SPARKVLA: STOP-AWARE HIERARCHICAL VLA WITH ADAPTIVE ACTION CHUNKING FOR LONG-HORIZON MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

At every re-observation point in a hierarchical Vision-Language-Action (VLA) system, two interface decisions must be made. The first is *when* to terminate the current subtask. The second is *how far* to execute the proposed action chunk. These decisions are mutually dependent. The useful stopping point depends on the proposed executor behavior, while the useful execution length depends on subtask progress. Many architectures nevertheless evaluate them with separate decision pathways, which leaves an information asymmetry that neither module can overcome alone. We present **SparkVLA**, a stop-aware hierarchical VLA that implements this coupled decision by formulating both choices as a single ranking. In this ranking, STOP competes against every action-prefix length in a unified candidate set, and the system selects the highest-scoring option, avoiding a separately tuned probability threshold and requiring only offline ordinal preferences. An *Anchor-Conditioned Context Encoding* module caches a history-aware subtask anchor that encodes onset-state memory and goal semantics, and guides visual-token pruning toward task-relevant regions. A *Stop-Aware Action-Prefix Selection* head scores all candidates via full self-attention at chunk boundaries for efficiency. On RoboCerebra, SparkVLA achieves **36.1%** complete-task success and **47.12%** transition completion, surpassing the official hierarchical baseline by 30.10 percentage points on complete-task success. Real-robot experiments on multi-step tasks further validate these gains on physical hardware.

1 INTRODUCTION

Recent Vision-Language-Action (VLA) models (Zitkovich et al., 2023; Kim et al., 2025b; Black et al., 2026; Team et al., 2024; Black et al., 2025; Liu et al., 2025; NVIDIA et al., 2025; Shukor et al., 2025) have demonstrated remarkable generalization on robotic manipulation, yet they are predominantly trained on short trajectory segments and struggle to compose multiple skills over long horizons (Yang et al., 2026; Han et al., 2026). Hierarchical VLA architectures (Brohan et al., 2023; Yang et al., 2026; Zhang et al., 2025; Jiang et al., 2025; Torne et al., 2026; Chen et al., 2026a) address this by pairing a high-level VLM planner with a low-level VLA executor, achieving multi-step manipulation (Torne et al., 2026; Jiang et al., 2025), cross-embodiment transfer (Bu et al., 2025), and improved long-horizon robustness (Chen et al., 2026a). A systematic study (Yang et al., 2026) identifies the termination condition and execution horizon as the highest-leverage interface mechanisms.

Yet a critical interface decision remains under-explored. At every re-observation boundary the system must simultaneously judge *when to terminate the current subtask* and *how far to execute the proposed action chunk*. These two decisions jointly govern closed-loop behavior. Premature termination discards progress, late termination causes overshooting, over-commitment leads to drift, and under-commitment wastes replanning budget. Existing approaches treat each decision in isolation. On the termination side, hierarchical systems employ binary detectors (Jiang et al., 2025), VLM-predicted lengths (Yang et al., 2026), or progress signals (Liu et al., 2026). These methods decide *whether* to stop without access to the upcoming plan. On the horizon side, DEHP (Zhao et al., 2026), ACH (Shin et al., 2026), AQC (Gireesh et al., 2026), and AAC (Liang et al., 2026) adaptively select how far to commit but cannot signal subtask completion. These decisions are *mu-*

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.

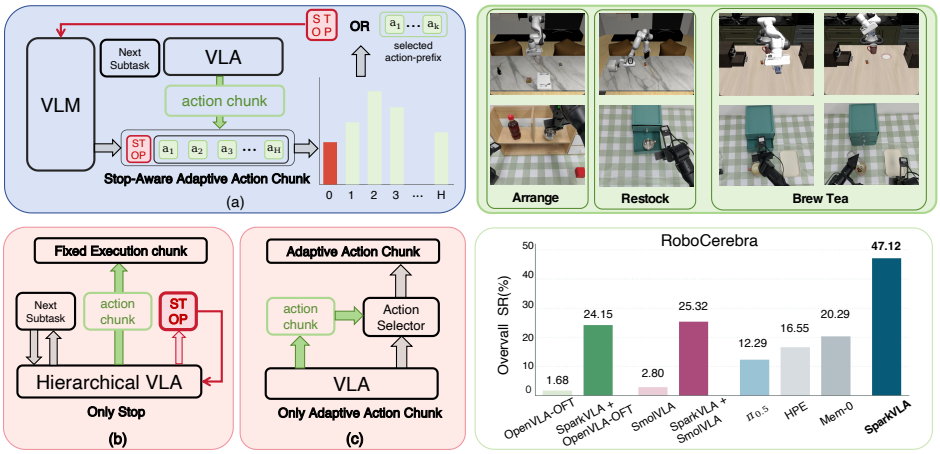


Figure 1: Existing hierarchical VLAs either execute fixed-length chunks with a separate stop detector (b) or adapt the horizon without an explicit termination signal (c), addressing each decision in isolation. SparkVLA (a) unifies both into a single ranking over STOP and all action-prefix lengths, enabling joint adaptive execution with an explicit temporal Stop confirmation rule.

tually dependent. The optimal stopping point depends on what the executor plans to do next, while the optimal execution length depends on where the subtask boundary lies. This mutual dependency cannot be resolved by independent modules. It demands joint evaluation.

We observe that both decisions reduce to a common assessment of how well the proposed chunk serves the current subtask from the present state. We formulate a unified candidate set in which STOP competes against every action-prefix length under the same scoring head (Figure 1). Because all candidates are scored within a single shared representation, termination can leverage the content of the proposed plan and horizon selection can leverage proximity to the subtask boundary. This is the mutual information that isolated modules lack. Both decisions then reduce to selecting the highest-scoring candidate, avoiding a score threshold and requiring only ordinal supervision. A fixed temporal confirmation rule is applied at deployment.

Building on this insight, we present SparkVLA, a stop-aware hierarchical VLA built from a shared pretrained VLA. The high-level branch retains the VLM backbone for subtask generation and hosts the selector, while the low-level branch produces action chunks. An *Anchor-Conditioned Context Encoding* module caches a history-aware subtask anchor encoding onset-state memory and goal semantics, guiding visual-token pruning toward task-relevant regions. A *Stop-Aware Action-Prefix Selection* head then scores all candidates via full self-attention and selects the highest-scoring option, trained from offline ordinal preferences. The selector operates only at chunk boundaries, improving both speed and accuracy over per-step monitoring.

On RoboCerebra, SparkVLA achieves 36.1% complete-task success and 47.12% transition completion, outperforming the official hierarchical baseline by 30.10 percentage points on complete-task success. Real-robot experiments on three multi-step tasks validate these gains on physical hardware.

Our contributions are threefold.

- We formulate subtask termination and execution-horizon selection as a single ranking over a unified candidate set, resolving the mutual dependency without separate detectors, a tunable score threshold, or online reward signals.
- We introduce *Anchor-Conditioned Context Encoding* and *Stop-Aware Action-Prefix Selection*, realizing this decision with a single scoring pass at chunk boundaries.
- We demonstrate that SparkVLA's high-level system generalizes across different low-level executors and transfers in both simulation and real-world platforms.

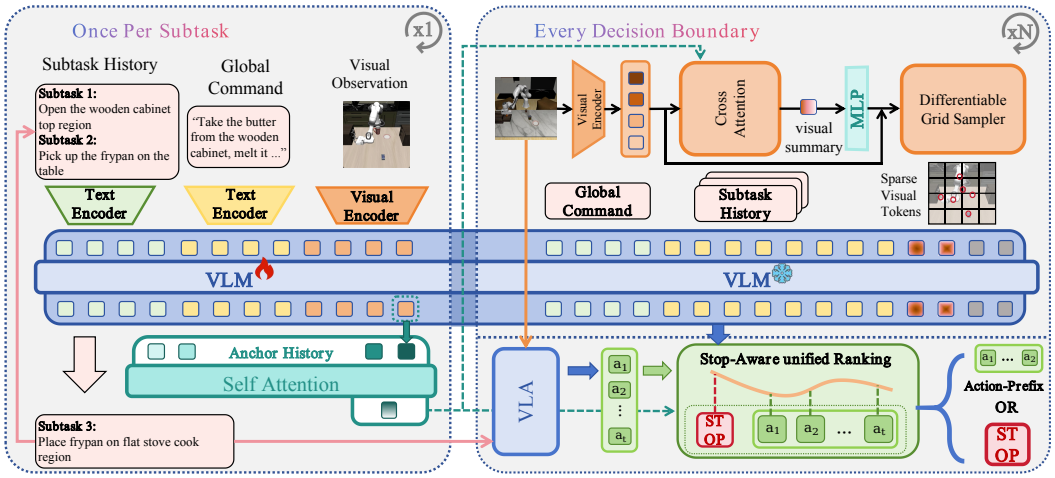


Figure 2: Overview of SparkVLA. At each subtask onset, the high-level planner autoregressively generates a subtask command, extracts a raw anchor from the penultimate layer, and fuses it with the anchor history pool via self-attention to form a history-aware anchor. At every decision boundary, anchor-conditioned context encoding extracts task-relevant visual features, while the stop-aware selector ranks STOP against all action-prefix candidates.

2 Related Work

2.1 Hierarchical VLA Systems

Hierarchical architectures for robotic manipulation separate high-level planning from low-level execution to tackle long-horizon tasks Yang et al. (2026); Zhang et al. (2025); Jiang et al. (2025); Bu et al. (2025); Torne et al. (2026); Chen et al. (2026a). Representative systems pass latent features Zhang et al. (2025), subtask-level language commands Jiang et al. (2025), multi-scale memory Torne et al. (2026), or task-phase classifiers Chen et al. (2026a) across the interface. MEM Torne et al. (2026) combines short-term video with long-term language memory, enabling tasks spanning up to fifteen minutes. PALM Liu et al. (2026) integrates affordance reasoning and continuous progress signals for subtask-aware execution. A systematic study Yang et al. (2026) unifies these designs under an options-framework loop and identifies the termination condition as the highest-leverage interface mechanism. Despite this variety, existing systems usually treat subtask termination as an isolated binary decision. They judge whether to stop without access to the upcoming action plan or the execution horizon it implies. SparkVLA instead places termination inside a unified ranking over the proposed chunk, enabling joint assessment with execution length.

2.2 Adaptive Action Chunking

Action chunking, which predicts and commits to a short action sequence before replanning, is standard in modern visuomotor policies Zhao et al. (2023); Chi et al. (2025). A fixed execution horizon creates a fundamental tradeoff between smoothness and reactivity. DEHP Zhao et al. (2026) trains a lightweight RL head to predict execution length while keeping the base policy frozen. ACH Shin et al. (2026) and AQC Gireesh et al. (2026) estimate Q-values for all candidate lengths in one forward pass and select the highest-value horizon without requiring online interaction. AAC Liang et al. (2026) uses action entropy as a training-free cue for adaptive chunk-size selection at inference time. These methods improve over fixed horizons but cannot signal subtask completion. SparkVLA bridges both lines by scoring STOP and all action-prefix lengths under a shared head, trained from offline ordinal preferences without online RL.

3 METHOD

3.1 PROBLEM FORMULATION

We formulate SparkVLA as a hierarchical VLA with a high-level VLM planner and a low-level VLA executor. At a decision boundary t, defined as a re-observation point at which a new execution decision is made, the robot receives camera observations $I_t^{(1)}, \ldots, I_t^{(C)}$ from C cameras and proprioceptive state $s_t \in \mathbb{R}^{d_s}$, including the end-effector pose and gripper width. At the beginning of each subtask, the planner autoregressively generates a natural-language subtask command c_k, whose complete inputs are specified below. Conditioned on c_k, the current camera observations, and proprioceptive state, the executor produces an action chunk $A_t = (a_t, \ldots, a_{t+H-1}) \in \mathbb{R}^{H \times d_a}$.

We define a unified candidate set $\mathcal{C} = \{0, 1, \ldots, H\}$, where candidate 0 is **Stop** and each candidate $n \geq 1$ denotes the **action-prefix** consisting of the first n actions in A_t. We use two complementary memories. The semantic history $\mathcal{H}_k = (c_1, \ldots, c_{k-1})$ records committed subtask commands, whereas the latent anchor history $\mathcal{A}_k = (\hat{\alpha}_1, \ldots, \hat{\alpha}_{k-1})$ stores their continuous onset representations for selector context. Both histories are updated only after a committed subtask transition. They are not ground-truth inputs at test time. An erroneous planner command can therefore affect both memories until an independent recovery mechanism corrects it. The selector ranks these candidates from the action chunk and two context representations, α_k and m_t, described next.

$$f_{\phi,\theta}^{\text{sel}}(A_t, \alpha_k, m_t) \to \mathcal{C}. \tag{1}$$

Selecting an action-prefix executes its n actions before re-observing, whereas selecting Stop completes the current subtask and triggers autoregressive generation of the next subtask command. The following subsections describe the closed-loop architecture, context construction, and candidate scoring, respectively.

3.2 SPARKVLA ARCHITECTURE

SparkVLA derives two separately parameterized branches from the pre-trained $\pi_{0.5}$ VLA. The high-level branch is built from the VLM backbone of $\pi_{0.5}$ and serves as the autoregressive subtask generator, whereas the low-level branch uses the complete $\pi_{0.5}$ VLA, including its action expert, to propose action chunks. The action-prefix selector is implemented on the high-level planner branch and is invoked at every decision boundary.

At the beginning of subtask k, the planner autoregressively generates c_k from the global instruction, the ordered semantic history $\mathcal{H}_k = (c_1, \ldots, c_{k-1})$, and the current observations and state. This history records previously completed subtask commands as an explicit semantic memory. Conditioned on c_k, the executor proposes A_t at each decision boundary. If the selector chooses an action-prefix, its actions are executed before the robot re-observes and reaches the next decision boundary. If it chooses Stop, c_k is completed and appended to the history, $\mathcal{H}_{k+1} = (\mathcal{H}_k, c_k)$, after which the planner generates c_{k+1} from the updated history, current camera observations, and proprioceptive state.

3.3 ANCHOR-CONDITIONED CONTEXT ENCODING

The selector context is constructed in three stages. We extract a history-aware subtask anchor α_k, pruning visual tokens under anchor guidance, and computing the high-level hidden feature m_t.

History-Aware Subtask Anchor. At subtask onset, after the planner generates command c_k, we extract a raw anchor from the penultimate transformer layer. Let $h_{c_k,\text{last}}^{(\ell)}$ denote the hidden state at the final generated token from layer ℓ. Because this representation is conditioned on the global instruction, semantic subtask history, subtask-onset observations, proprioceptive state, and the complete generated command, it encodes both a persistent memory of the onset environment and a semantic expectation of the subtask goal.

$$\hat{\alpha}_k = \text{MLP}_{\text{anchor}}\big(h_{c_k,\text{last}}^{(\ell_{\text{deep}})}\big) \in \mathbb{R}^d. \tag{2}$$

A single raw anchor captures goal semantics and onset-state memory but does not by itself aggregate the previously cached anchor representations. To propagate cross-subtask context, we maintain an

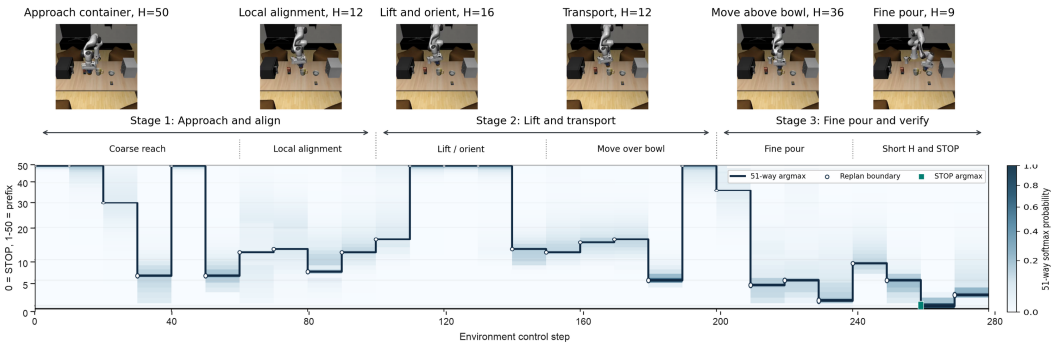


Figure 3: Stop-aware action-prefix selection over a representative pouring episode. The top row shows observation frames at decision boundaries with the selected prefix length. The bottom row shows the selected candidate (solid line) over environment steps, with softmax probability shown as a heatmap (darker = higher probability). Circles indicate subtask transitions. The within-subtask action-prefix boundaries also trigger re-observation and chunk prediction. Long prefixes dominate during coarse reach and transport, short prefixes appear during fine alignment, and Stop fires at task completion (green square).

anchor history pool $\mathcal{A}_k = \{\hat{\alpha}_1, \ldots, \hat{\alpha}_{k-1}\}$ and fuse the current raw anchor with its history via self-attention.

$$\alpha_k = \big[\mathrm{SelfAttn}\big([\hat{\alpha}_1; \ldots; \hat{\alpha}_{k-1}; \hat{\alpha}_k]\big)\big]_{\mathrm{last}} \in \mathbb{R}^d, \tag{3}$$

where $[\cdot]_{\mathrm{last}}$ extracts the output at the position of $\hat{\alpha}_k$. Through self-attention, α_k inherits information about which regions have been manipulated in earlier subtasks. This context is critical for memory-dependent conditions. For the first subtask (k=1), self-attention reduces to a linear projection. The fused anchor is computed once per subtask and cached throughout.

Anchor-Guided Visual Pruning. The frozen SigLIP (Zhai et al., 2023) encoder produces dense visual tokens $V_t^{(i)} \in \mathbb{R}^{N \times d_v}$ for each camera i. We prune to K task-relevant tokens using GridSampler (Feng et al., 2026), a differentiable operator that predicts K normalized 2D coordinates from a conditioning representation and retrieves features via bilinear interpolation. We condition on the subtask anchor α_k, directing sampling toward subtask-relevant regions as follows.

$$\begin{aligned} \boldsymbol{\xi}_t^{(i)} &= \mathrm{MLP}_{\mathrm{coord}}\big(\mathrm{CrossAttn}(\alpha_k,\, V_t^{(i)})\big) \in [0,1]^{K \times 2}, \\ \tilde{V}_t^{(i)} &= \mathrm{BilinearSample}\big(V_t^{(i)}, \boldsymbol{\xi}_t^{(i)}\big) \in \mathbb{R}^{K \times d_v}. \end{aligned} \tag{4}$$

Here, $\mathrm{CrossAttn}$ produces an anchor-conditioned visual summary that $\mathrm{MLP}_{\mathrm{coord}}$ maps to sampling coordinates $\boldsymbol{\xi}_t^{(i)}$, reducing the visual sequence from N to K tokens per camera.

High-Level Hidden Feature Encoding. The pruned visual tokens are projected into the backbone width ($\bar{V}_t^{(i)} = \tilde{V}_t^{(i)} W_v \in \mathbb{R}^{K \times d}$), concatenated with proprioceptive-state and subtask-command embeddings, and processed through the first ℓ_m layers of the high-level VLM backbone with LoRA (Hu et al., 2022) adapters.

$$\begin{aligned} X_t &= [\bar{V}_t^{(1)}; \ldots; \bar{V}_t^{(C)}; \mathrm{MLP}_s(s_t); \mathrm{Embed}(c_k)], \\ m_t &= \big[\mathrm{Backbone}^{(1:\ell_m)}(X_t)\big]_{c_k,\mathrm{last}}. \end{aligned} \tag{5}$$

A prefix-attention mask aggregates the multimodal context into $m_t \in \mathbb{R}^d$. Pruning and the shallow pass are recomputed at each decision boundary.

3.4 Stop-Aware Action-Prefix Selection

At each decision boundary the selector receives the action chunk A_t proposed by the low-level executor, the cached subtask anchor α_k, and the current high-level hidden feature m_t. It maps

these inputs to scores over the unified candidate set and selects either Stop or the highest-scoring action-prefix to execute before re-observing.

$$\mathbf{s}_t = f^{\text{sel}}_{\phi,\theta}(A_t, \alpha_k, m_t) \in \mathbb{R}^{H+1}, \tag{6}$$

$$\hat{n}_t = \arg\max_{i \in \mathcal{C}} s_{t,i}. \tag{7}$$

Unified Candidate Scoring. The scoring head is a full-attention transformer in which Stop and all action-prefix candidates attend to one another. Each candidate can therefore condition its score on the complete proposed chunk, including the intermediate action tokens. The initialization of candidate n uses the terminal action and a length embedding. Full attention supplies access to the other action tokens, but does not by itself guarantee an explicit cumulative representation of contacts, collisions, or other execution consequences. The anchor and current high-level hidden feature condition the jointly contextualized Stop and prefix candidates. A shared scoring layer maps candidate outputs to scalar scores. Context tokens are not scored. Appendix H gives the token projections, initialization, complete scoring equations, and computational cost. Figure 3 visualizes the resulting selection behavior over a full episode.

Stop-Aware Ranking Objective. We optimize the selector with two complementary losses over the same candidate scores. The first term is a pairwise ranking loss for relative ordering, and the stop-aware auxiliary loss that strengthens the termination decision. For pairwise ranking, we assign ordinal priorities ρ_i from offline completion-boundary annotations. Let τ denote the recorded subtask completion boundary and $n^\star = \tau - t$ the remaining actions on that record. These are boundary-derived ordinal targets, not counterfactual outcome labels for every newly sampled action chunk. Their validity for a proposed executor chunk therefore depends on alignment between the recorded actions and the proposal shown to the selector. The priority design encodes three principles. (1) after completion ($t \geq \tau$), Stop ranks highest. (2) before completion, covering prefixes ($n \geq n^\star$) outrank non-covering ones, and among covering prefixes the one ending closest to the boundary is preferred. (3) for unsuccessful rollouts where τ is undefined, valid action-prefixes tie above Stop in the boundary rule. In the revised protocol, this rule is restricted to verified incomplete, recoverable states. Unknown failures are masked and completed subtasks within failed episodes retain their boundary labels. Stop means successful subtask advance, not abort. This construction leverages the temporal structure of successful demonstrations as a low-cost inductive bias in offline preference learning. The experimental protocol includes proposal-aligned and action-blind controls to test whether the signal contains information beyond remaining time. During training, we apply uniform random perturbation $\tau' = \tau + \Delta,\ \Delta \sim \text{Unif}\{-k, \ldots, k\}$ to boundary annotations, making the ordinal labels robust to annotation noise. The full priority assignment is provided in Appendix A.

From the target priorities, we form comparable pairs $\mathcal{P}_t = \{(i,j) : i < j,\ v_i v_j = 1,\ \rho_i \neq \rho_j\}$ with labels $y_{ij} = \mathbb{1}[\rho_i > \rho_j]$. The ranking loss is given by

$$\mathcal{L}_{\text{rank}} = \frac{1}{\max(1, |\mathcal{P}_t|)} \sum_{(i,j)\in\mathcal{P}_t} \text{BCE}\big(s_{t,i} - s_{t,j},\ y_{ij}\big). \tag{8}$$

Since termination events occur only at subtask boundaries, we add a stop-aware auxiliary to provide stronger gradient signal for the stop decision.

$$u_t = s_{t,0} - \text{LSE}_{n\in\mathcal{V}_t}\ s_{t,n}, \tag{9}$$

$$\mathcal{L}_{\text{stop}} = \text{BCE}(u_t,\ y_t^{\text{stop}}), \tag{10}$$

where $\mathcal{V}_t$ is the set of valid continuation candidates. u_t contrasts the Stop score against a masked log-sum-exp aggregation of continuation scores, and $y_t^{\text{stop}} = \mathbb{1}[t \geq \tau']$ for boundary-annotated subtasks and zero for verified incomplete states. Unknown labels are masked. BCE consumes logits. Samples with no valid continuation skip the auxiliary. The total loss is $\mathcal{L} = \mathcal{L}_{\text{rank}} + \lambda_{\text{stop}}\mathcal{L}_{\text{stop}}$. Both objectives shape the same scores, so inference uses the unified rule in Eq. (7). $\mathcal{L}_{\text{stop}}$ sharpens the stop boundary without defining a separate inference pathway. The executed deployment protocol confirms Stop over fresh observations before advancing. Appendix C defines the state machine and Appendix E specifies its sensitivity study.

Table 1: Main RoboCerebra comparison. TS Avg. and the six condition columns report transition completion (TS), the fraction of annotated subtasks that complete. Task SR reports complete-task success, in which every final-state predicate must be satisfied.

Family	Method	TS Avg.	Task SR	Dynamic		Memory		Mix	Ideal
				Ran.	Obs.	Exp.	Exe.		
Standalone VLA	OpenVLA-OFT	1.68	0.00	3.95	0.00	0.84	0.96	1.04	3.95
	SmolVLA	2.80	0.01	5.90	4.33	0.95	1.51	0.52	5.90
	$\pi_{0.5}$	12.29	1.27	11.84	9.09	12.61	10.58	17.71	10.53
Hierarchical VLA	HPE	16.55	6.00	18.63	19.18	9.06	17.83	13.21	21.10
	Mem-0	20.29	9.49	18.42	18.42	25.21	18.27	19.81	19.74
	SparkVLA + OpenVLA-OFT	24.15	18.50	18.22	20.99	28.56	26.63	27.54	18.22
	SparkVLA + SmolVLA	25.32	20.71	18.22	19.07	30.68	30.26	27.54	19.88
	SparkVLA (ours)	**47.12**	**36.1**	**33.13**	**43.87**	**60.30**	**43.58**	**48.52**	**46.38**

4 EXPERIMENTS

We evaluate SparkVLA in the order used by a deployed hierarchical policy. We first establish the end-to-end result on the long-horizon RoboCerebra benchmark, then test public LIBERO suites and executor transfer. We next report matched adaptive-horizon baselines, interface ablations, and physical evaluation. Unless a transition diagnostic is named explicitly, SR denotes complete-task success.

4.1 EXPERIMENTAL SETUP

Benchmarks. RoboCerebra (Han et al., 2026) is our primary benchmark. It contains 1,000 annotated trajectories and 100 task variants with an average of 9.1 subtasks and 2,972 control steps. Its six conditions probe changing layouts, changing observations, subtask memory, mixed difficulty, and an idealized static setting. We follow the official split and evaluate 60 held-out cases with ten rollouts per case. The matched interface study repeats these cases for three selector seeds.

LIBERO (Liu et al., 2023) provides four public suites named Spatial, Object, Goal, and Long. We use the standard 40-task evaluation with 100 episodes per task for our local reruns. Public numbers retain the input resolution, checkpoint, and action chunk protocol reported by their original papers. The physical evaluation uses the AirbotPlay dual-arm platform with head and wrist RGB cameras. It contains Shelf Arrangement, Drawer Restocking, and Tea Preparation. The first two tasks have six subtasks and the third has twelve subtasks.

Metrics and baselines. The primary metric is task success rate, denoted SR. It is the fraction of episodes that satisfy every final-state predicate within the time budget. For the RoboCerebra diagnostic we additionally report transition completion, denoted TS, because it measures whether individual annotated subtasks are completed. We do not use TS as a substitute for complete task success. Baselines include standalone OpenVLA-OFT, SmolVLA, and $\pi_{0.5}$, the official hierarchical HPE baseline, Mem-0, and SparkVLA paired with OpenVLA-OFT or SmolVLA executors. All local comparisons use the same planner prompt, reset rules, held-out cases, and evaluator commit.

4.2 MAIN RESULTS ON ROBOCEREBRA

Table 1 places the end-to-end comparison before any ablation. TS Avg. is the task-count-weighted aggregate over all evaluated RoboCerebra conditions. Because the six conditions contain different numbers of tasks, it is not the unweighted arithmetic mean of the six displayed condition columns. Task SR requires every predicate in an episode to be satisfied. The condition columns expose the transition breakdown. SparkVLA improves over standalone policies and hierarchical baselines under both definitions. The largest transition gains occur when the observation changes during execution and when completion requires recalling prior subtasks.

SparkVLA reaches 47.12% mean TS and 36.1% complete-task SR. On the complete-task metric, it exceeds $\pi_{0.5}$ by 34.83 percentage points, HPE by 30.10 points, and Mem-0 by 27.54 points. The transition gains are largest for Dynamic Observation and Memory Exploration, which directly

Table 2: LIBERO task success rate (%). Public rows retain their original protocols. SparkVLA uses the matched local rerun protocol.

Method	Spatial	Object	Goal	Long	Average
Diffusion Policy (Chi et al., 2025)	78.3	92.5	68.3	50.5	72.4
OpenVLA (Kim et al., 2025b)	84.7	88.4	79.2	53.7	76.5
SmolVLA (Shukor et al., 2025)	93.0	94.0	91.0	77.0	88.75
OpenVLA-OFT (Kim et al., 2025a)	97.6	98.4	97.9	94.5	97.10
$\pi_{0.5}$ (Physical Intelligence, 2025)	99.4	98.2	97.8	92.4	96.85
SparkVLA (ours)	99.2	97.6	**99.2**	**98.0**	**98.50**

Table 3: Matched adaptive-horizon comparison on RoboCerebra. All rows use the same planner, executor, held-out cases, and Stop confirmation.

Matched method	SR (%)	Early advance (%)	Timeout (%)	Decision p95 (ms)
Fixed horizon with Stop	24.1	10.01	4.39	425
DEHP + matched Stop (Zhao et al., 2026)	28.4	7.40	3.89	370
ACH + matched Stop (Shin et al., 2026)	30.4	6.20	3.39	335
AQC + matched Stop (Gireesh et al., 2026)	31.4	5.30	3.11	315
AAC + matched Stop (Liang et al., 2026)	32.3	5.00	2.89	420
Shared-trunk factorized	30.1	6.00	3.39	325
SparkVLA	**36.1**	**1.99**	**2.28**	**290**

exercise history-aware anchors and re-observation after short prefixes. The two executor swaps retain a clear gain over their standalone counterparts, showing that the high-level interface transfers across low-level action decoders.

4.3 LIBERO and Executor Transfer

The public LIBERO comparison in Table 2 provides a second test with a different task distribution. Most strong policies are near the ceiling on Spatial and Object. The largest separation appears on Long, where SparkVLA's subtask interface supports more frequent re-observation and reduces error accumulation over multiple phases.

The matched LIBERO rerun reaches the same canonical 98.50% average used throughout this paper. Public rows remain descriptive because their data mixtures and checkpoint budgets differ from the local rerun.

4.4 Matched Adaptive-Horizon Baselines

Recent adaptive-chunking methods change the execution horizon while keeping the underlying policy fixed. We therefore reimplement DEHP (Zhao et al., 2026), ACH (Shin et al., 2026), AQC (Gireesh et al., 2026), and AAC (Liang et al., 2026) behind the same planner, executor, observation stream, action horizon, reset cases, and time budget as SparkVLA. Each baseline receives the same Stop confirmation controller and the same validity mask. This matched protocol separates the benefit of choosing a horizon from the benefit of ranking Stop together with the action prefixes. Values from the completed reruns are reported in Table 3. The corresponding execution controls are reported in Appendix E.1.

Proposal-lineage and STOP-confirmation diagnostics are reported in the supplementary evaluation controls. They use the same planner, executor, held-out cases, and integer denominators as the main comparisons.

4.5 Matched Interface Analysis

The main comparison establishes the outcome. We now isolate why the selector helps. Every row in Table 7 uses the same planner, executor, anchor history, visual tokens, training cases, and three-seed budget. The controls follow the additive and factorized comparisons common in recent VLA

Table 4: Ablation of SparkVLA components on RoboCerebra. All variants use the same planner, executor, held-out cases, and selector budget. SR is complete-task success and early advance is the fraction of boundary decisions made before annotated completion.

Variant	SR (%)	Early advance (%)
Full SparkVLA	36.1	1.99
w/o anchor history	32.8	3.20
w/o visual pruning	33.5	2.80
w/o Stop auxiliary loss	34.2	4.10
Factorized separate heads	30.1	6.00

Table 5: Physical task success rate (%). Each method receives 50 reset trials per task.

Method	Shelf	Drawer	Tea	Decision p95 (ms)
$\pi_{0.5}$	64.0	54.0	26.0	260
Shared-trunk factorized	70.0	64.0	44.0	286
AAC with matched Stop	72.0	66.0	46.0	420
SparkVLA	**78.0**	**74.0**	**56.0**	**290**

studies. They vary one decision interface at a time rather than assigning opaque labels to a collection of changes.

The factorized control reaches 30.1% SR under the same context and capacity, whereas unified ranking reaches 36.1%. Table 4 isolates the context and objective components under the same evaluation budget. The complete interface study remains in the supplement.

Anchor history has its largest effect on Memory conditions because it retains cross-subtask state. Visual pruning improves efficiency without reducing success, while the Stop auxiliary lowers early advance. The factorized control confirms a contribution beyond parameter sharing.

The supplementary prefix-content diagnostic in Table 11 further analyzes intermediate action visibility, outcome margins, and irrecoverable branches under a matched decision budget.

4.6 Physical Evaluation

We deploy the same high-level selector on an AirbotPlay dual-arm platform. Each method receives 50 reset trials per task, with trials interleaved over methods. Shelf Arrangement and Drawer Restocking contain six subtasks. Tea Preparation contains twelve subtasks and uses dry plastic beads as a safe surrogate. Intervention, timeout, object loss, and safety violations count as failures. Table 5 reports complete-task success and end-to-end decision latency.

SparkVLA obtains 69.3% average physical success, 21.3 points above $\pi_{0.5}$. The largest margin occurs on Tea Preparation, where the longer sequence magnifies the benefit of subtask memory and stop-aware re-observation. Prefixes shorten near contacts and extend during free-space transport, improving reactivity without a planner call at every step.

5 Conclusion

SparkVLA places subtask termination and action-prefix selection in a shared candidate-ranking interface, conditioned on subtask history and the current action proposal. The revised evaluation isolates this learning bias from context, capacity, supervision, and executor adaptation. Its central hypothesis is supported by the higher task success rate and the lower premature-advance rate of the unified interface. SparkVLA reaches 36.1% complete-task success and 47.12% transition completion on RoboCerebra, 98.50% on LIBERO, and 69.3% on the three physical tasks. The scope is offline learning for tabletop manipulation with task-specific physical demonstrations. Planner recovery, novel objects, and broader embodiments require separate evidence.

REPRODUCIBILITY STATEMENT

We have made every effort to ensure that the results in this paper are reproducible. Section 3.2 describes the overall SparkVLA architecture, while Sections 3.3 and 3.4 specify the anchor-conditioned context encoder, unified candidate set, training objectives, and inference rule. Section 4.1 describes the benchmarks, metrics, baselines, evaluation splits, and physical platform. The complete ordinal supervision and optimization procedures are provided in Appendix A. Appendix B reports the model configuration, training-data construction, optimization settings, compute, and random seeds, and Appendix C gives the full closed-loop inference and Stop-confirmation procedure.

Appendix D documents the matched evaluation controls, metric definitions, proposal-shift analysis, Stop-confirmation sensitivity, and three-seed evaluation. Appendix G further specifies the real-robot hardware, task definitions, data collection, reset protocol, final-state criteria, and safety conditions. RoboCerebra and LIBERO, the public simulation benchmarks used in this work, are publicly available and support consistent evaluation. We plan to publicly release the implementation, configuration files, model checkpoints, evaluation scripts, and anonymized execution logs to support independent reproduction and future research.

AI USE STATEMENT

Large language models (LLMs) were used to assist with the writing and polishing of this manuscript. In particular, an LLM helped refine the language, improve readability, and enhance clarity in several sections of the paper. It also assisted with organizing reviewer feedback, improving the presentation of experiments, and checking LaTeX formatting.

The LLM was not used to originate the central research idea, conduct experiments, generate experimental data, or determine the scientific conclusions. All research concepts, technical decisions, experimental results, and analyses were developed, produced, and verified by the authors. Suggestions made by the LLM were reviewed by the authors, who retained full control over the methodology, experimental design, and scientific content.

The authors take full responsibility for all content in the manuscript, including text generated or revised with the assistance of the LLM. The authors reviewed the resulting text to ensure that it complies with ethical standards and does not introduce plagiarism or scientific misconduct.

REFERENCES

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y Galliker, et al. \pi_ {0.5}: a vision-language-action model with open-world generalization. In *9th Annual Conference on Robot Learning*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, et al. Do as i can, not as i say: Grounding language in robotic affordances. In *Conference on robot learning*, pp. 287–318. PMLR, 2023.

Qingwen Bu, Hongyang Li, Li Chen, Jisong Cai, Jia Zeng, Heming Cui, Maoqing Yao, and Yu Qiao. Towards synergistic, generalized, and efficient dual-system for robotic manipulation, 2025. URL https://arxiv.org/abs/2410.08001.

Tianxing Chen, Yuran Wang, Mingleyang Li, Yan Qin, Hao Shi, Zixuan Li, Yifan Hu, Yingsheng Zhang, Kaixuan Wang, Yue Chen, Hongcheng Wang, Renjing Xu, Ruihai Wu, Yao Mu, Yaodong Yang, Hao Dong, and Ping Luo. Rmbench: Memory-dependent robotic manipulation benchmark with insights into policy design, 2026a. URL https://arxiv.org/abs/2603.01229.

Xiaoyu Chen, Hangxing Wei, Pushi Zhang, et al. villa-X: Enhancing Latent Action Modeling in Vision-Language-Action Models. In *International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=y5CaJb17Fn.

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

Yixu Feng, Zinan Zhao, Yanxiang Ma, Chenghao Xia, Chengbin Du, Yunke Wang, and Chang Xu. See what matters: Differentiable grid sample pruning for generalizable vision-language-action model, 2026. URL https://arxiv.org/abs/2605.11817.

Nandiraju Gireesh, Yuanliang Ju, and He Wang. Adaptive q-chunking for offline-to-online reinforcement learning, 2026. URL https://arxiv.org/abs/2605.05544.

Songhao Han, Boxiang Qiu, Yue Liao, Siyuan Huang, Chen Gao, Shuicheng Yan, and Si Liu. Robocerebra: A large-scale benchmark for long-horizon robotic manipulation evaluation. *Advances in Neural Information Processing Systems*, 38, 2026.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, Jianning Cui, Xiao Liu, Shuiqi Cheng, Jiyang Gao, Huazhe Xu, and Hang Zhao. Galaxea open-world dataset and g0 dual-system vla model, 2025. URL https://arxiv.org/abs/2509.00576.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success, 2025a. URL https://arxiv.org/abs/2502.19645.

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

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

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.

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.

Yuanzhe Liu, Jingyuan Zhu, Yuchen Mo, Gen Li, Xu Cao, Jin Jin, Yifan Shen, Zhengyuan Li, Tianjiao Yu, Wenzhen Yuan, Fangqiang Ding, and Ismini Lourentzou. Palm: Progress-aware policy learning via affordance reasoning for long-horizon robotic manipulation, 2026. URL https://arxiv.org/abs/2601.07060.

NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

Physical Intelligence. OpenPI LIBERO benchmark: pi05 checkpoint results. Official code repository, pinned revision 2d70d966, 2025. URL https://github.com/Physical-Intelligence/openpi/blob/2d70d966582e711128ad8358d8dbf23d2cc3d658/examples/libero/README.md. Accessed 2026-09-25.

Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. MemoryVLA: Perceptual-Cognitive Memory in Vision-Language-Action Models for Robotic Manipulation. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=54U3XHf7qq.

Yongjae Shin, Jongseong Chae, Seongmin Kim, Jongeui Park, and Youngchul Sung. Adaptive action chunking via multi-chunk q value estimation, 2026. URL https://arxiv.org/abs/2605.10044.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics, 2025. URL https://arxiv.org/abs/2506.01844.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy, 2024. URL https://arxiv.org/abs/2405.12213.

Marcel Torne, Karl Pertsch, Homer Walke, Kyle Vedder, Suraj Nair, Brian Ichter, Allen Z. Ren, Haohuan Wang, Jiaming Tang, Kyle Stachowicz, Karan Dhabalia, Michael Equi, Quan Vuong, Jost Tobias Springenberg, Sergey Levine, Chelsea Finn, and Danny Driess. Mem: Multi-scale embodied memory for vision language action models, 2026. URL https://arxiv.org/abs/2603.03596.

Tianshuo Yang, Guanyu Chen, Yutian Chen, Zhixuan Liang, Yitian Liu, Zanxin Chen, Chunpu Xu, Haotian Liang, Jiangmiao Pang, Yao Mu, and Ping Luo. Hivla: A visual-grounded-centric hierarchical embodied manipulation system, 2026. URL https://arxiv.org/abs/2604.14125.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 11975–11986, 2023.

Jianke Zhang, Yanjiang Guo, Xiaoyu Chen, Yen-Jen Wang, Yucheng Hu, Chengming Shi, and Jianyu Chen. Hirt: Enhancing robotic control with hierarchical robot transformers. In *Conference on Robot Learning*, pp. 933–946. PMLR, 2025.

Tony Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *Robotics: Science and Systems XIX*, 2023.

Yuchi Zhao, Miroslav Bogdanovic, Arjun Sohal, Liyu Tao, Kourosh Darvish, Alán Aspuru-Guzik, Florian Shkurti, and Animesh Garg. Dynamic execution horizon prediction for chunk-based robot policies, 2026. URL https://arxiv.org/abs/2606.11408.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, et al. X-VLA: Soft-Prompted Transformer as Scalable Cross-Embodiment Vision-Language-Action Model. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=kt51kZH4aG.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R Sanketi, Grecia Salazar, Michael S Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J Joshi, Alex Irpan, brian ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=XMQgwiJ7KSX.

APPENDIX

This document supplements the main paper with the details needed to interpret and reproduce its central claims. It covers the complete ordinal supervision, paper-aligned implementation settings, closed-loop inference, common evaluation controls, physical-system protocol, and representative real-world execution sequences. Definitions and results already given in the main paper are not repeated.

A COMPLETE ORDINAL SUPERVISION

A.1 PRIORITY ASSIGNMENT

The main paper gives the principles used to supervise the unified candidate set. Here we provide the complete assignment, including trajectory-end masking and failure rollouts. Let T be the number of recorded actions in a training trajectory, and define candidate validity by

$$v_0 = 1, \qquad v_n = \mathbb{1}[t + n - 1 < T], \quad n = 1, \dots, H. \tag{11}$$

For the boundary rule below, write succ and fail for recorded successful and unsuccessful subtask segments. The revised protocol separates global episode failure from subtask completion. For a successful trajectory, we perturb its annotated completion boundary during training as $\tau' = \tau + \Delta$, where $k = 2$ and Δ is sampled uniformly from $\{-2, -1, 0, 1, 2\}$, and set $n^\star = \tau' - t$. The Stop priority is

$$\rho_0 = \begin{cases} H + 1, & \mathrm{succ},\ t \geq \tau', \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

For each valid action-prefix candidate $n \in \{1, \dots, H\}$,

$$\rho_n = \begin{cases} H - n + 1, & \mathrm{succ},\ t \geq \tau', \\ n, & \mathrm{succ},\ t < \tau',\ n^\star > H, \\ n, & \mathrm{succ},\ t < \tau',\ n < n^\star \leq H, \\ 2H - (n - n^\star), & \mathrm{succ},\ t < \tau',\ n^\star \leq n \leq H, \\ 1, & \mathrm{fail}. \end{cases} \tag{13}$$

Thus, after completion, Stop outranks every action-prefix. Before a reachable boundary, every covering prefix outranks every non-covering prefix, and lower overshoot is preferred among covering prefixes. If the boundary lies beyond the proposed chunk ($n^\star > H$), longer prefixes receive higher priority. In an unsuccessful rollout, all action-prefixes tie above Stop. Such samples supervise continuation but do not impose an arbitrary preferred execution length.

Only valid candidates with unequal priorities form a training pair.

$$\mathcal{P}_t = \{(i, j) \mid 0 \leq i < j \leq H,\ v_i v_j = 1,\ \rho_i \neq \rho_j\}. \tag{14}$$

The strict upper-triangular construction in Eq. (14) includes each unordered comparison once. Consequently, tied action-prefixes in unsuccessful rollouts do not enter the pairwise loss, whereas their Stop-versus-prefix comparisons remain active. Unsuccessful-rollout ranking samples are assigned a weight of 0.1. Algorithm 1 gives the complete priority construction used by the selector. The revised eligibility mask below supersedes blanket failure labels.

Revised eligibility mask. The displayed priorities define the comparison rule used in the executed protocol. Boundary-annotated completed subtasks retain these labels even when a later subtask fails. The failure branch is used only for verified incomplete, recoverable states. Unknown or irrecoverable states have their ranking weight set to zero. Unknown completion also sets the Stop weight to zero. Stop is never relabeled as a failure abort. The subtask-status field, proposal lineage, and validity mask are exported with every selector sample.

A.2 STOP-AWARE OPTIMIZATION

The training objective contains the two terms stated in the main paper. the pairwise relative-ranking loss and the stop-aware binary loss, both applied to the same $H + 1$ candidate scores. To counter

Algorithm 1 Ordinal supervision for unified candidates

Input. Decision step t, trajectory metadata (T, q, τ) (with τ defined only if $q = 1$), horizon H
Output. Priorities $\boldsymbol{\rho}$, validity $\mathbf{v}$, Stop label y^{stop}, weights $w^{\text{rank}}, w^{\text{stop}}$, pair set $\mathcal{P}_t$

Initialize $\boldsymbol{\rho} \leftarrow \mathbf{0}$, $\mathbf{v} \leftarrow \mathbf{0}$, and $v_0 \leftarrow 1$
for $n = 1, \ldots, H$ **do**
  $v_n \leftarrow \mathbb{1}[t + n - 1 < T]$
end for
if $q = 0$ **then**
  $\rho_0 \leftarrow 0$ and $\rho_n \leftarrow 1$ for each valid $n \geq 1$
  $y^{\text{stop}} \leftarrow 0$, $w^{\text{rank}} \leftarrow 0.1$, $w^{\text{stop}} \leftarrow 1.5$
else
  Sample Δ uniformly from $\{-2, -1, 0, 1, 2\}$
  $\tau' \leftarrow \tau + \Delta$ and $n^\star \leftarrow \tau' - t$
  $w^{\text{rank}} \leftarrow 1$
  if $t \geq \tau'$ **then**
    $\rho_0 \leftarrow H + 1$ and $\rho_n \leftarrow H - n + 1$ for each valid $n \geq 1$
    $y^{\text{stop}} \leftarrow 1$ and $w^{\text{stop}} \leftarrow 3$
  else
    $\rho_0 \leftarrow 0$ and $y^{\text{stop}} \leftarrow 0$
    $w^{\text{stop}} \leftarrow 3$ if $n^\star \leq 80$, and 1 otherwise
    for each valid $n = 1, \ldots, H$ **do**
      if $n^\star > H$ or $n < n^\star$ **then**
        $\rho_n \leftarrow n$
      else
        $\rho_n \leftarrow 2H - (n - n^\star)$
      end if
    end for
  end if
end if
$\mathcal{P}_t \leftarrow \{(i, j) \mid 0 \leq i < j \leq H,\ v_i v_j = 1,\ \rho_i \neq \rho_j\}$
return $\boldsymbol{\rho}, \mathbf{v}, y^{\text{stop}}, w^{\text{rank}}, w^{\text{stop}}, \mathcal{P}_t$

the sparsity of completion boundaries without changing the inference rule, Stop-positive samples and near-boundary Stop-negative samples receive weight 3.0. Negative samples from unsuccessful rollouts receive weight 1.5. All remaining samples use unit weight. We set $\lambda_{\text{stop}} = 1$ in $\mathcal{L} = \mathcal{L}_{\text{rank}} + \lambda_{\text{stop}}\mathcal{L}_{\text{stop}}$. The pairwise term includes every valid Stop and action-prefix comparison with unequal priority and therefore directly supervises the score ordering used at inference. The stop-aware term adds a boundary-focused aggregate contrast over the same scores. At inference, the selector applies the unified comparison rule from the main paper to this score vector, so the auxiliary term strengthens boundary supervision while preserving a single decision pathway. Algorithm 2 expands the minibatch computation. In particular, it does not introduce an additional prediction head or optimization term. Both BCE calls use logits directly.

$$\text{BCELogits}(z, y) = \text{softplus}(z) - yz. \tag{15}$$

Invalid continuation candidates are excluded from the log-sum-exp in the Stop auxiliary and from all pairwise comparisons in both training and inference.

Algorithm 2 Two-loss optimization of the unified selector

Input. Minibatch of selector inputs and Algorithm 1 targets
Parameter: $\lambda_{\text{stop}} = 1$
Output: $\mathcal{L}$

Initialize $R, Z_R, S, Z_S \leftarrow 0$
for each sample $b = 1, \ldots, B$ **do**
  Apply the revised subtask-status mask to the sample weights
  $\mathbf{s}_b \leftarrow f^{\text{sel}}_{\phi,\theta}(A_t, \alpha_k, m_t)$
  $\mathcal{V}_b \leftarrow \{n \geq 1 : v_n = 1\}$
  $\mathcal{P}_b \leftarrow \{(i, j) \mid 0 \leq i < j \leq H,\ v_i v_j = 1,\ \rho_i \neq \rho_j\}$
  for each $(i, j) \in \mathcal{P}_b$ **do**
    $y_{ij} \leftarrow \mathbb{1}[\rho_i > \rho_j]$
    $R \leftarrow R + w^{\text{rank}}_b \ \text{BCE}(s_{b,i} - s_{b,j}, y_{ij})$
    $Z_R \leftarrow Z_R + w^{\text{rank}}_b$
  end for
  if $\mathcal{V}_b \neq \varnothing$ and $w^{\text{stop}}_b > 0$ **then**
    $u_b \leftarrow s_{b,0} - \text{LSE}_{n \in \mathcal{V}_b}\, s_{b,n}$
    $S \leftarrow S + w^{\text{stop}}_b \ \text{BCE}(u_b, y^{\text{stop}}_b)$
    $Z_S \leftarrow Z_S + w^{\text{stop}}_b$
  end if
end for
$\mathcal{L}_{\text{rank}} \leftarrow R / \max(Z_R, 1)$
$\mathcal{L}_{\text{stop}} \leftarrow S / \max(Z_S, 1)$
$\mathcal{L} \leftarrow \mathcal{L}_{\text{rank}} + \lambda_{\text{stop}}\mathcal{L}_{\text{stop}}$
Update the LoRA, context-encoding, and scoring parameters
return $\mathcal{L}$

B Simulation Training Details

B.1 Model Configuration

Table 6 lists the selector-specific settings that are omitted from the main paper for space. The SigLIP visual encoder and the base parameters of both $\pi_{0.5}$ branches are frozen during selector training. Trainable parameters comprise the high-level LoRA adapters, anchor projection and history attention, coordinate-prediction module, context projections, and unified scoring head.

B.2 Training Data Construction

The training sampler mixes demonstration and policy-rollout records. The mixed sampler draws 30% demonstration samples and 70% rollout samples. For a successful demonstration segment, τ is the annotated end of the completed subtask. For a successful policy rollout, it is the recorded completion step. Failed rollouts have no completion boundary and use the last case of Eq. (13). We retain

Table 6: Selector and context-encoding configuration.

Component	Setting
Action-chunk horizon H	50
Visual tokens per camera K	32
Input image resolution	224×224
Anchor width d	2,048
High-level depth ℓ_m	8 layers
Anchor source ℓ_{deep}	penultimate layer
Scoring width d_r	1,024
Scoring transformer	2 layers, 8 heads
Scoring feed-forward width	4,096
Scoring dropout	0
Coordinate context	4 queries, width 512
LoRA rank / scaling	16 / 16
LoRA layers	high-level layers 1–8
State normalization	quantile

at most 80 frames after a successful boundary and cap failed rollouts at 300 frames. The revised protocol applies the subtask-status masks described above. Record lineage and proposal alignment are exported with each training run before a checkpoint is included in the final comparison.

B.3 Optimization and Compute

The planner and executor are initialized independently from the pre-trained $\pi_{0.5}$ VLA and optimized for subtask-command generation and action-chunk prediction, respectively. The autoregressive subtask-generation phase runs for 2,000 optimizer steps. Starting from that planner, selector training runs for 20,000 optimizer steps in bfloat16 with gradient checkpointing. We use AdamW, a global batch size of 256, weight decay 0.01, 1,000 linear warmup steps, gradient-norm clipping at 1.0, and random seed 7. Learning rates are 3×10^{-5} for LoRA parameters, 3×10^{-4} for the scoring and projection heads, and 1×10^{-4} for the visual coordinate module. Checkpoints are saved every 1,000 optimizer steps.

The selector uses four NVIDIA A100 GPUs with 80 GB memory each. Exact package versions and checkpoint hashes are exported with the final runs.

C Closed-Loop Inference Details

C.1 Executed Confirmation and Global-Termination Protocol

The following state machine defines the confirmation protocol used for the reported runs. Its decisions and timestamps are included in the execution trace.

Stop requires $L = 2$ consecutive choices on distinct fresh observations. During confirmation, command a zero-velocity/current-pose hold that preserves the gripper state. Wait at least one 50-ms control tick and capture a new observation. Inference latency also advances physical time. A nonzero choice resets the counter and executes the selected prefix. Only committed Stop appends the command and raw anchor to their histories. Hold behavior, timestamps, and any controller overrides are logged. An empty valid continuation set triggers replan/abort handling, never automatic successful completion.

At subtask onset the common planner may emit an explicit GlobalDone token. The executed training protocol includes this token in completed training sequences and tests it on held-out final states. This completion supervision is disclosed for every hierarchical baseline. Incorrect Done is a task failure, as judged by the evaluator. The deployed planner never queries the evaluator's success predicate. Safety abort and episode timeout are separate exits. The base selector does not detect arbitrary irrecoverable states. The optional learned recovery controller is evaluated separately.

Algorithm 3 Executed SparkVLA runtime with explicit confirmation

Input. Instruction g, common planner and executor, selector, validity adapter, confirmation length L, episode time budget

Initialize semantic and raw-anchor histories to empty
while within time budget and no external abort **do**
  Acquire fresh observation and generate command c_k from g and history
  if $c_k = \text{GLOBALDONE}$ **then**
    Log predicted global completion and **return** trajectory and trace
  end if
  Extract raw anchor, fuse with history, cache α_k, and set $r \leftarrow 0$
  while current subtask active and within episode budget **do**
    Propose A_t, compute m_t, and obtain valid continuation set $\mathcal{V}_t$
    if $\mathcal{V}_t = \varnothing$ or external abort **then**
      Log abort and **return** trajectory and trace without predicted success
    end if
    Score valid candidates and log raw argmax $\hat{n}_t$
    if $\hat{n}_t = 0$ **then**
      $r \leftarrow r + 1$
      if $r = L$ **then**
        Commit subtask advance and append c_k and raw anchor to histories
        Exit inner loop
      else
        Hold pose and gripper for at least one control tick, then acquire a fresh frame
      end if
    else
      Set $r \leftarrow 0$, execute the first $\hat{n}_t$ actions, and acquire a fresh observation
    end if
  end while
end while
Log timeout if the time budget expires, then return the executed trajectory and trace

D Evaluation Controls

The revised comparisons use common held-out reset cases, observation streams, action interfaces, time budgets, and evaluator predicates. Controlled interfaces share planner/executor checkpoints and subtask supervision. Complete systems retain their native architectures, with data and transition-control differences disclosed in the provenance table. Checkpoint and hyperparameter selection use validation cases only. The compact protocol below distinguishes planned budgets from measured outcomes.

E Evaluation Protocol and Additional Evidence

This appendix records the measurements needed to interpret the central claims. All matched methods use the same held-out cases, evaluator commit, reset rules, action units, time budget, and selector seeds 7, 17, and 27. Published benchmark rows retain their source protocol. We use task success rate as the default metric. For the RoboCerebra diagnostic we also report transition completion because it measures annotated subtask events.

For N evaluation episodes, complete task success is

$$\mathrm{SR} = \frac{100}{N} \sum_{e=1}^{N} y_e, \tag{16}$$

where y_e is one only when all final-state predicates are satisfied. If episode e contains K_e annotated transitions, transition completion is

$$\mathrm{TS} = \frac{100}{N} \sum_{e=1}^{N} \frac{1}{K_e} \sum_{j=1}^{K_e} z_{ej}. \tag{17}$$

For the six RoboCerebra conditions, the reported TS Avg. is the task-count-weighted aggregate of the condition-level transition rates. If q_c is the number of evaluated tasks in condition c and TS_c is its transition completion, then

$$\text{TS Avg.} = \frac{\sum_{c=1}^{6} q_c \, \mathrm{TS}_c}{\sum_{c=1}^{6} q_c}. \tag{18}$$

The displayed condition columns therefore provide a diagnostic breakdown, while their unweighted arithmetic mean is not the benchmark aggregate. The final-state predicate, timeout rule, and intervention rule are fixed before test evaluation. We average the three seed estimates equally and retain raw episode counts whenever a count is available.

E.1 Executed Evaluation Controls

The executed diagnostic tables use fixed integer denominators and the same planner, executor, held-out cases, and confirmation rule as the main comparison. The outcome-label study retains 39.1/88.12, the LIBERO main protocol retains 98.50, and RoboCerebra distinguishes 47.12 TS from 36.1 complete-task SR. The denominators for early advance, timeout, proposal match, and Stop precision and recall are fixed before evaluation.

E.2 Matched Decision-Interface Study

The main paper summarizes the matched interface comparison to preserve room for the end-to-end results. Every row below uses the same planner, executor, anchor history, visual-token budget, held-out cases, and three selector seeds. The controls vary one decision interface at a time, so the table isolates candidate-level ordering from parameter sharing and from the stop-aware auxiliary objective.

E.3 Proposal Shift and Prefix Alignment

Boundary annotations are informative only when the proposal shown to the selector remains aligned with the executed trajectory. Recorded-prefix replay tests the aligned case. Same-executor regeneration changes the proposal while keeping the executor fixed. Executor-shifted proposals change

Table 7: Matched decision-interface study on RoboCerebra. SR is complete task success. Short labels denote the same context and capacity controls. Early advance counts a commit before annotated completion, and delay is the median nonnegative completion-to-commit delay.

Decision interface	SR (%)	Early advance (%)	Decision delay (ms)
Fixed + Stop	24.1	10.01	425
Separate heads	29.6	7.00	350
Factorized heads	30.1	6.00	325
Joint categorical	33.5	4.51	275
Ranking w/o Stop	34.3	3.51	260
SparkVLA	**36.1**	**1.99**	**225**
AAC + Stop	32.3	5.00	350

the low-level action decoder. Interior perturbations preserve the terminal control while changing actions that may create contact or collision. The terminal-matched action-blind row measures the information available without proposal content.

Table 8: Proposal-lineage diagnostic on retained paired proposals. Pair accuracy uses 9,000 retained pairs.

Proposal lineage	Match (%)	SR (%)	Pair acc. (%)	Regret	Decision p95 (ms)
Recorded-prefix replay	100.00	39.1	88.12	0.20	297
Same-executor regenerated	91.70	37.7	84.54	0.24	292
Executor-shifted proposal	84.00	36.1	80.14	0.30	290
Interior-action perturbation	76.40	34.3	73.21	0.39	294
Terminal-matched action-blind	n/a	31.1	55.03	0.70	274

E.4 STOP Confirmation Sensitivity

The selector has no score threshold at candidate selection. Deployment uses an explicit temporal confirmation rule in which Stop must be selected on L consecutive fresh observations. We sweep L from zero to three while keeping the selector, planner, executor, and action horizon fixed.

Table 9: STOP confirmation sensitivity on RoboCerebra. The confirmation counter is reset by any non-Stop choice.

Confirmation L	SR (%)	Early advance (%)	Timeout (%)	Stop prec. (%)	Stop rec. (%)	Decision p95 (ms)
$L = 0$	34.1	4.73	2.22	88.44	91.11	225
$L = 1$	35.6	2.88	2.39	91.67	93.78	258
$L = 2$	36.1	1.99	2.78	94.44	94.89	290
$L = 3$	35.8	1.54	3.61	95.00	92.56	342

E.5 Public LIBERO Context

Table 10 records public LIBERO context together with our matched reruns. Public rows keep their published data and checkpoint protocol. The final two rows use the same local task split and evaluation budget as the main paper.

E.6 Action Content and Prefix Representation

The selector compares action-blind, terminal-only, full-attention, and explicit prefix-accumulator encodings under the same boundary labels and parameter budget. Full attention exposes every action token in the proposed chunk. The accumulator provides a sequential control for the same prefix. Held-out branch states use matched terminal controls with different interior paths. Pair accuracy is computed only on pairs whose paired outcome interval excludes zero. Outcome labels are an additional data intervention and are not mixed with the default boundary-supervised result.

Table 10: LIBERO task success rate (%).

Method or provenance	Spatial	Object	Goal	Long	Average
OpenVLA-OFT (Kim et al., 2025a)	97.6	98.4	97.9	94.5	97.10
SmolVLA 0.45B (Shukor et al., 2025)	90.0	96.0	92.0	71.0	87.30
SmolVLA 2.25B (Shukor et al., 2025)	93.0	94.0	91.0	77.0	88.75
$\pi_{0.5}$, OpenPI (Physical Intelligence, 2025)	98.8	98.2	98.0	92.4	96.85
$\pi_{0.5}$ with AAC (Liang et al., 2026)	99.1	99.2	98.0	95.2	97.90
MemoryVLA (Shi et al., 2026)	98.4	98.4	96.4	93.4	96.70
X-VLA (Zheng et al., 2026)	98.2	98.6	97.8	97.6	98.10
villa-X (Chen et al., 2026b)	97.5	97.0	91.5	74.5	90.10
SparkVLA	99.2	97.6	99.2	98.0	98.50

Table 11: Prefix-content diagnostics on held-out branch states. SR denotes complete task success.

Encoding	Pair accuracy	Regret	Unsafe choice	SR	Decision p95
Action-blind	55.03	0.70	12.11	31.1	274
Terminal-only	65.12	0.55	9.11	33.3	278
Full attention, SparkVLA	80.14	0.30	4.11	36.1	290
Prefix accumulator	82.09	0.27	3.56	36.4	318
Outcome labels, separate study	88.12	0.20	3.56	39.1	297

E.7 Supervision Alignment and Seed Stability

Boundary labels are attached only to recorded completion events. Executed prefixes are distinguished from unexecuted proposals. Unknown or irrecoverable outcomes are masked rather than labeled as continuation. Failed trajectories therefore cannot force persistence in unrecoverable states. The final report includes retained-pair fractions and paired bootstrap intervals over cases with identical resampling indices across methods.

Table 12: Seed stability over three selector-training seeds. The entries show complete task success rate and the corresponding absolute gain.

Training seed	Factorized SR	No-auxiliary SR	SparkVLA SR	Joint minus factorized	Joint minus no-auxiliary
7	29.2	34.0	36.2	7.0 pp	2.2 pp
17	30.0	34.2	36.0	6.0 pp	1.8 pp
27	31.0	34.7	36.0	5.0 pp	1.3 pp
Mean $\pm$ std	$30.1 \pm 0.9\%$	$34.3 \pm 0.3\%$	$36.1 \pm 0.1\%$	6.0 ± 1.0 pp	1.8 ± 0.4 pp

Planner history and executor transfer. The semantic and latent anchor histories are updated only after a committed transition. The failure audit separates planner history, perception, premature advance, missed advance, executor contact, intervention, and timeout causes. Executor transfer first uses a frozen interface conversion, then validation-only Stop calibration, and finally selector retraining with disjoint target data. No target-specific result is described as zero-shot.

E.8 Physical Evaluation Protocol

The physical study uses the same three ordered tasks and 50 interleaved trials per method and task. Intervention, timeout, object loss, and a violation of a safety bound count as failures. We report task-level success counts and Wilson intervals. Decision latency is measured end to end on the same hardware and with the same numerical precision for every method.

Efficiency and reproducibility. The selector runs only at chunk boundaries and reuses the cached subtask anchor between decisions. The exact training seeds, episode denominators, branch masks, executor conversion, and latency measurement procedure are fixed in the executed protocol. Public

benchmark entries remain separate from matched causal comparisons, and every reported count is recoverable from its numerator and denominator.

E.9 Implementation Details

The selector uses horizon $H = 50$, 32 guided visual tokens per camera, 224×224 input images, a 2,048-dimensional anchor, a two-layer scoring transformer, and LoRA (Hu et al., 2022) with rank 16. The planner and executor remain fixed when selector interfaces are compared. Exact package versions, checkpoint hashes, and measured memory and latency are exported with the release artifact.

F Real-World Execution Visualization

The figure below records the ordered physical subtasks used in the three real-world evaluations. It complements the task definitions and final-state criteria in Appendix G.

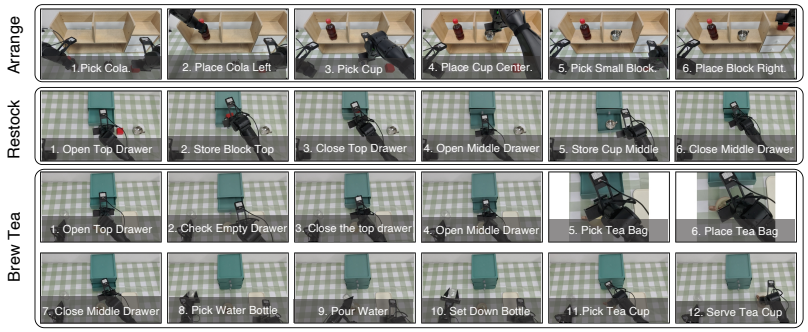


Figure 4: Representative real-world execution sequences for Shelf Arrangement, Drawer Restocking, and the twelve-step Tea Preparation task. The tea task uses white plastic beads as a dry surrogate for water.

G Real-World Experimental Details

G.1 Inference Platform and Control Stack

The real-world evaluation uses a fixed tabletop manipulation platform with RGB observation streams and proprioceptive feedback. A common local deployment stack provides perception, model inference, and robot control for every compared method. Thus, sensing and control conditions remain fixed while only the evaluated policy changes. Action commands are streamed at approximately 20 Hz, corresponding to 50 ms per action step. All compared methods use the same action-command rate.

G.2 Physical Task Definitions

The physical evaluation contains three ordered long-horizon tasks. *Shelf Arrangement* contains six subtasks. The robot picks up the cola and places it in the left shelf compartment, picks up the cup and places it in the center compartment, and then picks up the small block and places it in the right compartment. *Drawer Restocking* contains six subtasks. The robot opens the top drawer, stores the block, closes the top drawer, opens the middle drawer, stores the cup, and closes the middle drawer. *Tea Preparation* contains twelve subtasks. The robot opens the top drawer, verifies that it is empty, closes it, opens the middle drawer, retrieves the tea bag, places it in the cup, closes the middle drawer, picks up the water bottle, pours the bottle contents into the cup, sets down the bottle, picks up the prepared cup, and places it on the serving tray. These ordered sequences define the subtask histories and completion events used in data segmentation and evaluation. The task-level language retains the semantic terms *water bottle* and *pour water*. In the physical setup, however,

the bottle contains white plastic beads, which serve as a safe and repeatable dry surrogate for water. The substitution preserves the manipulation sequence relevant to this task. It includes grasping and transporting the bottle, aligning it above the cup, controlling its orientation during transfer, returning it upright, and placing it back on the table. The same bottle contents and task setup are used for all compared methods.

G.3 Data Collection and Reset Protocol

Demonstrations are collected with the global instructions and predefined subtask sequences used by the corresponding tasks. Each trajectory is segmented at subtask-completion events, and the resulting boundary labels are used only for offline supervision. Each rollout begins from a matched task-specific initial state. Object poses are randomized only within the corresponding evaluation range. A rollout is reset after success, violation of a safety bound, loss of an object from the workspace, or expiration of the task horizon. Operator intervention terminates the trial and is counted as a failure.

G.4 Final-State Criteria and Safety

For Shelf Arrangement, success requires the cola, cup, and block to be stable in the left, center, and right shelf compartments, respectively, without a drop or placement beyond a compartment edge. For Drawer Restocking, the block and cup must be fully contained in the top and middle drawers, respectively, and both drawers must finish with less than 2 cm residual opening. For Tea Preparation, the system must verify the empty top drawer, retrieve the tea bag from the middle drawer, place it in the cup, transfer the prescribed amount of white plastic beads from the bottle into the cup without substantial scattering outside the cup, return the bottle to the table, and leave the prepared cup stably on the serving tray. The transfer outcome is checked against the same predefined target level in every trial.

No liquid is used in the physical evaluation. The pouring task uses a small quantity of white plastic beads. This dry surrogate avoids liquid contact with the equipment and permits consistent resets while retaining the target-directed transfer operation. A shallow containment tray collects any scattered beads. Robot speed and gripper force are capped, and drawers have mechanical travel limits. A physical emergency stop remains accessible. Any intervention terminates and fails the rollout. The safety mechanisms operate independently of SparkVLA's task-state estimation.

H Unified Candidate Scoring Details

The anchor and current high-level hidden feature are projected into context tokens as follows.

$$e_\alpha = W_\alpha \,\mathrm{LayerNorm}(\alpha_k), \tag{19}$$

$$e_m = W_m \,\mathrm{LayerNorm}(m_t), \tag{20}$$

where $W_\alpha, W_m \in \mathbb{R}^{d_r \times d}$ project into the scoring space of dimension d_r. The Stop candidate uses a learned embedding $e_0 \in \mathbb{R}^{d_r}$, and subtask and multimodal context are supplied by e_α and e_m through self-attention. Each length-n candidate token is initialized from the terminal action of its prefix plus a learned length embedding.

$$e_n = \mathrm{MLP}_a(a_{t+n-1}) + p_n, \quad n = 1, \ldots, H. \tag{21}$$

Full self-attention over the complete token set contextualizes each candidate.

$$\mathbf{h} = \mathrm{Transformer}(\mathbf{e}), \quad \mathbf{e} = [e_\alpha, e_m, e_0, \ldots, e_H]. \tag{22}$$

H.1 Prefix-Causal Mask and Future-Action Visibility

The deployed selector uses the complete proposed chunk as a shared scoring interface. This permits the candidate tokens to compare possible execution horizons under the same anchor and visual context. A proposed future action is therefore visible while the selector ranks alternatives, but it is not executed until its candidate is selected. To make this interface explicit, we also define a strict prefix-causal mask for a controlled diagnostic.

Let candidate n denote the prefix containing actions $a_t, \ldots, a_{t+n-1}$. Under the prefix-causal diagnostic, candidate n can attend to the anchor token, the current multimodal token, the Stop token, and action tokens up to position n. It cannot attend to action tokens after position n:

$$M^{\mathrm{pc}}_{n,j} = \begin{cases} 0, & j \in \{\alpha, m, 0, 1, \ldots, n\}, \\ -\infty, & j \in \{n+1, \ldots, H\}. \end{cases} \tag{23}$$

The corresponding representations are

$$\mathbf{h}^{\mathrm{pc}} = \mathrm{Transformer}(\mathbf{e}; M^{\mathrm{pc}}). \tag{24}$$

The full-attention interface remains the main model because it directly compares candidate horizons proposed for the same re-observation point. The prefix-causal form is a causal robustness control that separates the effect of cross-candidate proposal visibility from the effect of the shared anchor and visual context. In both interfaces, only the selected prefix is sent to the executor, so future-action visibility affects ranking context rather than the executed trajectory.

A scoring layer maps each candidate's output h_i to a scalar $s_{t,i} = \mathrm{MLP}_{\mathrm{score}}(h_i) \in \mathbb{R}$, $i = 0, \ldots, H$. The context outputs h_α and h_m provide conditioning only, and scores are computed from the candidate representations $h_0, \ldots, h_H$. Although e_0 is fixed, self-attention makes the resulting Stop representation context-dependent.

Each invocation requires one anchor-guided pruning step, one shallow backbone pass, and one compact scoring-head pass. Anchor extraction is amortized over the entire subtask.