CHUNKRL: LEARNING ADAPTIVE ACTION HORIZONS FOR FAST AND ACCURATE ROBOT MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Robotic manipulation is not uniformly difficult: free-space motion can tolerate long open-loop execution, while brief contact-rich stages demand frequent re-observation and correction. Yet modern Vision-Language-Action models commit to a fixed action-chunk horizon, forcing every stage to share the same trade-off between redundant policy evaluations and delayed feedback. We present ChunkRL, which turns the action horizon from a static inference hyperparameter into an observation-conditioned feedback policy. ChunkRL first builds a parameter-efficient bank of short-, medium-, and long-horizon experts from offline demonstrations while freezing the backbone. It then optimizes an Adaptive Chunk Selector with per-decision Group Relative Policy Optimization, allowing the selector to infer feedback needs from closed-loop rollouts and trade off task success against feedback cost. We introduce Adaptive Layer-Pruning, which couples temporal feedback allocation with per-call computation by conditioning policy depth on the selected horizon and VLM residual signals. Across LIBERO, VLABench, and CALVIN, our 0.5B model improves average success rates by 0.8, 6.1, and 3.2 percentage points, respectively, while delivering $1.42\times$–$3.43\times$ inference speedups. These results show that scheduling when to observe and how much to compute improves manipulation reliability and efficiency.

1 INTRODUCTION

Robotic manipulation varies in difficulty over time. A robot may spend much of an episode moving through free space, yet require precise, rapidly corrected control during brief contact-rich stages. Vision-Language-Action models (VLAs) Black et al. (2024); Intelligence et al. (2025); Kim et al. (2024; 2025); Yue et al. (2024); Hu et al. (2024) have substantially advanced language-conditioned manipulation by mapping visual observations and instructions to robot actions. Many modern VLAs predict fixed action chunks Zhao et al. (2023); Zheng et al. (2025); Li et al. (2024); Tian et al. (2025) in a receding-horizon closed loop: given the current observation, the policy predicts a chunk, executes it for the selected horizon, and replans from the next observation. The chunk horizon therefore controls both chunk length and feedback frequency for correcting execution errors.

This coupling creates a tension. Short horizons provide dense feedback and timely correction, but require expensive policy evaluations. Long horizons reduce policy calls, but lengthen the interval between closed-loop feedback updates and allow errors to persist. More importantly, the appropriate trade-off changes within an episode. Stable motion can proceed with sparse feedback, whereas grasping, contact, and placement demand frequent re-observation. A fixed horizon must consequently either pay the computational cost of the hardest stage throughout the episode or accept delayed correction when precise control is required. As illustrated in Fig. 1, the central question is not simply how many actions to predict, but when the policy should observe again.

We introduce **ChunkRL**, a framework that treats the action horizon as an observation-conditioned feedback decision. Built on VLA-Adapter Wang et al. (2026), ChunkRL uses an Adaptive Chunk Selector to choose a feedback horizon at each macro-decision. The selected horizon activates the corresponding action expert and determines how long its chunk is executed before the next observation is acquired. This allows the policy to use longer chunks when feedback is redundant and switch to shorter chunks when timely correction is needed, thereby adapting feedback frequency to the evolving requirements of each manipulation stage.

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.

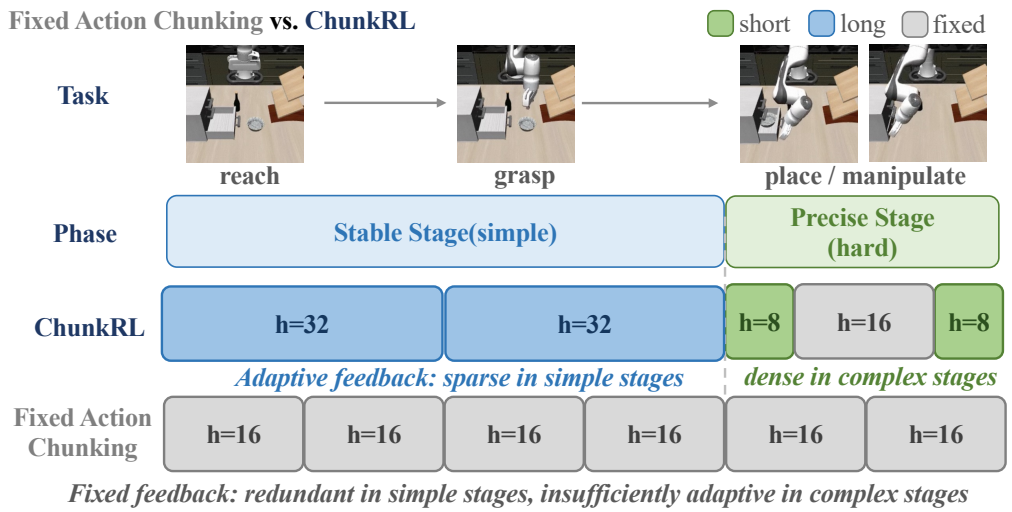


Figure 1: **ChunkRL at a glance.** ChunkRL adaptively allocates feedback for error correction, using sparse feedback in simple stages and dense feedback in challenging stages, in contrast to fixed chunking, which maintains a constant feedback frequency throughout the episode.

Learning this behavior poses two challenges: action generation must remain reliable across horizons, and the discrete horizon decisions lack direct supervision. We address them with the two-stage procedure shown in Fig. 2. In Stage 1, we construct a parameter-efficient bank of multi-horizon experts from offline demonstrations. The VLM and policy weights remain frozen; shared VLM LoRA adapters refine the visual-language representation, while independent horizon-specific AdaLN modules and lightweight action heads specialize the policy for short-, medium-, and long-horizon action generation. In Stage 2, we freeze the expert bank and optimize the Adaptive Chunk Selector using per-decision Group Relative Policy Optimization (GRPO) Guo et al. (2025). An outcome-aware reward favors fewer feedback calls for successful rollouts and denser feedback after failures, enabling the selector to learn when efficient execution with sparse closed-loop feedback is safe and when timely correction is necessary for reliable execution across manipulation stages.

Adaptive feedback changes not only how often the policy runs, but also how much computation each call requires. Executing the same policy depth at every feedback frequency is suboptimal: frequent feedback under short horizons limits how long approximation errors can persist, whereas sparse feedback under long horizons demands more conservative computation. We therefore introduce **Adaptive Layer-Pruning (ALP)**, which conditions the executed policy depth on the selected horizon. In our aligned VLM–policy architecture, ALP estimates policy-layer importance from VLM residual updates, skips more low-importance layers for short horizons, and retains more layers for long horizons. ChunkRL thus allocates computation jointly across time, through adaptive feedback scheduling, and within each policy call, through horizon-aware depth selection.

We evaluate ChunkRL on LIBERO Liu et al. (2023), VLABench Zhang et al. (2025b), and CALVIN Mees et al. (2022). At 0.5B, ChunkRL improves average success from 98.1% to 98.9% on LIBERO and 31.1% to 37.2% on VLABench. On CALVIN, it raises five-task completion from 69.2% to 74.7% and average sequence length from 4.27 to 4.43. It delivers $1.42\times$–$3.43\times$ inference speedups on LIBERO and CALVIN. These results show that execution-context-aware feedback improves manipulation reliability and inference efficiency across benchmarks. Our contributions are:

- We formulate horizon selection as adaptive feedback scheduling and introduce ChunkRL to replace a fixed horizon with observation-conditioned feedback decisions.
- We develop a two-stage framework combining parameter-efficient multi-horizon experts, per-decision GRPO, and ALP for feedback-conditioned policy-depth allocation.
- Experiments on LIBERO, VLABench, and CALVIN demonstrate consistent improvements in task performance and inference efficiency using only 0.5B parameters.

2 Related Work

Vision-Language-Action models (VLAs) extend pretrained vision-language models to map visual observations and language instructions to robot actions Black et al. (2024); Intelligence et al. (2025); Qu et al. (2025). Existing VLAs mainly follow two paradigms: autoregressive models such as OpenVLA Kim et al. (2024) tokenize and sequentially decode actions, whereas flow-matching and diffusion-based models, represented by the π series Black et al. (2024); Intelligence et al. (2025), use action experts to generate continuous control. Many modern VLAs employ action chunking introduced by ACT Zhao et al. (2023), while retaining closed-loop control by repeatedly replanning from new observations. The chunk horizon therefore controls both inference frequency and the temporal granularity of feedback, yet it is typically fixed throughout an episode.

2.1 Action Chunking and Feedback Frequency

The action horizon creates a direct trade-off between efficiency and correction: long chunks reduce policy calls but delay feedback, while short chunks provide timely correction at higher computational cost. AAC selects chunk sizes from action entropy without retraining Liang et al. (2026), and MoH uses cross-horizon consensus to choose variable-length prefixes Jing et al. (2025). RTC Black et al. (2025) and A2C2 Sendai et al. (2025) improve reactivity within fixed chunking schemes, whereas PD-VLA Song et al. (2025) and OpenVLA-OFT Kim et al. (2025) accelerate action decoding. However, these methods largely rely on fixed schedules or training-free heuristics. ChunkRL instead formulates horizon selection as episode-level feedback allocation, learning when sparse feedback is sufficient and when dense feedback is required for error correction.

2.2 Reinforcement Learning for Adaptive Feedback

Policy-optimization methods such as PPO Schulman et al. (2017), DPO Rafailov et al. (2023), and GRPO Guo et al. (2025) enable post-training beyond supervised objectives. Recent VLA studies further improve long-horizon control through trajectory-aware optimization Chen et al. (2025) or scalable online reinforcement learning with limited demonstrations Li et al. (2025). ChunkRL uses reinforcement learning for a complementary goal: optimizing the discrete feedback schedule. By treating the horizon as an action, GRPO learns when to commit to long feedback-sparse chunks and when to request earlier feedback through shorter chunks.

2.3 Efficient Inference via Adaptive Layer-Pruning

LayerDrop Fan et al. (2019) enables depth-adjustable subnetworks, while MoLe-VLA Zhang et al. (2025a) dynamically routes VLA inputs through selected transformer layers. ChunkRL couples layer pruning with feedback scheduling rather than treating them independently. Short chunks receive new observations more frequently, so small approximation errors caused by skipping low-importance layers can be corrected at the next feedback step; long chunks receive sparser feedback and therefore use more conservative pruning. This feedback-aware coupling reduces the cost of each policy call while controlling how long pruning-induced errors can persist.

3 Method

3.1 Overview

Given the observation o_t and language instruction ℓ, the VLM f extracts a visual-language feature q_t. Conditioned on the feedback horizon h, the policy model Φ maps q_t to a policy feature ϵ_t, decoded by an action head into the predicted action chunk A_t comprising h future actions:

$$q_t = f(o_t, \ell), \quad \epsilon_t = \Phi(q_t \mid h), \quad A_t = \mathrm{Head}(\epsilon_t). \tag{1}$$

Action-chunking VLAs keep h unchanged throughout execution and replan only after the chunk has been executed. Since the horizon determines when the next observation is acquired, this produces a fixed-rate receding-horizon closed loop. This results in redundant policy calls during stable stages, while delayed feedback in stages requiring precise control prevents timely error correction.

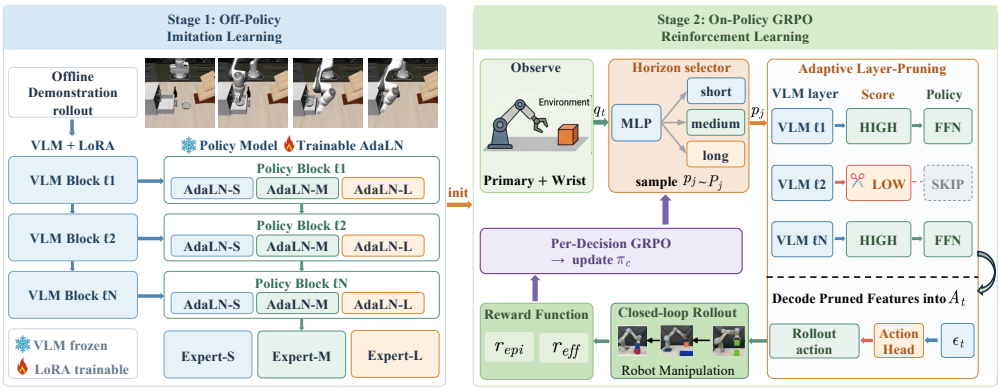


Figure 2: **Overview of the ChunkRL framework. Stage 1** uses offline demonstrations to train VLM LoRA adapters and horizon-specific AdaLN modules while freezing backbones, yielding short-, medium-, and long-horizon experts. **Stage 2** freezes the experts and trains the Adaptive Chunk Selector with on-policy GRPO. At each macro-decision, the sampled horizon determines the active expert and feedback interval, while ALP uses that horizon and VLM-layer importance to bypass low-importance policy layers. After each rollout, a single trajectory-level reward composed of the episode-outcome term r_{epi} and the outcome-conditioned feedback-efficiency term r_{eff} provides the GRPO learning signal for all recorded horizon decisions.

Thus, ChunkRL formulates horizon selection as an observation-conditioned feedback decision rather than a fixed inference hyperparameter. Specifically, the Adaptive Chunk Selector π_c predicts a distribution P_j over the candidate horizons from q_t and samples a horizon decision p_j:

$$P_j = \pi_c(q_t), \quad p_j \sim P_j, \quad \epsilon'_t = \Phi(q_t \mid p_j). \tag{2}$$

The sampled p_j dynamically selects the corresponding horizon expert, replacing the fixed feedback horizon h used by conventional VLAs. The selected expert maps q_t to a policy feature $\epsilon'_t = \Phi(q_t; p_j)$, which is decoded by the action head into the predicted action chunk $A'_t = \text{Head}(\epsilon'_t)$. To support feedback-driven dynamic horizon sampling, we keep the shared policy model frozen and assign each candidate horizon an independent set of trainable AdaLN parameters. These parameters modulate the intermediate policy features, enabling the experts to specialize in short-, medium-, and long-horizon action generation. The selected horizon determines how long A'_t is executed before the next observation is acquired, thereby forming an adaptive feedback loop.

However, executing the same number of policy layers at every feedback frequency within the feedback loop is suboptimal, as the accumulation of approximation errors depends on horizon. This observation motivates Adaptive Layer-Pruning (ALP), which adjusts the number of executed policy layers according to p_j. For short horizons, frequent feedback limits error accumulation and therefore permits more aggressive pruning. In contrast, long horizons increase the interval between closed-loop feedback updates, requiring more policy layers to ensure reliable action prediction.

Overall, the complete framework follows the two-stage procedure illustrated in Figure 2. Stage 1 learns the multi-horizon experts from offline demonstrations by optimizing LoRA adapters and horizon-specific AdaLN modules while keeping the original VLM and policy weights frozen. Stage 2 freezes the learned experts and optimizes π_c with per-decision GRPO. At each macro-decision, the sampled horizon selects the corresponding expert and determines the feedback interval, while ALP uses the same horizon and the aligned VLM-layer importance scores to determine the executed policy depth. After each rollout, a single trajectory-level reward provides the relative advantage shared by all recorded horizon decisions.

3.2 Multi-Horizon Experts

During Stage 1, we construct a bank of multi-horizon experts that generate action chunks at different feedback intervals. Each expert is trained using horizon-matched action chunks from offline demonstrations. As defined in the overview, the VLM f extracts the shared visual-language feature q_t. We freeze the VLM backbone and train its LoRA adapters so all experts share the condition feature.

The policy model Φ is kept frozen. To specialize it for short-, medium-, and long-horizon action generation, we insert independent trainable AdaLN modules after its attention blocks. Let ϵ_t^b denote the feature obtained when q_t passes through the b-th policy block. The AdaLN operation is

$$\mathrm{AdaLN}(\epsilon_t^b \mid p_j) = \left(1 + \Delta\gamma^b\right) \odot \mathrm{LN}(\epsilon_t^b) + \beta^b, \tag{3}$$

where $\Delta\gamma^b$ and β^b denote the trainable residual scale and shift at the b-th policy layer, respectively. Each horizon expert learns its own set of AdaLN parameters for horizon-specific feature modulation.

After the final policy block, the feature ϵ_t is decoded by the lightweight action head as $A_t = \mathrm{Head}(\epsilon_t)$. Each expert is associated with a horizon: short-horizon experts provide frequent feedback for timely correction, whereas long-horizon experts use longer intervals between closed-loop feedback updates and fewer policy calls. Throughout Stage 1, we train these experts on offline demonstrations, ensuring each is adapted to its designated feedback frequency.

3.3 GRPO Policy Learning for Adaptive Feedback

Building on the frozen bank of horizon-specific action experts learned in Stage 1, Stage 2 optimizes the Adaptive Chunk Selector π_c while keeping all experts fixed. At each macro-decision, the selector samples a horizon that activates the corresponding expert and determines the interval to the next observation and closed-loop replanning step. Horizon selection is formulated as a discrete policy decision for adaptive feedback scheduling, while the action-generation process remains unchanged.

Since horizon selection is discrete and no direct supervision for the optimal horizon is available, it cannot be directly optimized through standard supervised learning. We therefore employ per-decision GRPO in this work to learn observation-conditioned horizon choices at different execution stages, enabling adaptive feedback scheduling. From each initial state, the selector rolls out a group of complete trajectories. At every macro-decision, we record the predicted horizon distribution, the sampled horizon, and its associated normalized feedback count c_i. Once a rollout terminates, every recorded horizon decision receives a reward that is explicitly decomposed into an episode-outcome term and an outcome-conditioned feedback-efficiency term:

$$r_i = r_{epi} + r_{eff}. \tag{4}$$

The episode-level term r_{epi} is shared by decisions within rollouts, providing trajectory-level supervision for task completion. We set $r_{epi} = 1$ for successful episode and $r_{epi} = -1$ for failed episode. The decision-level term r_{eff} shapes feedback allocation according to the episode outcome:

$$r_{eff} = \begin{cases} -\lambda_s c_i, & \text{if the episode succeeds,} \\ -\lambda_f (1 - c_i), & \text{if the episode fails.} \end{cases} \tag{5}$$

Here, $c_i \in [0, 1]$ denotes the normalized feedback count induced by the selected horizon, where a larger c_i corresponds to denser feedback and a shorter execution horizon. The coefficients λ_s and λ_f control the feedback-shaping strengths under successful and failed rollouts, respectively.

For successful rollouts, r_{eff} encourages sparser feedback and longer-horizon execution, reducing redundant evaluations when task completion is reliable. For failed rollouts, it biases the selector toward denser feedback and shorter horizons, enabling timely error correction. Together, these terms separate task success from feedback scheduling and discourage the selector from collapsing to a single fixed horizon.

GRPO then constructs a relative learning signal directly from rollouts without training a separate value model. We pool all decision rewards from trajectories initialized at the same state and normalize them within the rollout group:

$$\hat{A}_i = \frac{r_i - \mu}{\sigma + \epsilon}, \tag{6}$$

where μ and σ are the group mean and standard deviation, and ϵ is a small constant for numerical stability. $\hat{A}_i$ measures whether a horizon decision performs better or worse than the alternatives explored under the same rollout context. Finally, the selector is updated with the per-decision objective

$$\mathcal{L}_{\mathrm{GRPO}} = \mathbb{E}_i \left[-\hat{A}_i \log(P_j) - \alpha \mathbb{H}(P_j) \right]. \tag{7}$$

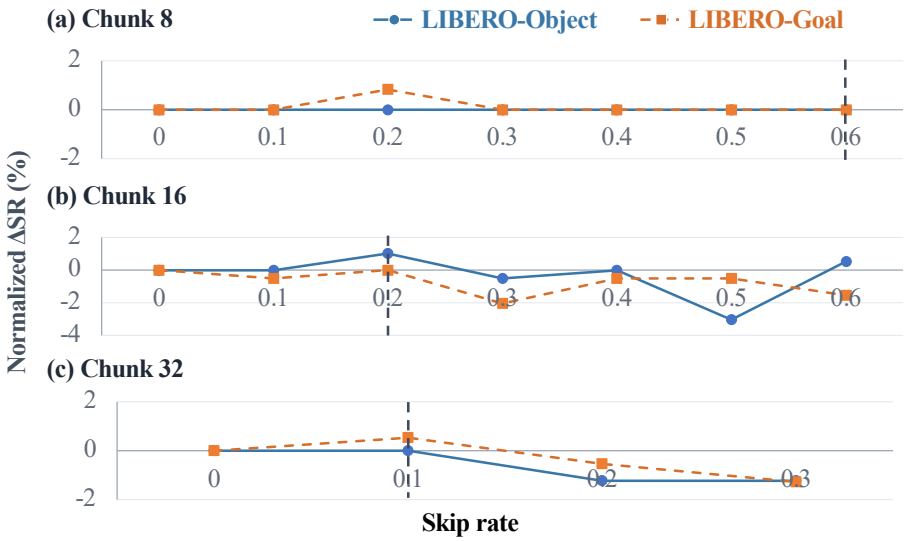


Figure 3: ALP skip-rate trade-off on LIBERO-Goal and LIBERO-Object. The curves show the normalized change in success rate relative to $\rho = 0$. A larger skip rate executes fewer policy layers and enables faster inference. Vertical markers indicate the selected horizon-aware operating points: $\rho = 0.6$ for $h = 8$, $\rho = 0.2$ for $h = 16$, and $\rho = 0.1$ for $h = 32$.

In the policy-gradient term, $\log(P_j)$ denotes the log-probability of the sampled horizon under the predicted distribution P_j. This term increases the likelihood of decisions with positive relative advantage and decreases that of decisions with negative advantage. The entropy term $\alpha\mathbb{H}(P_j)$ maintains exploration over the candidate horizons and prevents premature collapse. We omit KL regularization because its coefficient is set to zero. Throughout Stage 2, gradients are restricted to π_c; the frozen expert bank continues to provide reliable action chunks, while GRPO learns when each feedback interval should be used.

3.4 Adaptive Layer-Pruning

After sampling the feedback horizon p_j, the selected expert executes its action chunk until the next observation is acquired. The horizon therefore determines both the feedback interval and how long approximation errors persist. Applying the same policy depth to every horizon is suboptimal. Short horizons require frequent policy evaluations and therefore consume more computational resources, but their dense feedback enables rapid error correction and permits more aggressive pruning. In contrast, long horizons execute longer action sequences, over which errors may accumulate, and therefore require more conservative pruning. Motivated by this distinction, Adaptive Layer-Pruning (ALP) adopts a horizon-dependent pruning strategy that minimizes inference overhead while keeping error accumulation under control.

Many VLA architectures employ VLM and policy model with the same number of blocks, allowing features at corresponding layers to interact through cross-attention or shared-attention mechanisms. This one-to-one layer alignment enables ALP to estimate the importance of each policy layer directly from the VLM forward pass, without repeatedly evaluating the policy model. Specifically, for the b-th VLM layer, we measure the relative magnitude of its residual update:

$$\mathcal{I}_b = \frac{\|\Delta q_b\|}{\|q_b\| + \delta}, \tag{8}$$

where $\mathcal{I}_b$ denotes the importance score of the b-th layer, q_b is the input feature of the corresponding VLM layer, Δq_b is its residual update, and δ is a small constant for numerical stability. A smaller $\mathcal{I}_b$ means that the layer changes the VLM representation less, indicating that its aligned policy layer is less important for the current input.

Given p_j, ALP first determines a horizon-conditioned skip rate ρ and ranks the policy layers according to their aligned importance scores. It then selects the fraction ρ of layers with the lowest

Table 1: Comparison on LIBERO and VLABench. In the Avg. SR columns, values in parentheses report speedup over the baseline; all other values are success rates. Bold denotes the best result.

Method	Scale	LIBERO					VLABench					
		Spatial	Object	Goal	Long	Avg. SR	In-Dist.	Cross-Cat.	Common Sense	Semantic Instr.	Unseen Texture	Avg. SR
Large Models												
OpenVLA Kim et al. (2024)	7B	84.7	88.4	79.2	53.7	76.5	–	–	–	–	–	–
ThinkAct Huang et al. (2026)	7B	88.3	91.4	87.1	70.9	84.4	–	–	–	–	–	–
PD-VLA Song et al. (2025)	7B	95.5	96.7	94.9	91.7	94.7	–	–	–	–	–	–
Small Models												
π_0 Black et al. (2024)	3B	96.8	98.8	95.8	85.2	94.2	47.0	21.2	29.1	17.3	32.2	29.4
π_0-FAST Pertsch et al. (2025)	3B	96.4	96.8	88.6	60.2	85.5	40.6	22.6	18.0	16.1	25.6	24.6
MOH Jing et al. (2025)	3B	99.0	98.8	96.4	91.4	96.4	–	–	–	–	–	–
AAC Liang et al. (2026)	3B	94.4	98.6	94.2	92.8	95.0	–	–	–	–	–	–
Tiny Models												
Seer Tian et al. (2025)	0.6B	–	–	–	78.7	78.7	–	–	–	–	–	–
X-VLA Zheng et al. (2025)	0.9B	98.2	98.6	97.8	97.6	98.1	40.0	11.0	33.0	39.0	24.0	29.4
VLA-OS Gao et al. (2026)	0.5B	87.0	96.5	92.7	66.0	85.6	–	–	–	–	–	–
Baseline Wang et al. (2026)	0.5B	**99.6**	99.6	98.2	95.0	98.1 (1.0×)	40.2	25.6	34.8	34.8	20.2	31.1 (1.0×)
ChunkRL	**0.5B**	99.2	**99.6**	**98.6**	**98.3**	**98.9 (1.9×)**	**50.0**	**34.3**	**38.8**	**35.6**	**27.5**	**37.2 (1.5×)**

$\mathcal{I}_b$ values and bypasses these layers through their residual paths, while executing the remaining layers normally. Short horizons adopt a larger skip rate because frequent closed-loop feedback enables rapid error correction, whereas long horizons retain more layers to maintain reliable predictions over extended open-loop execution. Finally, the resulting policy feature is decoded by the action head of the selected expert to produce the action chunk A'_t.

ALP reuses the residual updates already computed when extracting q_t, requiring neither an additional importance estimator nor repeated policy forward passes. By allowing p_j to determine both the feedback interval and the executed policy depth, ALP couples feedback scheduling with computation allocation: it saves more computation when upcoming feedback can quickly correct approximation errors and preserves more policy capacity when feedback is sparse.

3.5 Discussion

Figure 3 visualizes the skip rate trade-off, showing that increasing the skip rate accelerates inference, while the amount of pruning that can be tolerated varies substantially across execution horizons. Under short-horizon execution, deviations introduced by skipping low-importance layers can be corrected at the next, nearby policy call. As the horizon increases, however, the same approximation error may affect more open-loop actions before a new observation becomes available. Consequently, medium-horizon execution favors moderate pruning, whereas long-horizon execution requires a more conservative pruning strategy to preserve prediction reliability. A fixed skip rate would therefore either incur unnecessary computational overhead at short horizons or compromise prediction reliability at long horizons. To address this trade-off, ALP jointly adjusts the feedback interval and retained policy depth according to p_j, while using residual-based importance scores to determine which layers to bypass.

ChunkRL is effective under synchronous execution. Asynchronous chunking remains complementary but faces spatial inconsistencies from different action-entry times. Recent in-context-learning results also support video-referenced synchronous chunks, suggesting that synchronous and asynchronous chunking are distinct research directions.

4 Experiments

4.1 Implementation Details

We use VLA-Adapter with a 0.5B-parameter VLM backbone and train and evaluate all models on a single NVIDIA H200 GPU. The reported model scale refers only to the parameter count of the VLM backbone and excludes the policy model. To keep the system lightweight, we share the VLM–policy

Table 2: Results on CALVIN ABC→D. Columns 1–5 report success rates (%), Avg. Len. denotes the average number of consecutively completed tasks, and Avg. SR is the mean success rate.

Method	Scale	1 ↑	2 ↑	3 ↑	4 ↑	5 ↑	Avg. Len. ↑	Avg. SR ↑
Large Models								
UniVLA Wang et al. (2025)	7B	95.5	85.8	75.4	66.9	56.5	3.80	76.0
OpenVLA Kim et al. (2024)	7B	91.3	77.8	62.0	52.1	43.5	3.27	65.3
Small Models								
DeeR Yue et al. (2024)	3B	86.2	70.1	51.8	41.5	30.4	2.82	56.0
RoboFlamingo Li et al. (2024)	3B	82.4	61.9	46.6	33.1	23.5	2.48	49.5
VPP Hu et al. (2024)	1.5B	95.7	91.2	86.3	81.0	**75.0**	4.33	85.8
Tiny Models								
X-VLA Zheng et al. (2025)	0.9B	–	–	–	–	–	4.43	–
Seer Tian et al. (2025)	0.6B	96.3	91.6	86.1	80.3	74.0	4.28	85.7
Baseline	0.5B	97.5	**96.7**	85.2	78.3	69.2	4.27	85.4 (1.0×)
ChunkRL	**0.5B**	**99.4**	95.5	**90.1**	**83.1**	74.7	**4.43**	**88.6** (3.4×)

Table 3: Component ablation on LIBERO-Object using the 0.5B model.

Method	Success (%) ↑	Time (s) ↓
Baseline	99.6	1.496
+ Multi-Horizon Experts	98.6	0.825
+ GRPO	99.4	0.790
+ ALP (Full)	**99.6**	**0.622**

Table 4: Effect of GRPO rollout group on LIBERO-Goal.

Group Size	Success (%) ↑	Time (s) ↓
$K = 2$	97.4	**0.821**
$K = 4$	97.7	0.833
$K = 8$	**98.6**	0.864
$K = 16$	98.5	0.890

projection layer, yielding a 97 MB trainable-parameter footprint. Unless stated otherwise, we use $\lambda_s = 0.3$, $\lambda_f = 6.0$, and $K = 8$ GRPO rollouts per initial state. The fixed-horizon baseline executes chunk-8 actions, while ChunkRL selects from candidate horizons online.

4.2 Comparison Results

We evaluate ChunkRL on LIBERO, VLABench, and CALVIN to determine whether adaptive feedback can jointly improve manipulation success, generalization, and inference efficiency. As shown in Tables 1–2, our 0.5B model achieves the best average success rates of 98.9%, 37.2%, and 88.6%, respectively, while accelerating inference and competing with models up to 7B parameters.

On LIBERO, ChunkRL raises average success from 98.1% to 98.9%, mainly through the Long suite (95.0% to 98.3%), while Spatial and Object remain within 0.4 and 0.2 points of the baseline. Its $1.42\times$–$2.52\times$ speedups show that adaptive feedback reduces redundant evaluations while preserving timely correction. On VLABench, it improves all settings and raises average success from 31.1% to 37.2%, with gains of $+9.8$, $+8.7$, and $+7.3$ points on in-distribution, cross-category, and unseen-texture evaluation and $1.42\times$–$1.48\times$ speedups. This indicates adaptation to changing contexts rather than a fixed schedule. On CALVIN, ChunkRL raises average success from 85.4% to 88.6% and sequence length from 4.27 to 4.43. Gains of 4.9, 4.8, and 5.5 points at depths 3–5 reduce long-sequence error accumulation, with a $3.43\times$ speedup using 0.5B parameters.

Overall, ChunkRL is not simply choosing longer chunks: it uses sparse feedback during stable motion, dense feedback for precise correction, and ALP to reduce per-call computation when errors can be corrected soon. Allocating computation according to feedback needs therefore improves both reliability and efficiency at the same time.

4.3 Ablation Study

We conduct all ablations on LIBERO to examine the key design choices of ChunkRL. To isolate each factor, we vary one setting at a time while keeping the remaining training and evaluation configurations fixed. Following the method pipeline, we first isolate the contribution of each component and then analyze the optimization of the GRPO selector, the candidate horizons, its conditioning, and the horizon-aware skip rates used by ALP.

Table 5: Effect of candidate horizons on LIBERO-Object.

Horizons	Success (%) ↑	Time (s) ↓
8	**99.6**	1.496
(8, 16, 32)	99.6	0.622
(8, 32, 64)	97.8	**0.463**
(8, 16, 32, 64)	93.8	0.507
(8, 16, 32, 64, 128)	78.2	0.582

Table 6: Effect of selector input on LIBERO-Object.

Input	Success (%) ↑	Time (s) ↓
Fixed chunk-8	99.6	1.496
Primary only	98.4	0.520
Wrist only	97.0	**0.515**
State only	98.1	0.668
Concat	**99.6**	0.622

Analysis of Components. Table 3 adds three components cumulatively. Multi-Horizon Experts reduces the per-episode time from 1.496 s to 0.825 s, but success drops from 99.6% to 98.6%, indicating that horizon diversity alone is insufficient to determine when each expert should be used. Adding GRPO adaptively learns feedback timing and produces horizon allocations, restoring success to 99.4% while reducing time to 0.790 s. With ALP, the full model matches the baseline success of 99.6% at 0.622 s, reducing inference cost by 58.4%. This progression shows that adaptive feedback scheduling and horizon-aware policy depth are complementary: GRPO restores reliability, while ALP converts the selected feedback pattern into additional computational savings.

Analysis Of GRPO Rollout Group Size. Table 4 evaluates the number of trajectories used to estimate group-relative advantages. Increasing K from 2 to 8 steadily improves success from 97.4% to 98.6%, indicating that a larger group provides a more reliable relative learning signal. Increasing the group further to $K = 16$ yields no additional gain, while the inference times remain comparable across settings. We therefore use $K = 8$ as the default group size.

Analysis Of Candidate Horizons. Table 5 investigates how far feedback can be sparsified on the 0.5B model by varying the candidate horizons. Introducing long horizons such as 64 and 128 may appear to reduce inference time. However, overly sparse feedback sharply increases the failure rate, causing many episodes to terminate prematurely and artificially lowering the average episode-level inference time. Thus, the apparent speedup does not reflect efficient successful execution. Considering both task success and meaningful inference efficiency, including 64- or 128-step horizons as candidates does not yield a favorable trade-off. We therefore adopt $(8, 16, 32)$ as the final horizon set, which retains 99.6% success while reducing inference time from 1.496 s to 0.622 s. Overall, moderate horizons provide sufficient feedback diversity without sacrificing reliable error correction.

Analysis Of GRPO Selector Conditioning. Table 6 ablates four observation encodings for the GRPO horizon selector. *State only* uses the 8-D proprioceptive state, consisting of the end-effector position, axis-angle orientation, and gripper joint positions. *Primary only* and *Wrist only* condition the selector on the third-person RGB observation and the wrist-mounted RGB observation, respectively. *Concat* resizes both views and concatenates them along the channel dimension, yielding a six-channel visual input to the same lightweight MLP. Although all adaptive variants reduce inference time relative to the fixed chunk-8 baseline, relying on a single view may omits complementary information and results in lower success. The concatenated representation combines the primary view's global scene context with the wrist view's local manipulation cues, achieving 99.6% success while reducing inference time from 1.496 s to 0.622 s, a 58.4% reduction.

5 Conclusion

We presented **ChunkRL**, which reframes action-horizon selection as adaptive feedback scheduling for VLA manipulation. Rather than paying for dense replanning or committing to sparse feedback throughout an episode, ChunkRL learns to request observations at the granularity required by the execution stage. Its parameter-efficient multi-horizon experts provide reliable actions across horizons, while per-decision GRPO learns when each expert should be used. Adaptive Layer-Pruning couples this temporal allocation with horizon-aware policy depth, reducing computation within each feedback call. Across LIBERO, VLABench, and CALVIN, ChunkRL improves average success rates by 0.8, 6.1, and 3.2 percentage points, respectively, while delivering $1.42\times$–$3.43\times$ inference speedups using a 0.5B VLM backbone. These results support the view of efficient closed-loop control: a robot policy should adapt not only what actions to predict, but also when to observe and how much computation to spend before the feedback update.

AI USE STATEMENT

AI tools were used to polish the manuscript's written presentation.

ETHICS STATEMENT

This work involves no potential ethical risks and fully complies with the official ethics requirements; therefore, no human oversight is required.

REPRODUCIBILITY STATEMENT

The main text specifies the model components, training objectives, hyperparameters, and ablation settings used in the experiments. The appendix provides benchmark protocols, evaluation metrics, additional optimization details, and state-dependent feedback evaluations. Together, these sections document the information needed to reproduce the reported methodology and analyses. The relevant experimental results and code will be made publicly available upon acceptance.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. *ArXiv*, abs/2506.07339, 2025. URL https://api.semanticscholar.org/CorpusID:279251645.

Ze Chen, Runliang Niu, He Kong, and Qi Wang. Tgrpo :fine-tuning vision-language-action model via trajectory-wise group relative policy optimization. *ArXiv*, abs/2506.08440, 2025. URL https://api.semanticscholar.org/CorpusID:279260912.

Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*, 2019.

Chongkai Gao, Zixuan Liu, Zhenghao Chi, Junshan Huang, Xin Fei, Yiwen Hou, Yuxuan Zhang, Yudi Lin, Zhirui Fang, and Lin Shao. Vla-os: Structuring and dissecting planning representations and paradigms in vision-language-action models. *Advances in Neural Information Processing Systems*, 38:136705–136736, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu,

Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Frank Wang, and Fu-En Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. *Advances in Neural Information Processing Systems*, 38:82782–82802, 2026.

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.

Dong Jing, Gang Wang, Jiaqi Liu, Weiliang Tang, Zelong Sun, Yunchao Yao, Zhenyu Wei, Yunhui Liu, Zhiwu Lu, and Mingyu Ding. Mixture of horizons in action chunking. *ArXiv*, abs/2511.19433, 2025. URL https://api.semanticscholar.org/CorpusID:283245149.

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

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. Simplevla-rl: Scaling vla training via reinforcement learning. *ArXiv*, abs/2509.09674, 2025. URL https://api.semanticscholar.org/CorpusID:281252318.

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. In *International Conference on Learning Representations*, volume 2024, pp. 26703–26721, 2024.

Yuanchang Liang, Xiaobo Wang, Kai Wang, Shuo Wang, Xiaojiang Peng, Hao Chen, David Kim Huat Chua, and Prahlad Vadakkepat. Adaptive action chunking at inference-time for vision-language-action models. *ArXiv*, abs/2604.04161, 2026. URL https://api.semanticscholar.org/CorpusID:287202540.

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, 2023.

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.

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.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

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

Kohei Sendai, Maxime Alvarez, Tatsuya Matsushima, Yutaka Matsuo, and Yusuke Iwasawa. Leave no observation behind: Real-time correction for vla action chunks. *ArXiv*, abs/2509.23224, 2025. URL https://api.semanticscholar.org/CorpusID:281676331.

Wenxuan Song, Jiayi Chen, Pengxiang Ding, Han Zhao, Wei Zhao, Zhide Zhong, Zongyuan Ge, Jun Ma, and Haoang Li. Pd-vla: Accelerating vision-language-action model integrated with action chunking via parallel decoding. *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 13162–13169, 2025. URL https://api.semanticscholar.org/CorpusID:276770282.

Yang Tian, Sizhe Yang, Jia Zeng, Ping Wang, Dahua Lin, Hao Dong, and Jiangmiao Pang. Predictive inverse dynamics models are scalable learners for robotic manipulation. In *International Conference on Learning Representations*, volume 2025, pp. 92033–92052, 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.

Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025.

Yang Yue, Yulin Wang, Bingyi Kang, Yizeng Han, Shenzhi Wang, Shiji Song, Jiashi Feng, and Gao Huang. Deer-vla: Dynamic inference of multimodal large language models for efficient robot execution. *Advances in Neural Information Processing Systems*, 37:56619–56643, 2024.

Rongyu Zhang, Menghang Dong, Yuan Zhang, Liang Heng, Xiaowei Chi, Gaole Dai, Li Du, Dan Wang, Yuan Du, and Shanghang Zhang. Mole-vla: Dynamic layer-skipping vision language action model via mixture-of-layers for efficient robot manipulation. In *AAAI Conference on Artificial Intelligence*, 2025a. URL https://api.semanticscholar.org/CorpusID:277321753.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, et al. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11142–11152, 2025b.

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

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.

SUPPLEMENTARY MATERIAL

This supplementary material provides the dataset and evaluation details together with additional analyses of ChunkRL beyond the main-paper results and their practical implications. It is organized into three parts:

A) Dataset and Evaluation Metrics. We describe the three benchmarks and the metrics used to evaluate manipulation success and inference efficiency.

B) Per-Decision Optimization Details. We clarify the optimization unit, trajectory-level outcome signal, and group-relative update used to train the Adaptive Chunk Selector.

C) State-Dependent Feedback Evaluation. We use counterfactual horizon swaps and sequence shuffling to examine whether the selector learns when feedback is valuable, rather than simply preferring long chunks.

A DATASET AND EVALUATION METRICS

LIBERO is a widely adopted benchmark for general-purpose language-conditioned manipulation across diverse environments. It comprises Spatial, Object, Goal, and Long suites, for which we report suite-wise and average success rates.

VLABench evaluates generalization performance across in-distribution tasks, cross-category objects, commonsense requirements, semantic instructions, and unseen textures. We report the success rate for each setting and their average.

CALVIN evaluates long-horizon manipulation through five-task sequences. We report the cumulative success rate at each completion depth and average completed sequence length.

We report ChunkRL's inference speedups over the baseline on three benchmarks.

B PER-DECISION OPTIMIZATION DETAILS

Owing to the page limit of the main paper, we provide here additional clarification of the per-decision optimization details. ChunkRL optimizes the selector using the reward and objective in Eqs. (4)–(7) of the main paper. Both r_{epi} and r_{eff} are evaluated once after a complete rollout. Their sum therefore provides a trajectory-level outcome signal to every recorded horizon decision. This assignment captures the delayed effect of a feedback choice: selecting a horizon determines the open-loop interval and the state at which the policy next receives an observation.

The optimization unit is a feedback decision: an observation and a VLA call. For a rollout group from the same initial state, all recorded decisions are flattened and weighted equally. A trajectory containing more short-horizon decisions thus contributes more update terms, consistently accounting for its higher feedback frequency and greater policy-evaluation cost. The implementation uses the group-normalized advantage in Eq. (6), the sampled-horizon log-probability, and the entropy regularizer in Eq. (7), without an old-policy importance ratio, clipping objective, or KL term.

The component ablation in Table 3 of the main paper demonstrates the effect of this group-relative learning signal with fixed experts. Adding GRPO to the multi-horizon experts improves success from 98.6% to 99.4% and reduces inference time from 0.825 s to 0.790 s. These gains indicate that relative outcome feedback improves feedback-time selection beyond using multiple horizons alone.

Group normalization is particularly useful here because the alternatives in a rollout group begin from the same initial state. It therefore compares horizon decisions by their relative closed-loop outcomes, without requiring the selector to fit a separate absolute value model. In this way, the update can favor a shorter interval when additional feedback supports task completion and favor a longer interval when the same task can be completed with fewer policy evaluations.

Table 7: Probability-swap counterfactual evaluation. The unmodified column reports average selector probabilities over original decision states. The swapped column is their algebraic $P_8 \leftrightarrow P_{32}$ exchange, which specifies the post-softmax intervention applied before multinomial sampling at every counterfactual macro-decision; it is not re-averaged over counterfactual states. Each outcome cell reports success rate / measured episode-level inference time from newly executed rollouts. The final column reports the success-rate change in percentage points and relative speedup, computed as $t_{\text{unmodified}}/t_{\text{swapped}} - 1$.

Dataset	Average selector probability (%) $h = 8/16/32$		Success rate (%) Inference time (s)		Δ Success rate (pp) Relative speedup (%)
	Unmodified	Swapped	Unmodified	$P_8 \leftrightarrow P_{32}$	
LIBERO-Goal	53.2/35.9/10.9	10.9/35.9/53.2	98.6% / 0.867s	93.4% / 0.770s	-5.2 / $+12.6\%$
LIBERO-Object	55.4/23.8/20.8	20.8/23.8/55.4	99.6% / 0.622s	98.0% / 0.897s	-1.6 / -30.7%
LIBERO-Long	71.5/15.2/13.2	13.2/15.2/71.5	98.3% / 2.863s	83.3% / 1.673s	-15.0 / $+71.1\%$

Table 8: Per-trial outcomes and executed horizon frequencies under strong-ratio horizon sequence shuffle. The first block reports success rate / inference time; the second reports the frequency of consumed horizons in the order $h = 8/16/32$.

Suite	Success rate (%) / inference time (s)				
	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
LIBERO-Goal	96.0% / 1.48s	96.4% / 1.37s	94.0% / 1.42s	96.4% / 1.27s	96.8% / 1.24s
LIBERO-Object	97.8% / 1.25s	96.8% / 1.43s	97.2% / 1.32s	98.8% / 1.32s	98.4% / 1.13s
LIBERO-Long	87.5% / 3.03s	89.2% / 3.06s	89.2% / 2.58s	95.8% / 2.66s	88.3% / 2.75s
Suite	Executed horizon frequency (%), $h = 8/16/32$				
	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
LIBERO-Goal	53.7/39.7/6.7	53.4/34.2/12.4	48.8/41.8/9.4	46.0/41.9/12.1	41.2/49.3/9.6
LIBERO-Object	39.6/25.9/34.5	43.5/31.9/24.6	37.5/30.4/32.2	42.2/33.1/24.7	32.9/33.3/33.8
LIBERO-Long	53.6/21.0/25.5	52.6/22.6/24.8	46.7/25.0/28.3	54.5/25.0/20.4	52.4/26.1/21.5

C State-Dependent Feedback Evaluation

State-dependent feedback experiments use ChunkRL with fixed multi-horizon experts and horizon-dependent ALP. Apart from the intervention, inference settings remain unchanged.

The central purpose of the selector is to allocate dense feedback during task execution when timely and reliable correction is important and sparse feedback when additional observations are redundant. We evaluate this behavior with a post-softmax probability intervention and two discrete horizon-sequence controls. Under the learned policy, the average selector probability of the short horizon is 53.2%, 55.4%, and 71.5% on LIBERO-Goal, LIBERO-Object, and LIBERO-Long, respectively. The learned policy therefore does not collapse to the longest candidate horizon.

Probability Swap. At every macro-decision during each closed-loop rollout, the selector is evaluated normally and its post-softmax $P(h{=}8)$ and $P(h{=}32)$ entries are exchanged before multinomial sampling, while $P(h{=}16)$ remains unchanged. The swapped probability column in Table 7 is the algebraic $P_8 \leftrightarrow P_{32}$ exchange of the learned distribution averaged over the original decision states. It is a reference for the intervention, not a probability average recomputed over the later counterfactual states. The outcome columns are measured from counterfactual rollouts, whose state visitation and subsequently evaluated selector distributions may differ from the unmodified rollouts. This intervention preserves the selector architecture, task initialization, action experts, and horizon-dependent ALP while reversing its short–long preference. It decreases success by 5.2, 1.6, and 15.0 percentage points on Goal, Object, and Long. The effect is especially pronounced on LIBERO-Long, where decisions made at contact-sensitive or corrective stages affect later task completion.

The swap reduces policy calls per episode from 9.946 to 5.262 on Goal, from 9.400 to 6.288 on Object, and from 21.917 to 11.650 on Long. This corresponds to call-count reductions of 47.1%, 33.1%, and 46.8%, respectively. Episode-level inference time follows the trend on Goal and Long. The outcome on Object illustrates that policy-call count is not the only contributor to measured runtime: state visitation, episode length, and horizon-dependent per-call cost also matter.

Horizon Sequence Shuffle. We evaluate two complementary controls that bypass the selector during closed-loop rollout execution. *Original-Ratio Horizon Sequence Shuffle* reads the chosen-horizon list from the learned rollout log, forms a dataset-level multiset of the sampled horizons, and randomly permutes that multiset. *Strong-Ratio Horizon Sequence Shuffle*, reported in Table 8, first draws a seed-specific target ratio with a Dirichlet distribution, constructs a dataset-level multiset with the corresponding horizon counts, and then randomly permutes it. During either control, a global cursor directly supplies the next 8, 16, or 32 horizon; the selector does not compute logits or probabilities and does not sample a probability vector. If the source multiset is exhausted, it is reshuffled and reused. Consequently, the reported frequencies are computed from horizons actually consumed by the rollouts and can vary with episode termination.

Table 8 reports five independent strong-ratio trials; Trials 1–5 correspond, respectively, to seeds 20260729, 20260730, 20260731, 20260732, and 20260733. Across these trials, strong-ratio horizon sequence shuffle yields mean success rates of 95.9%, 97.8%, and 90.0% on Goal, Object, and Long, compared with 98.6%, 99.6%, and 98.3% under the learned schedule. The degradation is particularly severe on Long despite a mixed executed-horizon frequency.

These controls emphasize that chunk selection decides *when* feedback should be acquired. Executed horizon frequency summarizes how often each horizon is used but not which state receives it. A long interval at a contact-sensitive stage delays correction, whereas a short interval during stable motion adds redundant computation. Thus, matching a global horizon multiset while disrupting its state-dependent ordering is insufficient to recover the learned scheduler's accuracy–efficiency behavior.

The controls test complementary alternatives. Probability Swap modifies post-softmax selector distribution while retaining state-conditioned inference; Horizon Sequence Shuffle bypasses the selector and breaks the alignment between a discrete horizon sequence and execution states while controlling its global marginal composition. Their consistent degradation shows that ChunkRL benefits both from selecting an appropriate amount of feedback and from assigning that feedback to the appropriate stage of the rollout.

C.1 Fixed-Horizon Baseline Comparison

To isolate the benefit of adaptive feedback scheduling, we compare fixed-horizon policies with the learned selector using the same 0.5B backbone and evaluation protocol on LIBERO-Goal. As shown in Table 9, the adaptive $(8, 16, 32)$ policy achieves 98.6% success at 0.864 s. Among the fixed policies, $h = 8$ attains the highest success rate of 98.2% but requires 1.500 s; the adaptive policy improves success by 0.4 percentage points while reducing inference time by 42.4%. Compared with fixed $h = 16$, which has essentially the same inference time (0.863 s), the adaptive policy improves success by 1.1 percentage points. Fixed $h = 32$ and $h = 64$ further reduce reliability. We observe the same trend on the other benchmarks: adaptive feedback selection, together with ALP, improves success while markedly reducing forward-pass time. Table 9 also clarifies the roles of the experts and feedback intervention. Because no single fixed-horizon expert matches the adaptive policy in both success and inference cost, the gain does not arise from any one expert. Moreover, GRPO only learns a state-dependent preference over feedback horizons and does not inject additional task information into the model.

Table 9: Fixed- and adaptive-horizon comparison on LIBERO-Goal.

Policy	Success (%) ↑	Time (s) ↓
Fixed $h = 8$	98.2	1.500
Fixed $h = 16$	97.5	0.863
Fixed $h = 32$	94.5	**0.465**
Fixed $h = 64$	15.2	1.024
Adaptive $(8, 16, 32)$	**98.6**	0.864