Coarse-to-Control: Action-Token Planning for Vision-Language-Action Models

Anonymous Author(s)

Abstract: Most vision-language-action (VLA) models map observations directly to actions without explicit intermediate planning, which limits performance on long-horizon tasks where early mistakes compound. We propose *Coarse-to-Control*, a plan-execute VLA that introduces planning natively in the action-token space. The key idea is to let the policy first predict a compact sequence of coarse action tokens that summarize the intended future trajectory, and then generate executable action tokens conditioned on this plan. Because both planning and execution share a unified discrete action vocabulary, the plan stays close to the control manifold and provides directly actionable guidance rather than an abstract hint that must be translated back to motor commands. Experiments on LIBERO, SimplerEnv-WidowX, and real-world manipulation tasks show that action-token planning consistently improves over direct action generation, with the largest gains on long-horizon multi-stage tasks.

Keywords: Vision-Language-Action Models, Robot Learning

1 Introduction

Vision-language-action (VLA) models have made rapid progress by learning to map visual observations and language instructions directly to robot actions [1, 2, 3, 4, 5]. Yet this direct-generation paradigm faces a fundamental representational tension: language instructions specify *what* to accomplish—"pick up the cup," "place the carrot on the plate"—but say nothing about *how* to move. To generate precise motor commands, the policy must silently resolve approach direction, wrist orientation, grasp pose, and waypoint sequence from a high-level goal description. Without an explicit intermediate layer that bridges semantic intent and motor detail, all of this resolution is compressed into a single forward pass, asking the policy to operate at two disparate levels of abstraction simultaneously.

Humans avoid this problem because the brain maintains distinct levels of motor representation. Research on skilled movement suggests that action is organized hierarchically: the nervous system first specifies a high-level plan (goal, movement direction, and grasp configuration), then refines execution through lower-level motor commands under continuous sensory feedback [6, 7, 8]. Crucially, this intermediate representation is not a language description. The instruction *place the carrot on the plate* conveys the semantic goal but encodes nothing about arm trajectory, wrist orientation, or gripper timing. A coarse motor plan carries spatial and temporal structure that language abstracts away; providing this structure explicitly, rather than leaving it implicit in the goal description, is what enables more consistent and precise execution.

Recent reasoning-augmented VLAs address this gap by inserting intermediate representations before action generation: textual rationales [9, 10], predicted visual subgoals [11, 12], or spatial and structured reasoning representations [13, 14, 15]. Each improves task understanding, but none fully bridges the underlying mismatch: text and images operate at a semantic or perceptual level rather than the level of motor intent, leaving the policy to still infer the spatial and temporal structure that precise execution requires.

This motivates a planning medium that is more control-aligned than text or images yet more structured than direct action prediction: one that lives in the action space itself. We propose *Coarse-to-Control*, a plan-execute VLA that first predicts coarse planning tokens summarizing the intended future trajectory and then generates executable action tokens conditioned on this plan. The key enabler is a joint plan-execute tokenizer that maps both planning and execution into a shared residual-VQ action vocabulary, so the plan remains close to the control manifold and provides more actionable guidance than an abstract hint that must be translated back to motor commands.

The policy is trained end-to-end from demonstrations with the same autoregressive objective used for action prediction: planning tokens are generated as an internal prefix, and executable tokens are generated conditioned on that prefix. At inference, only executable tokens are decoded into robot actions, keeping planning as lightweight latent guidance without a separate planner-controller interface. Coarse-to-Control reaches 97.90% average success on LIBERO and 83.3% on SimplerEnv-WidowX, and improves robustness across four real-world manipulation tasks.

Our contributions are:

- We identify that generating precise motor commands benefits from an intermediate motor-level intent, and argue that the action space is a more natural medium for this representation than text or images.
- We propose *Coarse-to-Control*, a plan-execute VLA with a joint tokenizer that places coarse planning tokens and executable action tokens in a shared discrete vocabulary, so the plan stays close to the control manifold and provides more actionable guidance than text or image intermediates.
- We demonstrate on LIBERO, SimplerEnv-WidowX, and real-world manipulation tasks that action-token planning consistently improves over direct action generation and outperforms textual, visual, and spatial CoT baselines.

2 Related Work

Vision-Language-Action Models. VLA models adapt pretrained vision-language backbones to robot control by learning to output actions from multimodal observations and language instructions [1, 2, 3, 4]. Recent systems such as π_0, π_0-FAST, and OpenVLA-OFT improve generality, action-token efficiency, and speed-success trade-offs [5, 16, 17]. We build on these action-generation backbones and add explicit planning before execution.

Intermediate Reasoning for VLAs. Recent VLAs introduce intermediate reasoning before control, including textual [9, 18, 10], visual-linguistic [11, 19], 3D-aware spatial [13, 15], and trajectory-based reasoning [20, 21], as summarized in Figure 1. Action CoT is closest to ours in moving reasoning toward motor outputs [22]. We instead represent reasoning as a coarse future trajectory in the same executable action-token space used for control, directly unifying planning and execution.

Action Tokenization. Discrete action representations make continuous robot control compatible with sequence models. Existing methods primarily treat action tokens as compact representations for policy execution, using per-dimension binning [2, 4], DCT-based compression [16], or neural tokenization [23, 24, 25] to balance reconstruction fidelity, compression efficiency, and structural expressiveness. Our contribution is complementary: through a joint tokenizer shared across both stages, we use action tokens not only for low-level execution, but also as an interface for high-level planning.

3 Coarse-Action Planning as Chain-of-Thought

Given visual observations o_t, a language instruction l, and robot state s_t, our goal is to generate a short sequence of executable robot actions. Standard VLAs model this as direct action genera-

tion. Figure 1 summarizes the main reasoning paradigms for VLA control, contrasting direct action generation with textual CoT, visual CoT, and our action-token CoT. In Coarse-to-Control, this action-token CoT is implemented as a planning stage followed by executable action generation:

$$z_t^{\text{plan}} \sim p_\theta(z^{\text{plan}} \mid o_t, l, s_t), \tag{1}$$

$$z_t^{\text{exec}} \sim p_\theta(z^{\text{exec}} \mid o_t, l, s_t, z_t^{\text{plan}}), \tag{2}$$

where z_t^{plan} denotes coarse action planning tokens and z_t^{exec} denotes executable action tokens. During inference only z_t^{exec} is decoded and executed by the robot. In our formulation, chain-of-thought takes the form of a coarse action trajectory that guides subsequent execution.

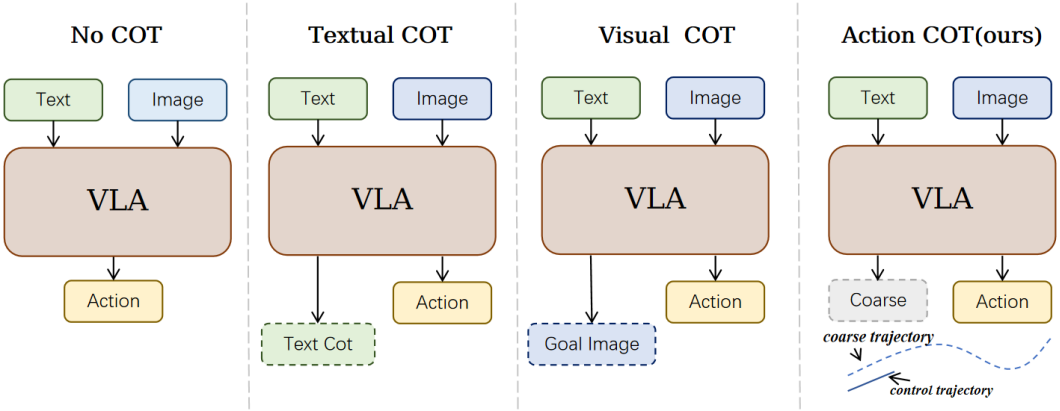

Figure 1: Comparison of reasoning paradigms for VLA control: No CoT, textual CoT, visual CoT, and our action-token CoT. Our method uses a compact action-level planning prefix before executable action generation.

3.1 Coarse-Fine Action Joint Tokenization

Action Sub-resolution to Obtain Coarse Action To represent a coarse action plan, we summarize long-horizon future behavior into a shorter trajectory that captures stage-level motion intent rather than high-frequency executable control. Concretely, for each time step t we take a long-horizon future action sequence $A_{t:t+H_p-1}$ and compress it into a coarse planning trajectory $\bar{A}_t$ with K plan steps. Here H_p is the number of source action timesteps, K is the number of coarse plan chunks, and $c = H_p/K$ is the chunk size. For the relative action representation used by our joint-mode tokenizer, this compression is implemented as

$$\bar{a}_{t+i}^{\text{motion}} = \sum_{j=0}^{c-1} a_{t+ic+j}^{\text{motion}}, \forall i \in \{i = 0, \ldots, K-1\}, \tag{3}$$

$$\bar{a}_{t+i}^{\text{gripper}} = a_{t+(i+1)c-1}^{\text{gripper}}, \forall i \in \{i = 0, \ldots, K-1\}. \tag{4}$$

Each coarse plan step stores the net relative motion over one chunk and the final gripper state of that chunk. The resulting $\bar{A}_t$ preserves the direction and stage-level intent of the future trajectory while discarding high-frequency details; concrete settings are given in the implementation details.

Dual-granularity Action Tokenizer We train a joint-mode residual-VQ action tokenizer with two modes: an execution mode $m = 0$ and a planning mode $m = 1$. The execution mode tokenizes short-horizon action chunks $A_{t:t+H_e-1}$, while the planning mode tokenizes the coarse long-horizon trajectory $\bar{A}_t$. Both modes share the same discrete action vocabulary, which encourages planning and execution tokens to remain in the same space.

Let $Q(\cdot, m)$ and $D(\cdot, m)$ be the mode-conditioned tokenizer and decoder. The target tokens are

$$z_t^{\text{plan}} = Q(\bar{A}_t, m = 1), \tag{5}$$

$$z_t^{\text{exec}} = Q(A_{t:t+H_e-1}, m = 0). \tag{6}$$

The tokenizer is trained jointly across the two modes with an action reconstruction objective and a residual-VQ regularization term that includes both codebook and commitment losses:

$$\mathcal{L}_{\text{tok}} = \mathcal{L}^{m}_{\text{rec}} + \mathcal{L}^{m}_{\text{vq}}, \tag{7}$$

This objective encourages the planning branch to encode low-frequency long-horizon intent while keeping planning and execution in the same control-aligned vocabulary. Here H_p and H_e denote the planning and executable horizons, respectively.

3.2 Plan-Execute Autoregressive VLA

After tokenization, the VLA is trained to predict the concatenated action suffix $z_t = [z_t^{\text{plan}}, z_t^{\text{exec}}]$ conditioned on image tokens, language tokens, and proprioceptive state. The autoregressive factorization is

$$p(z_t \mid o_t, l, s_t) = \prod_i p(z_{t,i}^{\text{plan}} \mid x_t, z_{t,<i}^{\text{plan}}) \prod_j p(z_{t,j}^{\text{exec}} \mid x_t, z_t^{\text{plan}}, z_{t,<j}^{\text{exec}}), \tag{8}$$

where $x_t = (o_t, l, s_t)$ is the multimodal context. The policy is trained with teacher-forced next-token prediction:

$$\mathcal{L}_{\text{VLA}} = -\sum_k \log p_\theta(z_{t,k} \mid x_t, z_{t,<k}). \tag{9}$$

At inference time, the model follows the same ordering: it first generates planning tokens and then executable tokens conditioned on the generated plan. Only the executable tokens are decoded into robot actions, so planning remains an internal context inside the autoregressive policy rather than a separate planner-controller interface. Figure 2 shows the corresponding plan-then-execute flow.

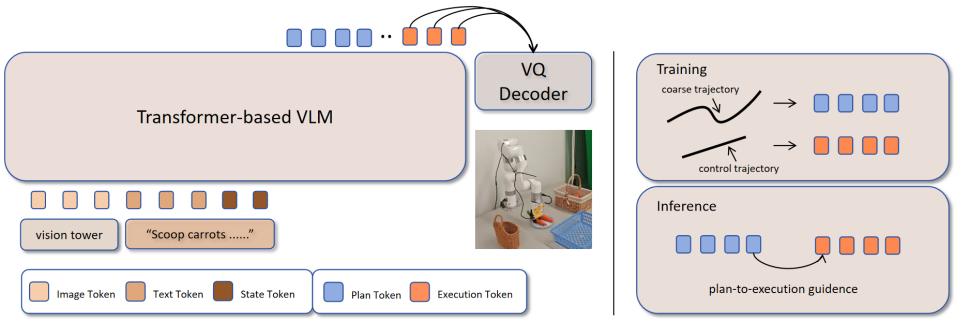

Figure 2: Plan-execute VLA framework. The policy conditions on observations, language, and proprioception, predicts coarse planning action tokens before executable action tokens, and decodes only the executable branch into continuous robot actions.

4 Experiments

We evaluate Coarse-to-Control on simulation benchmarks and real-world robot manipulation tasks. Our experiments are designed to address the following questions:

- How effective is Coarse-to-Control compared with state-of-the-art methods on simulation benchmarks? (§4.1)
- How well does Coarse-to-Control perform on long-horizon real-world manipulation tasks? (§4.2)
- What additional advantages does action-token planning offer, and which design choices matter most? (§4.3)

4.1 Simulation Experiments

Benchmarks and Datasets. We evaluate on LIBERO [26] and SimplerEnv-WidowX [27]. LIBERO contains four suites (Spatial, Object, Goal, and Long), and we report per-suite and average success over 50 rollouts per task. SimplerEnv-WidowX evaluates real-to-sim generalization on four WidowX manipulation tasks, and we report per-task and average success over 24 rollouts per task.

Baselines. We compare against recent VLA baselines spanning four reasoning styles: No-CoT, textual CoT, visual CoT, and Action CoT. The full method lists for LIBERO and SimplerEnv-WidowX are reported in Tables 1 and 2. For SimplerEnv-WidowX, we follow the standard evaluation protocol.

Results. Tables 1 and 2 show that Coarse-to-Control achieves the best overall performance on both benchmarks. These results support the central claim of this paper: reasoning directly in the action-token space provides more control-aligned guidance than explicit textual or visual CoT.

Table 1: Performance comparisons with state-of-the-art methods on LIBERO, grouped by different CoT paradigms.

CoT Type	Method	Spatial ↑	Object ↑	Goal ↑	Long ↑	Overall ↑
No CoT	π_0-FAST [16]	96.4	96.8	88.6	60.2	85.5
	SmolVLA [28]	93.0	94.0	91.0	77.0	88.8
	GR00T-N1 [29]	94.4	97.6	93.0	90.6	93.9
	π_0 [5]	96.8	98.8	95.8	85.2	94.2
	OpenVLA-OFT [17]	97.6	98.4	97.9	94.5	97.1
Textual CoT	ThinkAct [10]	88.3	91.4	87.1	70.9	84.4
	$\pi_{0.5}$ [18]	**98.8**	98.2	**98.0**	92.4	96.8
Visual CoT	CoT-VLA-7B [11]	87.5	91.6	87.6	69.0	81.1
	WorldVLA [30]	87.6	96.2	83.4	60.0	81.8
	DreamVLA [12]	97.5	94.0	89.5	89.5	92.6
	UniVLA [31]	95.4	98.8	93.6	94.0	95.5
	F1 [32]	98.2	97.8	95.4	91.3	95.7
	UD-VLA [33]	94.1	95.7	91.2	89.6	92.7
Action CoT	MolmoAct-7B-D [20]	87.0	95.4	87.6	77.2	86.6
	Coarse-to-Control (ours)	**98.8**	**100.0**	97.8	**95.0**	**97.9**

Table 2: Performance comparisons with state-of-the-art methods on SimplerEnv-WidowX, grouped by different CoT paradigms.

CoT Type	Method	Put Spoon	Put Carrot	Stack Block	Put Eggplant	Overall ↑
No CoT	OpenVLA [4]	0.0	0.0	0.0	4.1	1.0
	Octo [3]	47.2	9.7	4.2	56.9	29.5
	π_0 [5]	29.1	0.0	16.7	62.5	40.1
	CogACT [34]	71.7	50.8	15.0	67.5	51.3
Textual CoT	ThinkAct [10]	58.3	37.5	8.7	70.8	43.8
Visual CoT	F1 [32]	50.0	70.8	50.0	66.7	59.4
	UD-VLA [33]	58.3	62.5	54.1	75.0	62.5
Action CoT	Coarse-to-Control (ours)	**100.0**	**95.8**	**79.2**	58.3	**83.3**

4.2 Real-World Experiments

Real-World Setup. We evaluate Coarse-to-Control on four physical manipulation tasks: putting the carrot on the plate, putting the carrot on the plate and pressing the button, moving carrots from the plate into the basket, and clearing fruits and vegetables from the table into the plate placed in the basket. All methods are trained on the same 50-demonstration-per-task budget, evaluated over 20 rollout trials under identical scene setups and success criteria. We compare the plan-based policy with faster, π_0, and π_0-fast variants.

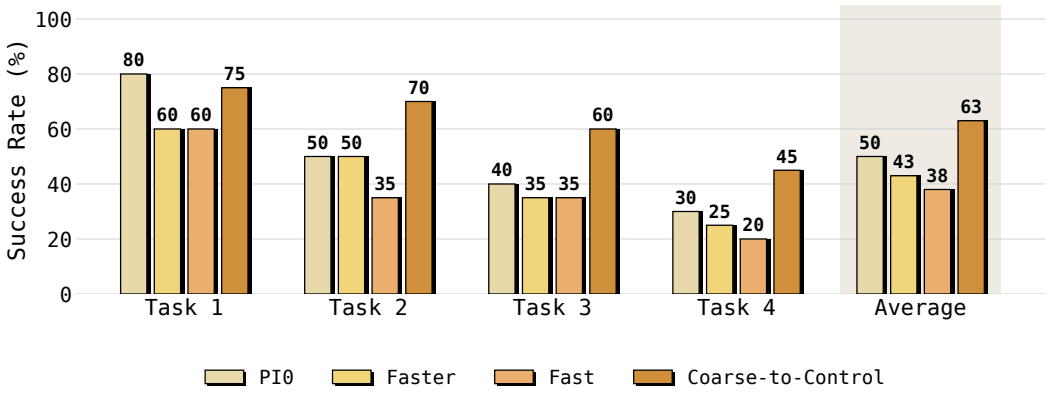

Figure 3: Real-world task success rates (%). We report final task success for each task and the average success rate across four tasks. Carrot: put the carrot on the plate; Carrot+Button: put the carrot on the plate and press the button; Plate→Basket: move carrots from the plate into the basket; Cleanup: clear fruits and vegetables from the table into the plate placed in the basket.

Results. As shown in Figure 3, the plan-based policy achieves the highest average success rate among all compared methods, reaching 62.5% over four real-world tasks. It obtains the best performance on three out of four tasks. Although π_0 performs best on the shortest single-stage carrot placement task, its advantage does not transfer to the longer multi-stage settings. These results suggest that action-token planning improves robustness to error accumulation and helps preserve temporal coordination over long-horizon physical manipulation.

4.3 Analysis

We organize the analysis around four questions that explain the main results: how planning horizon affects performance, why action-token plans are effective, whether planning and execution should share the same token space, and whether planning improves real-world long-horizon performance.

How Does Planning Horizon Matter? Table 3 uses a fixed joint action tokenizer and varies only the planning horizon. First, planning itself is beneficial: increasing the planning horizon from 0 to 160 improves the LIBERO average success rate from 96.45% to 97.90%. This interpretation is also consistent with Table 4, where the Separate variant, which uses explicit planning tokens with two independent VQ tokenizers for planning and execution rather than a shared tokenizer vocabulary, still improves overall success from 95.40% to 96.60% relative to the no-plan Faster-AR baseline, which uses a single VQ tokenizer for execution only. This suggests that the gain does not come merely from the shared discrete action vocabulary. If shared tokenization were the sole driver, the $H_p = 0$ variant would already recover most of the benefit. Instead, the gap shows that explicit plan supervision helps the policy encode future task structure before action decoding, reducing the burden on the executable branch to infer long-horizon intent from only the current observation. Second, once planning is introduced, the amount of future context also matters. Increasing the planning source horizon from 40 to 160 improves the overall LIBERO score and gives a consistent gain on the Long suite, showing that planning quality depends on access to sufficiently long future context. The 40-step variant already shows that a short plan is better than no plan, but the 160-step variant performs better still, indicating that a short plan can encode the next local maneuver whereas a longer source horizon better captures stage transitions such as approach, grasp, transport, and placement. This is especially important when the policy must choose actions that are locally valid yet only make sense in the context of a later subgoal.

Why Are Action-Token Plans Effective? Prior work suggests that chain-of-thought is effective not merely because it produces longer outputs, but because it introduces explicit intermediate states that transform difficult one-shot prediction into structured sequential computation [35, 36, 37]. In

embodied control, however, the benefit of CoT also depends on the representation medium. Textual CoT can provide high-level semantic decomposition but remains weakly constrained with respect to low-level motor behavior [9], while visual CoT offers intuitive spatial structure yet requires generating long non-executable visual prefixes before action prediction [11]. By contrast, our method uses coarse action tokens as the intermediate state, so the reasoning trace lies closer to the control manifold and can align more directly with downstream executable actions. This coarse-to-fine design is also inspired by visual autoregressive modeling such as VAR, which first predicts global structure and then progressively refines local detail [38].

Table 3. LIBERO planning-horizon H_p ablation using joint-mode action-token planning.

H_p	Spatial ↑	Object ↑	Goal ↑	Long ↑	Overall ↑
0	97.00	99.60	95.00	94.20	96.45
40	98.60	99.60	97.80	94.20	97.55
160	98.80	100.00	97.80	95.00	97.90

(a) Attention: plan vs. w/o plan. (b) Plan visualization.

Figure 5. Qualitative analysis of joint-mode action-token planning.

To qualitatively inspect this mechanism, we use two diagnostics: attention over image tokens and a decoded coarse trajectory from the predicted planning tokens. Figure 5a compares the first-frame attention map produced with planning tokens against the no-plan baseline. With planning, attention concentrates more strongly on task-relevant regions around the target object, gripper, and target area, whereas the no-plan baseline is less anchored to the target-object interaction region. Figure 5b shows that the decoded coarse plan already points toward the target before any executable actions are generated. Together, these diagnostics suggest that the planning tokens carry object-target relations and future motion direction in a form that can guide subsequent control. In other words, the plan branch appears to establish a grounded coarse intent before the execution branch predicts detailed motor commands, which may help explain why action-space CoT can provide a more direct control signal for low-level control than higher-level textual or visual intermediates.

Should Planning and Execution Share Tokens? Table 4 isolates the tokenizer design. Faster-AR provides the no-plan reference, where the policy directly predicts executable action tokens without an explicit planning prefix. Separate uses two independent VQ action tokenizers for planning and execution, so the policy still predicts plan tokens before execution tokens but the two branches no longer share a vocabulary. Joint-mode improves overall success from 96.60% to 97.90% and raises the Long-suite score from 91.60% to 95.00%, supporting the hypothesis that plan tokens should live on the same action-semantic manifold as executable tokens. When the two branches use separate tokenizers, the execution policy must implicitly translate from a planning vocabulary into an execution vocabulary, which weakens the conditioning signal. A shared token space reduces this interface mismatch and makes the predicted plan easier to reuse as actionable guidance rather than as an abstract hint.

Table 4: Tokenizer-sharing ablation. Faster-AR is the no-plan baseline. Separate uses two independent VQ tokenizers for planning and execution, while Joint-mode shares the vocabulary across the two branches.

Tokenizer	Spatial ↑	Object ↑	Goal ↑	Long ↑	Overall ↑
Faster-AR	99.40	98.80	94.80	88.60	95.40
Separate	97.40	99.60	97.80	91.60	96.60
Joint-mode (ours)	98.80	100.00	97.80	95.00	97.90

Does Planning Improve Real-World Long-Horizon Performance? Figure 6 summarizes average full-task success on the three multi-stage real-world tasks. Coarse-to-Control achieves the highest long-horizon average and outperforms all baselines. This suggests

that the benefit of planning is especially pronounced when the robot must preserve task progress across multiple stages instead of solving only a short single-step placement. The plan-based policy preserves more intermediate progress before final completion, suggesting that action-token plans help maintain the full task trajectory under real-world perturbations, where a short-term action policy can complete early subtasks but fail to finish the entire sequence.

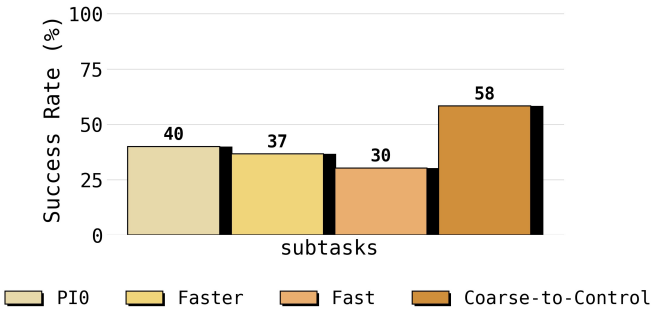

Figure 6: Average success on the multi-stage real-world tasks.

5 Limitations

Although Coarse-to-Control shows strong performance in both simulation and real-world robot manipulation, it represents only one point in the broader design space of action-space reasoning. In particular, our current formulation realizes action-token chain-of-thought by predicting coarse future actions before execution, but how to construct more expressive and adaptive action-space reasoning schemes remains an open direction for future work. Moreover, while our results show that a joint action tokenizer can effectively align coarse planning actions and executable actions within a shared token space, how to more organically unify these two granularities and better capture shared structure in the action space remains to be explored.

6 Conclusion

We presented Coarse-to-Control, a plan-execute framework for Vision-Language- Action models that internalizes chain-of-thought reasoning into the action-token space. Rather than generating explicit textual rationales or visual subgoals, the policy first predicts a compact coarse action plan and then conditions executable action generation on this plan. A joint plan-execute tokenizer aligns long-horizon planning tokens and short-horizon executable tokens within a shared discrete action vocabulary, keeping planning close to the control manifold. Experiments on LIBERO, SimplerEnv-WidowX, and real-world robot manipulation tasks show that action-token reasoning improves benchmark performance, strengthens long-horizon robustness, and is especially effective on multi-stage tasks where intermediate progress must be preserved. More broadly, these results suggest that action-token planning provides a compact and promising alternative to explicit CoT generation for embodied policy learning without introducing a separate planner-controller interface.

References

[1] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. doi:10.48550/arXiv.2212.06817. URL https://arxiv.org/abs/2212.06817.

[2] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023. doi:10.48550/arXiv.2307.15818. URL https://arxiv.org/abs/2307.15818.

[3] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024. doi:10.48550/arXiv.2405.12213. URL https://arxiv.org/abs/2405.12213.

[4] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. doi:10.48550/arXiv.2406.09246. URL https://arxiv.org/abs/2406.09246.

[5] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. doi:10.48550/arXiv.2410.24164. URL https://arxiv.org/abs/2410.24164.

[6] D. A. Rosenbaum. *Human motor control*. Academic press, 2009.

[7] N. A. Bernstein. *The Co-ordination and Regulation of Movements*. Pergamon Press, Oxford, 1967.

[8] K. S. Lashley. The problem of serial order in behavior. In L. A. Jeffress, editor, *Cerebral Mechanisms in Behavior*, pages 112–136. Wiley, New York, 1951.

[9] M. Zawalski, W. Chen, K. Pertsch, O. Mees, C. Finn, and S. Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024. doi:10.48550/arXiv.2407.08693. URL https://arxiv.org/abs/2407.08693.

[10] C.-P. Huang, Y.-H. Wu, M.-H. Chen, Y.-C. F. Wang, and F.-E. Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. *arXiv preprint arXiv:2507.16815*, 2025. doi:10.48550/arXiv.2507.16815. URL https://arxiv.org/abs/2507.16815.

[11] Q. Zhao, Y. Lu, M. J. Kim, Z. Fu, Z. Zhang, Y. Wu, Z. Li, Q. Ma, S. Han, C. Finn, A. Handa, M.-Y. Liu, D. Xiang, G. Wetzstein, and T.-Y. Lin. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. *arXiv preprint arXiv:2503.22020*, 2025. doi:10.48550/arXiv.2503.22020. URL https://arxiv.org/abs/2503.22020.

[12] W. Zhang, H. Liu, Z. Qi, Y. Wang, X. Yu, J. Zhang, R. Dong, J. He, F. Lu, H. Wang, Z. Zhang, L. Yi, W. Zeng, and X. Jin. Dreamvla: A vision-language-action model dreamed with comprehensive world knowledge. *arXiv preprint arXiv:2507.04447*, 2025. doi:10.48550/arXiv.2507.04447. URL https://arxiv.org/abs/2507.04447.

[13] D. Qu, H. Song, Q. Chen, Y. Yao, X. Ye, Y. Ding, Z. Wang, J. Gu, B. Zhao, D. Wang, and X. Li. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025. doi:10.48550/arXiv.2501.15830. URL https://arxiv.org/abs/2501.15830.

[14] C. Gao, Z. Liu, Z. Chi, J. Huang, X. Fei, Y. Hou, Y. Zhang, Y. Lin, Z. Fang, Z. Jiang, and L. Shao. Vla-os: Structuring and dissecting planning representations and paradigms in vision-language-action models. *arXiv preprint arXiv:2506.17561*, 2025. doi:10.48550/arXiv.2506.17561. URL https://arxiv.org/abs/2506.17561.

[15] H. Huang, M. Cen, K. Tan, X. Quan, G. Huang, and H. Zhang. Graphcot-vla: A 3d spatial-aware reasoning vision-language-action model for robotic manipulation with ambiguous instructions. *arXiv preprint arXiv:2508.07650*, 2025. doi:10.48550/arXiv.2508.07650. URL https://arxiv.org/abs/2508.07650.

[16] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025. doi:10.48550/arXiv.2501.09747. URL https://arxiv.org/abs/2501.09747.

[17] M. J. Kim, C. Finn, and P. Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025. doi:10.48550/arXiv.2502.19645. URL https://arxiv.org/abs/2502.19645.

[18] K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. doi:10.48550/arXiv.2504.16054. URL https://arxiv.org/abs/2504.16054.

[19] Z. Zhong, J. Li, J. He, H. Yan, X. Gong, G. Zhao, Y. Cai, J. Gao, X. Yan, B. Liu, Y. Chen, L. Yang, and H. Li. Dualcot-vla: Visual-linguistic chain of thought via parallel reasoning for vision-language-action models. *arXiv preprint arXiv:2603.22280*, 2026. doi:10.48550/arXiv.2603.22280. URL https://arxiv.org/abs/2603.22280.

[20] J. Lee, J. Duan, H. Fang, Y. Deng, S. Liu, B. Li, B. Fang, J. Zhang, Y. R. Wang, S. Lee, W. Han, W. Pumacay, A. Wu, R. Hendrix, K. Farley, E. VanderBilt, A. Farhadi, D. Fox, and R. Krishna. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025. doi:10.48550/arXiv.2508.07917. URL https://arxiv.org/abs/2508.07917.

[21] H. Fang, J. Duan, D. Clay, S. Wang, S. Liu, W. Huang, X. Fan, W.-C. Tsai, S. Chen, Y. R. Wang, S. Xing, J. Cho, J. S. Park, A. Eftekhar, P. Sushko, K. Farley, A. Wadhwa, C. Harrison, W. Han, Y.-C. Lee, E. VanderBilt, R. Hendrix, S. Ellawela, L. Ngoo, J. Chai, Z. Ren, A. Farhadi, D. Fox, and R. Krishna. Molmoact2: Action reasoning models for real-world deployment. *arXiv preprint arXiv:2605.02881*, 2026. doi:10.48550/arXiv.2605.02881. URL https://arxiv.org/abs/2605.02881.

[22] L. Zhong, Y. Liu, Y. Wei, Z. Xiong, M. Yao, S. Liu, and G. Ren. Acot-vla: Action chain-of-thought for vision-language-action models. *arXiv preprint arXiv:2601.11404*, 2026. doi:10.48550/arXiv.2601.11404. URL https://arxiv.org/abs/2601.11404.

[23] Y. Liu, S. Zhang, Z. Dong, B. Ye, T. Yuan, X. Yu, L. Yin, C. Lu, J. Shi, L. J.-T. Yu, L. Zheng, T. Jiang, J. Gong, X. Qiu, and H. Zhao. Faster: Toward efficient autoregressive vision language action modeling via neural action tokenization. *arXiv preprint arXiv:2512.04952*, 2025. doi:10.48550/arXiv.2512.04952. URL https://arxiv.org/abs/2512.04952.

[24] Z. Dong, Y. Liu, S. Zhang, B. Ye, Y. Yuan, F. Ni, J. Gong, X. Qiu, H. Zhao, Y. Li, et al. Actioncodec: What makes for good action tokenizers. *arXiv preprint arXiv:2602.15397*, 2026.

[25] Y. Wang, H. Zhu, M. Liu, J. Yang, H.-S. Fang, and T. He. Vq-vla: Improving vision-language-action models via scaling vector-quantized action tokenizers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11089–11099, 2025.

[26] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

[27] X. Li, K. Hsu, J. Gu, K. Pertsch, O. Mees, H. R. Walke, C. Fu, I. Lunawat, I. Sieh, S. Kirmani, S. Levine, J. Wu, C. Finn, H. Su, Q. Vuong, and T. Xiao. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024.

[28] M. Shukor, D. Aubakirova, F. Capuano, P. Kooijmans, S. Palma, A. Zouitine, M. Aractingi, C. Pascal, M. Russi, A. Marafioti, S. Alibert, M. Cord, T. Wolf, and R. Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025. doi:10.48550/arXiv.2506.01844. URL https://arxiv.org/abs/2506.01844.

[29] NVIDIA, J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. J. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, J. Jang, Z. Jiang, J. Kautz, K. Kundalia, L. Lao, Z. Li, Z. Lin, K. Lin,

G. Liu, E. Llontop, L. Magne, A. Mandlekar, A. Narayan, S. Nasiriany, S. Reed, Y. L. Tan, G. Wang, Z. Wang, J. Wang, Q. Wang, J. Xiang, Y. Xie, Y. Xu, Z. Xu, S. Ye, Z. Yu, A. Zhang, H. Zhang, Y. Zhao, R. Zheng, and Y. Zhu. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025. doi:10.48550/arXiv.2503.14734. URL https://arxiv.org/abs/2503.14734.

[30] J. Cen, C. Yu, H. Yuan, Y. Jiang, S. Huang, J. Guo, X. Li, Y. Song, H. Luo, F. Wang, D. Zhao, and H. Chen. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025. doi:10.48550/arXiv.2506.21539. URL https://arxiv.org/abs/2506.21539.

[31] Y. Wang, X. Li, W. Wang, J. Zhang, Y. Li, Y. Chen, X. Wang, and Z. Zhang. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025. doi:10.48550/arXiv.2506.19850. URL https://arxiv.org/abs/2506.19850.

[32] Q. Lv, W. Kong, H. Li, J. Zeng, Z. Qiu, D. Qu, H. Song, Q. Chen, X. Deng, and J. Pang. F1: A vision-language-action model bridging understanding and generation to actions. *arXiv preprint arXiv:2509.06951*, 2025. doi:10.48550/arXiv.2509.06951. URL https://arxiv.org/abs/2509.06951.

[33] J. Chen, W. Song, P. Ding, Z. Zhou, H. Zhao, F. Tang, D. Wang, and H. Li. Unified diffusion vla: Vision-language-action model via joint discrete denoising diffusion process. *arXiv preprint arXiv:2511.01718*, 2025. doi:10.48550/arXiv.2511.01718. URL https://arxiv.org/abs/2511.01718.

[34] Q. Li, Y. Liang, Z. Wang, L. Luo, X. Chen, M. Liao, F. Wei, Y. Deng, S. Xu, Y. Zhang, X. Wang, B. Liu, J. Fu, J. Bao, D. Chen, Y. Shi, J. Yang, and B. Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024. doi:10.48550/arXiv.2411.19650. URL https://arxiv.org/abs/2411.19650.

[35] K. Li, L. Clouatre, R. Pathak, and S. Ermon. Chain of thought empowers transformers to solve inherently serial problems. *arXiv preprint arXiv:2402.12875*, 2024. doi:10.48550/arXiv.2402.12875. URL https://arxiv.org/abs/2402.12875.

[36] M. Nye, A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan, C. Sutton, L. Soares, Y. Hu, D. Chen, and O. Habryka. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021. doi:10.48550/arXiv.2112.00114. URL https://arxiv.org/abs/2112.00114.

[37] X. Wang, J. Wei, D. Schuurmans, Q. V. Le, E. H. Chi, and S. Narang. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2023. doi:10.48550/arXiv.2203.11171. URL https://arxiv.org/abs/2203.11171.

[38] K. Tian, Y. Jiang, Z. Yuan, B. Peng, and L. Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *arXiv preprint arXiv:2404.02905*, 2024. doi:10.48550/arXiv.2404.02905. URL https://arxiv.org/abs/2404.02905.