, and output projection remain frozen. Because the steering-to-generator projection is bias-free, zero steering produces $A_t = 0$ and therefore $R_t = I$. The second term thus biases the initialized editor toward the identity transformation on demonstration-covered states.

RND initialization. The RND target network g_{tar} is randomly initialized and kept fixed. The predictor N_{ϕ_n} is pretrained on the compact representations extracted from the demonstration trajec-

tories by minimizing the RND prediction error:

$$\mathcal{L}_{\mathrm{RND}}^{\mathrm{demo}} = \mathbb{E}_{\mathcal{D}_{\mathrm{demo}}} \left[\| N_{\phi_n}(z_{c,t}) - g_{\mathrm{tar}}(z_{c,t}) \|_2^2 \right]. \tag{21}$$

Gradients from $\mathcal{L}_{\mathrm{RND}}^{\mathrm{demo}}$ update only ϕ_n and are not propagated into the compressor. This pretraining makes demonstration-covered states initially familiar to the RND module. The Failure Critic Q_{ϕ_q} is not trained on $\mathcal{D}_{\mathrm{demo}}$ because the demonstrations contain neither intervention events nor pre-intervention labels. The PPO value network V_ψ is likewise not trained during offline initialization because no online novelty–risk returns are available. After offline initialization, ϕ_c is frozen, while ϕ_e remains trainable during online adaptation.

A.4 Online Interaction and Intervention Labeling

During online adaptation, the robot executes a frozen copy of the current edit policy while a human operator monitors the rollout. The operator takes control when continued autonomous execution is unsafe or unlikely to recover without assistance, including imminent collision, excessive force, object damage, liquid spilling, loss of grasp, severe misalignment, or prolonged stagnation. Minor pose errors are not treated as interventions when autonomous recovery remains plausible.

Let τ denote the first human-controlled policy step. The preceding K autonomous states within the same rollout segment form the pre-intervention risk window

$$\mathcal{W}_\tau = \{\tau - K, \ldots, \tau - 1\}. \tag{22}$$

Let $\mathcal{I}$ denote the set of intervention steps. For an autonomous state indexed by i, the intervention label is

$$y_i = \begin{cases} 1, & \text{if } i \in \bigcup_{\tau \in \mathcal{I}} \mathcal{W}_\tau, \\ 0, & \text{if } i \text{ is autonomous and outside all risk windows.} \end{cases} \tag{23}$$

The resulting risk replay buffer is

$$\mathcal{D}_{\mathrm{risk}} = \{(z_{c,i}, y_i)\}, \qquad y_i \in \{0, 1\}. \tag{24}$$

When the operator takes control, the current autonomous rollout segment is terminated. Human-controlled recovery transitions are excluded from the PPO rollout buffer, risk supervision, and GAE computation. Once the operator returns control, autonomous data collection resumes as a new rollout segment. Thus, policy gradients and value targets do not propagate through human-controlled recovery trajectories. The resulting autonomous rollout and risk replay buffers are used for the online updates described in Appendix A.5.

A.5 Reward Construction and PPO Optimization

PPO actor. During online adaptation, the compact representation $z_{c,t}$ is treated as the PPO state, and the steering vector $v_t \in \mathbb{R}^{d_v}$ is treated as the continuous actor action. Using the mean output μ_t defined in Eq. 18, the stochastic edit policy is

$$\pi_{\phi_e}(v_t \mid z_{c,t}) = \mathcal{N}\left(v_t; \mu_t, \mathrm{diag}(\sigma^2)\right), \tag{25}$$

where $\log \sigma \in \mathbb{R}^{d_v}$ is a trainable state-independent parameter. During rollout collection, the steering vector is sampled as

$$v_t = \mu_t + \sigma \odot \varepsilon_t, \qquad \varepsilon_t \sim \mathcal{N}(0, I). \tag{26}$$

During deterministic evaluation, the actor mean $v_t = \mu_t$ is used.

At the beginning of each rollout batch, the current actor is copied and frozen as the behavior policy $\pi_{\phi_e^{\mathrm{old}}}$. Rollouts are collected using $\pi_{\phi_e^{\mathrm{old}}}$. For each autonomous transition, the compact state, sampled steering vector, behavior-policy log probability, continuation mask, and successor state are stored in the autonomous rollout buffer $\mathcal{B}_{\mathrm{aut}}$. The novelty–risk reward is computed after rollout collection and added before PPO optimization.

Novelty–risk reward. The RND prediction error defines the novelty signal

$$r_t^{\mathrm{nov}} = \|N_{\phi_n}(z_{c,t}) - g_{\mathrm{tar}}(z_{c,t})\|_2^2, \tag{27}$$

where g_{tar} is the fixed target network and N_{ϕ_n} is the predictor. The Failure Critic estimates intervention risk as

$$\rho_t = Q_{\phi_q}(z_{c,t}), \qquad \rho_t \in [0,1]. \tag{28}$$

The online reward is

$$r_t = \alpha r_t^{\mathrm{nov}} - \beta \rho_t, \tag{29}$$

where α and β control the contributions of novelty and intervention risk, respectively. Novelty values are computed before the RND predictor is updated on the newly collected states. The resulting rewards are stored in $\mathcal{B}_{\mathrm{aut}}$ and remain fixed during the corresponding PPO updates.

Clipped PPO objective. The probability ratio is computed from the current and behavior-policy log probabilities,

$$r_t^{\mathrm{PPO}}(\phi_e) = \exp\left[\log \pi_{\phi_e}(v_t \mid z_{c,t}) - \log \pi_{\phi_e^{\mathrm{old}}}(v_t \mid z_{c,t})\right]. \tag{30}$$

The clipped actor loss is

$$\mathcal{L}_{\mathrm{PPO}} = -\mathbb{E}_t\left[\min\Big(r_t^{\mathrm{PPO}}(\phi_e)\widehat{A}_t, \mathrm{clip}\big(r_t^{\mathrm{PPO}}(\phi_e), 1-\epsilon_{\mathrm{clip}}, 1+\epsilon_{\mathrm{clip}}\big)\,\widehat{A}_t\Big)\right], \tag{31}$$

where $\epsilon_{\mathrm{clip}} > 0$ is the PPO clipping parameter and $\widehat{A}_t$ is the estimated advantage.

Value estimation. A separate value network V_ψ estimates the return induced by the novelty–risk reward. The Failure Critic Q_{ϕ_q} is not the PPO value network. For each autonomous transition, the temporal-difference residual is

$$\delta_t = r_t + \gamma m_t V_\psi(z_{c,t+1}) - V_\psi(z_{c,t}), \tag{32}$$

where $\gamma \in [0,1)$ is the discount factor and m_t is the continuation mask,

$$m_t = \begin{cases} 0, & \text{if the autonomous segment terminates after transition } t, \\ 1, & \text{otherwise.} \end{cases} \tag{33}$$

Human-controlled recovery transitions are excluded from $\mathcal{B}_{\mathrm{aut}}$, $\mathcal{D}_{\mathrm{risk}}$, and GAE computation. Advantages are computed using generalized advantage estimation,

$$\widehat{A}_t = \delta_t + \gamma\lambda_{\mathrm{GAE}} m_t \widehat{A}_{t+1}, \tag{34}$$

where $\lambda_{\mathrm{GAE}} \in [0,1]$ is the GAE parameter.

The value target is

$$\widehat{V}_t^{\mathrm{targ}} = \widehat{A}_t + V_\psi(z_{c,t}), \tag{35}$$

and the value loss is

$$\mathcal{L}_V = \mathbb{E}_t\left[\left(V_\psi(z_{c,t}) - \mathrm{sg}\left[\widehat{V}_t^{\mathrm{targ}}\right]\right)^2\right]. \tag{36}$$

Risk-conditioned edit regularization. BELT-VLA suppresses unnecessary editing in states predicted to be reliable through

$$\mathcal{L}_{\mathrm{edit}} = \mathbb{E}_t\left[(1 - \mathrm{sg}[\rho_t])\,\mathbb{E}_{\widetilde{v}_t \sim \pi_{\phi_e}(\cdot \mid z_{c,t})}\left[\|\widetilde{v}_t\|_2^2\right]\right]. \tag{37}$$

The inner expectation is evaluated under the current actor rather than the behavior policy. For the diagonal Gaussian actor in Eq. 25,

$$\mathbb{E}_{\widetilde{v}_t \sim \pi_{\phi_e}(\cdot \mid z_{c,t})}\left[\|\widetilde{v}_t\|_2^2\right] = \|\mu_{\phi_e}(z_{c,t})\|_2^2 + \|\sigma\|_2^2. \tag{38}$$

The penalty is strongest in low-risk states and is relaxed as the estimated intervention risk increases.

The complete edit-policy loss is

$$\mathcal{L}_{\mathrm{BELT}} = \mathcal{L}_{\mathrm{PPO}} + \lambda_{\mathrm{edit}}\mathcal{L}_{\mathrm{edit}}. \tag{39}$$

Only the edit-policy parameters ϕ_e are optimized using $\mathcal{L}_{\mathrm{BELT}}$. The value-network parameters ψ are optimized separately using $\mathcal{L}_V$.

Algorithm 1 Online Optimization of BELT-VLA

Require: Autonomous rollout buffer $\mathcal{B}_{\text{aut}}$ and risk replay buffer $\mathcal{D}_{\text{risk}}$
Require: Coefficients α, β, and λ_{edit}
Update ϕ_q on $\mathcal{D}_{\text{risk}}$ using $\mathcal{L}_{\text{FC}}$
for each transition t in $\mathcal{B}_{\text{aut}}$ **do**
  Compute $r_t^{\text{nov}} \leftarrow \|N_{\phi_n}(z_{c,t}) - g_{\text{tar}}(z_{c,t})\|_2^2$
  Compute $\rho_t \leftarrow Q_{\phi_q}(z_{c,t})$
  Compute $r_t \leftarrow \alpha r_t^{\text{nov}} - \beta \rho_t$
  Store r_t and $\text{sg}[\rho_t]$ in $\mathcal{B}_{\text{aut}}$
end for
Compute advantages $\widehat{A}_t$ and value targets $\widehat{V}_t^{\text{targ}}$ using segment-wise GAE
for each PPO optimization epoch **do**
  for each minibatch sampled from $\mathcal{B}_{\text{aut}}$ **do**
    Compute $\mathcal{L}_{\text{PPO}}$ using the stored steering vectors and behavior-policy log probabilities
    Compute $\mathcal{L}_{\text{edit}}$ under the current actor using $\text{sg}[\rho_t]$
    Update ϕ_e using $\mathcal{L}_{\text{BELT}}$
    Compute $\mathcal{L}_V$ using $\widehat{V}_t^{\text{targ}}$
    Update ψ using $\mathcal{L}_V$
  end for
end for
Compute $\mathcal{L}_{\text{RND}}^{\text{online}}$ using the states in $\mathcal{B}_{\text{aut}}$
Update only ϕ_n using $\mathcal{L}_{\text{RND}}^{\text{online}}$

Auxiliary online updates. The Failure Critic is trained from the intervention-labeled replay buffer $\mathcal{D}_{\text{risk}}$ using

$$\mathcal{L}_{\text{FC}} = \mathbb{E}_{(z_{c,i},y_i)\sim\mathcal{D}_{\text{risk}}}\left[\text{BCE}\big(Q_{\phi_q}(z_{c,i}), y_i\big)\right]. \tag{40}$$

The Failure Critic is optimized independently of PPO. Gradients from $\mathcal{L}_{\text{PPO}}$, $\mathcal{L}_{\text{edit}}$, and $\mathcal{L}_V$ are not propagated into Q_{ϕ_q}.

After the novelty rewards for a rollout batch have been stored, the RND predictor is updated on autonomous states by minimizing

$$\mathcal{L}_{\text{RND}}^{\text{online}} = \mathbb{E}_{z_{c,t}\sim\mathcal{B}_{\text{aut}}}\left[\|N_{\phi_n}(z_{c,t}) - g_{\text{tar}}(z_{c,t})\|_2^2\right]. \tag{41}$$

The target network g_{tar} remains fixed throughout online adaptation. The complete online optimization procedure is summarized in Algorithm 1.

The main training and online adaptation hyperparameters are summarized in Table 4.

A.6 Grouped Orthogonal Editing

This section provides the implementation details of the grouped orthogonal editor used in BELT-VLA. Grouping reduces the dimensionality and computational cost of constructing the orthogonal transformation while preserving the state-conditioned Cayley parameterization described in the main paper.

Steering projection and grouped parameterization. At policy step t, the edit policy produces a steering vector

$$v_t \in \mathbb{R}^{d_v}, \tag{42}$$

where d_v denotes the steering dimension. The latent action-flow representation has feature dimension d_f. Directly parameterizing a dense skew-symmetric matrix in $\mathbb{R}^{d_f\times d_f}$ requires

$$M_{\text{dense}} = \frac{d_f(d_f-1)}{2} \tag{43}$$

independent coefficients. To reduce this cost, we partition the feature dimension into G non-overlapping groups of equal width d_g, such that

$$d_f = G d_g. \tag{44}$$

Table 4: Training and online adaptation hyperparameters.

Category	Hyperparameter	Value
Offline initialization	Demonstrations per task	100
Offline initialization	Steering regularization λ_0	0.1
Risk labeling	Pre-intervention window K	3 policy steps
Risk labeling	Risk replay buffer capacity	2048 transitions
Online reward	Novelty weight α	0.05
Online reward	Risk weight β	0.5
PPO actor	Initial log standard deviation $\log \sigma$	-1.0
PPO actor	Actor learning rate	3×10^{-4}
PPO value	Value-network learning rate	3×10^{-4}
PPO	Maximum rollout horizon	30 policy steps
PPO	Clipping parameter ϵ_{clip}	0.2
PPO	Discount factor γ	0.99
PPO	GAE parameter λ_{GAE}	0.95
PPO	Transitions per PPO update	256
PPO	Minibatch size	64
PPO	Edit regularization λ_{edit}	1.0
PPO	Optimization epochs per update	4
Failure Critic	Learning rate	3×10^{-4}
Failure Critic	Minibatch size	64
RND	Predictor learning rate	1×10^{-4}
RND	Minibatch size	64
Optimization	Optimizer	Adam

Each group requires

$$m_g = \frac{d_g(d_g - 1)}{2} \tag{45}$$

independent coefficients to parameterize a $d_g \times d_g$ skew-symmetric matrix. The total number of grouped coefficients is therefore

$$M = G m_g = G \frac{d_g(d_g - 1)}{2}. \tag{46}$$

Rather than requiring the edit policy to directly predict all M coefficients, the compact steering vector is mapped to the grouped skew-symmetric parameter space through a learned linear projection:

$$\theta_t = W_p v_t, \qquad W_p \in \mathbb{R}^{M \times d_v}, \qquad \theta_t \in \mathbb{R}^M. \tag{47}$$

We use a bias-free projection so that

$$v_t = 0 \quad \Longrightarrow \quad \theta_t = 0. \tag{48}$$

This property ensures that a zero steering vector exactly corresponds to an identity latent edit.

The projected coefficients are partitioned into G groups:

$$\theta_t = \left[\theta_t^{(1)}; \ldots; \theta_t^{(G)}\right], \qquad \theta_t^{(g)} \in \mathbb{R}^{m_g}. \tag{49}$$

Grouped skew-symmetric construction. For each group g, the coefficient vector $\theta_t^{(g)}$ parameterizes a skew-symmetric matrix

$$A_t^{(g)} \in \mathbb{R}^{d_g \times d_g}. \tag{50}$$

Let $\kappa(i, j)$ denote a fixed one-to-one indexing of the strictly upper-triangular entries for $1 \le i < j \le d_g$. The entries of $A_t^{(g)}$ are defined as

$$\left[A_t^{(g)}\right]_{ij} = \begin{cases} \left[\theta_t^{(g)}\right]_{\kappa(i,j)}, & i < j, \\ 0, & i = j, \\ -\left[\theta_t^{(g)}\right]_{\kappa(j,i)}, & i > j. \end{cases} \tag{51}$$

By construction,

$$A_t^{(g)\top} = -A_t^{(g)}. \tag{52}$$

The complete skew-symmetric matrix is formed by placing the G group matrices along the diagonal:

$$A_t = \mathrm{BlockDiag}\left(A_t^{(1)}, \ldots, A_t^{(G)}\right) \in \mathbb{R}^{d_f \times d_f}. \tag{53}$$

Since each block satisfies Eq. 52, the complete matrix also satisfies

$$A_t^\top = -A_t. \tag{54}$$

Accordingly, the mapping $\Psi(\cdot)$ introduced in the main paper can be written as

$$A_t = \Psi(v_t), \tag{55}$$

where $\Psi(\cdot)$ consists of the bias-free projection in Eq. 47 followed by the grouped skew-symmetric construction in Eq. 53.

Grouped Cayley transformation. For each group g, the corresponding orthogonal transformation is obtained through the Cayley transform:

$$R_t^{(g)} = \left(I_{d_g} - A_t^{(g)}\right)\left(I_{d_g} + A_t^{(g)}\right)^{-1}, \tag{56}$$

where I_{d_g} denotes the $d_g \times d_g$ identity matrix.

Because $A_t^{(g)}$ is real and skew-symmetric, its eigenvalues are purely imaginary. Therefore, -1 cannot be an eigenvalue of $A_t^{(g)}$, and $I_{d_g} + A_t^{(g)}$ is nonsingular. The Cayley transformation in Eq. 56 is thus well defined.

The resulting matrix is orthogonal:

$$R_t^{(g)\top} R_t^{(g)} = I_{d_g}. \tag{57}$$

The complete transformation is assembled as

$$R_t = \mathrm{BlockDiag}\left(R_t^{(1)}, \ldots, R_t^{(G)}\right) \in \mathbb{R}^{d_f \times d_f}. \tag{58}$$

Since all blocks are orthogonal,

$$R_t^\top R_t = I_{d_f}. \tag{59}$$

The grouped construction is equivalent to applying the Cayley transform directly to the block-diagonal skew-symmetric matrix A_t:

$$R_t = \left(I_{d_f} - A_t\right)\left(I_{d_f} + A_t\right)^{-1}. \tag{60}$$

Because the projection is bias free, zero steering gives

$$v_t = 0 \implies \theta_t = 0 \implies A_t = 0 \implies R_t = I_{d_f}. \tag{61}$$

Thus, the zero steering vector exactly recovers the unedited latent representation of the frozen base policy.

Sharing across suffix positions and flow-matching steps. BELT-VLA predicts one steering vector and constructs one orthogonal transformation at each policy step. We use t to index the policy step and k to index the internal flow-matching integration step.

Let

$$f_t^{(k)} \in \mathbb{R}^{L_f \times d_f}, \qquad k = 1, \ldots, N_{\text{flow}}, \tag{62}$$

denote the latent action-flow representation at the k-th flow-matching step during action-chunk generation at policy step t. Here, L_f denotes the number of suffix positions and d_f denotes the feature dimension.

For suffix position ℓ, the corresponding feature vector is

$$f_{t,\ell}^{(k)} \in \mathbb{R}^{1\times d_f}, \qquad \ell = 1, \ldots, L_f. \tag{63}$$

The grouped orthogonal edit is applied along the feature dimension:

$$f_{t,\ell}^{(k)\prime} = f_{t,\ell}^{(k)} R_t^\top. \tag{64}$$

Equivalently, the complete suffix representation is transformed as

$$f_t^{(k)\prime} = f_t^{(k)} R_t^\top. \tag{65}$$

The steering vector v_t is sampled once at policy step t according to Eq. 26, and the resulting transformation R_t is reused for all internal flow-matching steps,

$$k = 1, \ldots, N_{\text{flow}}. \tag{66}$$

Therefore, BELT-VLA does not predict separate transformations for different suffix positions or different flow-matching integration steps. Instead, one state-conditioned orthogonal transformation is generated per policy step and shared throughout the generation of the corresponding action chunk. For notational simplicity, the flow-step superscript k is omitted in the main paper.

Computational complexity. Constructing a dense $d_f \times d_f$ Cayley transformation requires solving a dense linear system and has computational complexity

$$\mathcal{O}\left(d_f^3\right). \tag{67}$$

For grouped editing, each of the G blocks has size $d_g \times d_g$. Constructing all Cayley blocks therefore requires

$$\mathcal{O}\left(G d_g^3\right) = \mathcal{O}\left(d_f d_g^2\right). \tag{68}$$

The steering projection is evaluated once per policy step and requires

$$\mathcal{O}\left(M d_v\right) \tag{69}$$

operations.

Applying a dense transformation to all L_f suffix positions at one flow-matching step requires

$$\mathcal{O}\left(L_f d_f^2\right) \tag{70}$$

operations.

In contrast, applying the grouped transformation requires

$$\mathcal{O}\left(L_f G d_g^2\right) = \mathcal{O}\left(L_f d_f d_g\right) \tag{71}$$

operations per flow-matching step.

Because the same transformation R_t is reused across all N_{flow} flow-matching steps, it is constructed only once per policy step but applied N_{flow} times. The total additional computation introduced by grouped editing for one action chunk is therefore

$$\mathcal{O}\left(M d_v + d_f d_g^2 + N_{\text{flow}} L_f d_f d_g\right). \tag{72}$$

For comparison, a dense implementation requires

$$\mathcal{O}\left(d_f^3 + N_{\text{flow}} L_f d_f^2\right) \tag{73}$$

operations, excluding the cost of predicting the dense skew-symmetric coefficients.

The transformation storage is also reduced from

$$\mathcal{O}\left(d_f^2\right) \tag{74}$$

for a dense transformation to

$$\mathcal{O}\left(G d_g^2\right) = \mathcal{O}\left(d_f d_g\right) \tag{75}$$

for the grouped implementation. Since $d_g \ll d_f$, grouped editing substantially reduces both transformation construction and application costs.

Table 5: Implementation settings of the grouped orthogonal editor.

Hyperparameter	Value
Action-flow feature dimension d_f	1024
Steering dimension d_v	64
Number of groups G	64
Feature dimension per group d_g	16
Independent coefficients per group m_g	120
Total grouped coefficients M	7680
Flow-matching steps N_{flow}	10

A.7 Baseline and Evaluation Details

We compare BELT-VLA with supervised fine-tuning, pretrained robot foundation model fine-tuning, and online human-in-the-loop adaptation baselines. All methods are evaluated on the same AgileX ALOHA hardware, observation streams, action representation, task initial-state randomization, and success criteria described in Appendix A.1. All task-specific policies use the same set of 100 teleoperated demonstrations per task.

$\pi_{0.5}$+LoRA. The $\pi_{0.5}$+LoRA baseline fine-tunes the pretrained $\pi_{0.5}$ policy using the official OpenPI LoRA configuration, while freezing the non-LoRA backbone parameters. LoRA adapters are applied to the attention and feed-forward modules of the Gemma-2B vision-language backbone and the Gemma-300M action expert, with ranks/scaling factors $(r, \alpha_{\text{LoRA}}) = (16, 16)$ and $(32, 32)$, respectively, and no LoRA dropout. The policy is trained on the 100 task demonstrations with behavior-cloning / flow-matching supervision for 50,000 gradient steps. No online rollouts, intervention labels, or task rewards are used.

GR00T N1.6 fine-tuning. For GR00T N1.6, we use the official NVIDIA Isaac-GR00T N1.6 fine-tuning interface and initialize from the released pretrained checkpoint. The model is adapted to each task using the same 100 teleoperated demonstrations and an ALOHA-compatible observation/action modality configuration. We use the default supervised action-prediction objective and the official fine-tuning defaults: the vision-language backbone is frozen, while the projector and diffusion action head are trainable. GR00T N1.6 is evaluated only as an offline fine-tuned policy, without online intervention feedback.

RL-Token. We reimplement RL-Token following the published method. The pretrained VLA remains frozen, and a lightweight actor–critic policy is trained on a compact token representation extracted from the VLA features. The learned policy refines the reference VLA actions while remaining regularized toward the frozen policy. RL-Token uses the same autonomous interaction budget as BELT-VLA. Human takeover is used only for safe execution, and intervention transitions are not used for training.

HIL-SERL. We retain HIL-SERL's off-policy actor–critic optimization and replay-buffer treatment of demonstration and human-correction data, while adapting the observation and action interfaces to the ALOHA platform. The human takeover criterion and total autonomous interaction budget are matched to those used for BELT-VLA. Intervention and recovery transitions are stored according to the HIL-SERL training protocol.

Online training and evaluation protocol. All online adaptation methods use the same budget of 5,120 autonomous policy steps per task. Human-controlled intervention and recovery steps, environment resets, and evaluation trials are excluded from this budget. When a baseline uses intervention data by design, those transitions are processed according to its original training protocol. Policies are evaluated periodically during online adaptation, with learning paused during evaluation. At each evaluation point, we run 50 fully autonomous trials per task with no human intervention or online parameter updates.

Table 6: Training and intervention settings of the evaluated methods.

Method	Task-specific update	Online signal	Intervention use
$\pi_{0.5}$+LoRA	LoRA adapters; offline	Flow-matching supervision	Not used
GR00T N1.6	Projector and diffusion head; offline	Action-prediction supervision	Not used
RL-Token	Actor–critic head; online	Sparse task reward	Safety takeover only
HIL-SERL	Actor–critic policy; online	Sparse task reward	Corrective transitions
BELT-VLA	Edit policy and auxiliary networks; online	Novelty–risk reward	Pre-intervention labels

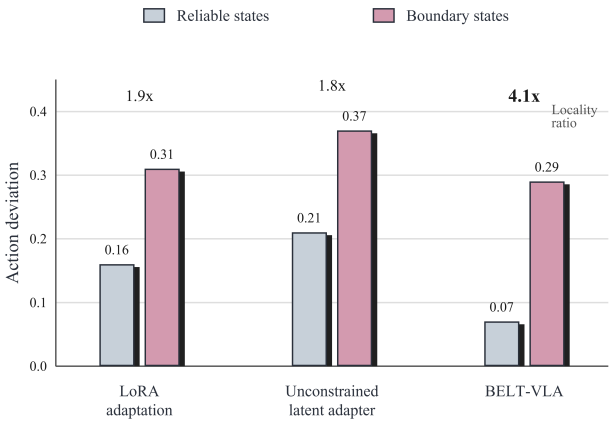


Figure 7: Normalized action deviations on reliable and boundary states. Each method is compared with its corresponding pre-adaptation policy using five matched initial flow-matching noise seeds. Values above the bars indicate the locality ratio.

Trial randomization. For the 50 autonomous evaluation trials, initial object poses are randomized within task-specific valid workspace ranges, including object position, in-plane orientation, and relative placement. The same randomized initial-state list is used for all methods within each task, ensuring that success rates are compared under matched task difficulty.

A.8 Additional Locality Analysis

We examine whether each adaptation method concentrates its policy changes near intervention-defined failure regions. For method m, let $\pi_{\text{ref}}^{(m)}$ denote the policy before adaptation and $\pi_{\text{adapt}}^{(m)}$ the final adapted policy. For BELT-VLA, these correspond to π_{off} and π_{BELT}, respectively. All observations used in this analysis are held out from training.

For each observation o_t, the two policies generate $J = 5$ action chunks using matched initial flow-matching noise seeds. We define their normalized action deviation as

$$D_t^{(m)} = \frac{1}{JH} \sum_{j=1}^{J} \sum_{h=1}^{H} \left\| \left(\mathbf{a}_{\text{adapt},t,h}^{(m,j)} - \mathbf{a}_{\text{ref},t,h}^{(m,j)} \right) \oslash \boldsymbol{\sigma}_a \right\|_2 , \tag{76}$$

where H is the action-chunk horizon, $\boldsymbol{\sigma}_a$ is the per-dimension action standard deviation estimated from the demonstrations, and $\oslash$ denotes element-wise division. BELT-VLA and the unconstrained latent adapter use their mean steering outputs during evaluation.

Boundary states, denoted by $\mathcal{B}$, consist of the K autonomous steps immediately preceding human takeover. Reliable states, denoted by $\mathcal{R}$, are sampled from successful intervention-free trajectories

and matched to the boundary states by task stage. For each method, we compute the locality ratio as

$$\rho_{\mathrm{loc}}^{(m)} = \frac{\mathbb{E}_{t\in\mathcal{B}}\left[D_t^{(m)}\right]}{\mathbb{E}_{t\in\mathcal{R}}\left[D_t^{(m)}\right]}. \tag{77}$$

A larger $\rho_{\mathrm{loc}}^{(m)}$ indicates that adaptation-induced changes are more concentrated near intervention-defined failure regions.

As shown in Fig. 7, BELT-VLA produces mean action deviations of 0.07 on reliable states and 0.29 on boundary states, yielding a locality ratio of 4.1. The corresponding ratios for $\pi_{0.5}$+LoRA and the unconstrained latent adapter are 1.9 and 1.8, respectively. These results show that BELT-VLA concentrates its policy changes more strongly near failure-related states while producing smaller deviations in reliable regions.