Fast ECoT: Efficient Embodied Chain-of-Thought via Thoughts Reuse

Zhekai Duan[1], **Yuan Zhang**[2], **Shikai Geng**[1], **Gaowen Liu**[3], **Joschka Boedecker**[2], **Chris Xiaoxuan Lu**[1]
[1]University College London, UK
[2]University of Freiburg, Germany
[3]Cisco Research, USA

Abstract: Embodied Chain-of-Thought (ECoT) reasoning enhances vision-language-action (VLA) models by improving performance and interpretability through intermediate reasoning steps. However, its sequential autoregressive token generation introduces significant inference latency, limiting real-time deployment. We propose Fast ECoT, an inference-time acceleration method that exploits the structured and repetitive nature of ECoT to (1) cache and reuse high-level reasoning across timesteps and (2) parallelise the generation of modular reasoning steps. Additionally, we introduce an asynchronous scheduler that decouples reasoning from action decoding, further boosting responsiveness. Fast ECoT requires no model changes or additional training and integrates easily into existing VLA pipelines. Experiments in both simulation (LIBERO) and real-world robot tasks show up to a 7.5× reduction in latency with comparable or improved task success rate and reasoning faithfulness, bringing ECoT policies closer to practical real-time deployment.

Keywords: Embodied Chain-of-Thought Reasoning, Inference Acceleration, Vision-Language-Action Models

1 Introduction

Large-scale vision-language-action (VLA) models have recently advanced robotic control by leveraging internet-scale visual and textual knowledge [1, 2, 3]. By combining pre-trained vision-language backbones with policy learning, these models exhibit impressive generalisation across open-world tasks. Among their most powerful capabilities is *chain-of-thought* (CoT) reasoning—the ability to iteratively generate intermediate reasoning steps before taking an action. In robotics, *Embodied Chain-of-Thought* (ECoT) [4] extends this concept by enabling robots to "think out loud"—generating step-by-step textual reasoning traces (e.g., plans, subgoals, grounded visual inferences) at each time step, explicitly encoding the robot's thought process before emitting an action. This augmentation improves model interpretability and boosts success rates.

While ECoT offers these benefits, they come at a steep computational cost. Generating reasoning traces involves producing dozens of tokens *autoregressively* at each time step, resulting in sequential generation delays. As tasks grow more complex, the length of these reasoning chains increases, compounding the latency. In real-time robotic control, policies must react quickly to new observations; the inference overhead introduced by ECoT can slow the control loop to impractical speeds. In other words, the robot idles much of its time "thinking" rather than acting. Reducing this latency without sacrificing reasoning quality or task performance is essential for making ECoT viable in real-world deployments.

In this work, we address this inference bottleneck by proposing `Fast ECoT`, a novel method for accelerating embodied chain-of-thought reasoning through thought reuse and parallelised reasoning. Our key insight is that ECoT outputs structurally and exhibits a high degree of temporal locality: many reasoning steps—such as recalling the task goal or rechecking the state of a target object—are

repeated across time steps. Rather than regenerating the full reasoning trace at every step, we identify and cache recurring reasoning segments, reusing them in subsequent time steps. This reduces redundant computation while preserving the structure and interpretability of the thought process.

Building on this reuse, we introduce a partial parallelisation strategy that transforms the traditionally sequential reasoning process into a batched one. By branching only when necessary to handle novel information, Fast ECoT enables multiple reasoning steps to be generated in parallel—substantially reducing inference latency. We further introduce an *asynchronous scheduling* mechanism that decouples action and reasoning generation. Recognising that robot actions evolve more slowly than sensor observations, we prioritise fast action decoding while allowing reasoning traces to update asynchronously in the background. This design reduces latency without compromising decision quality, as the cached reasoning remains stable over short time horizons.

In summary, our contributions are as follows:

- We propose `Fast ECoT`, a novel inference-time acceleration method that caches and reuses intermediate reasoning steps, enabling partially parallel chain-of-thought generation for the first time in embodied policies.
- We introduce an asynchronous scheduling strategy that decouples reasoning and action generation, allowing high-frequency control signals to be emitted while reasoning is updated in the background.
- Our method requires no model changes or additional training and can be seamlessly integrated into existing VLA models that perform embodied reasoning, making it broadly usable in practical robotic systems.

2 Related Work

Foundation Models for Robotic Manipulation. Recent advances in robot learning have produced large-scale generalist policies to excel at robotic manipulation tasks, by first pre-training on diverse multimodal datasets [1, 3, 2], and further fine-tuning on extensive robot-specific data collections [5, 6]. Vision-language-action (VLA) architectures [1, 3, 2]—which integrate vision-language models pretrained on internet-scale corpora [7, 8]—further unify perception, language, and control into a single transformer-based policy, demonstrating state-of-the-art performance on benchmarks. Nonetheless, current VLA approaches purely generate robot actions and largely underutilise the inherent multi-step reasoning capabilities of their language backbones.

Reasoning for Robotic Control. Chain-of-thought (CoT) prompting [9, 10] has proven effective in enhancing large language models by encouraging step-by-step reasoning. Prior methods employ pre-trained language models for high-level planning [11, 12, 13], often requiring separate low-level controllers for execution. More recent work explores end-to-end integration of reasoning into robotic policies. Notably, Embodied Chain-of-Thought (ECoT) [4] enables robots to generate grounded, interpretable reasoning traces referencing visual and environmental observations, leading to improved performance when paired with VLA models like OpenVLA [1]. EMMAX [14] similarly embeds reasoning into action pipelines. Our work builds on ECoT, targeting its major bottleneck—high inference latency from autoregressive token generation—by proposing a partially parallelised reasoning framework that preserves interpretability while substantially accelerating inference.

Inference Optimisation in Language and Multimodal Models. A wide range of techniques have been proposed to speed up inference in autoregressive models. Speculative decoding [15, 16] accelerates generation by predicting tokens with a lightweight draft model and verifying them with the full model. Non-autoregressive and parallel decoding strategies have also been adopted, particularly in machine translation and, more recently, in robotic control [17, 18]. Quantisation methods [19, 20, 21] are widely used to reduce model precision for faster computation. In contrast to these methods, which typically operate at the token or model level, our work focuses on *reasoning-level acceleration*—breaking CoT generation into reusable, semantically meaningful segments that

can be cached and executed in parallel. This approach is model-agnostic and does not rely on auxiliary models, fine-tuning, or precision reduction. By enabling efficient reuse and branching during reasoning, our method offers a lightweight and integrable solution for speeding up VLA policies with CoT reasoning, without compromising interpretability or task success.

3 Preliminaries

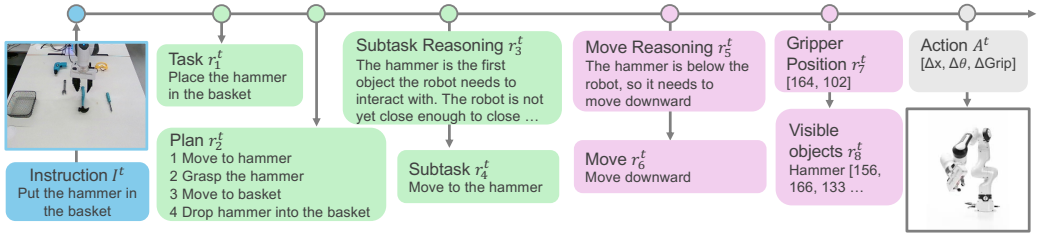


Figure 1: ECoT [4] reasoning autoregressively generates high-level **(green)** and low-level **(purple)** reasoning steps to enhance VLA performance.

3.1 Embodied Chain-of-Thought Reasoning (ECoT)

Vision-Language-Action (VLA) models [1] build on large pre-trained vision-language models, fine-tuning them to map a natural language instruction I^t and image observation O^t directly to low-level robot actions A^t via autoregressive token prediction at each control step t. Embodied Chain-of-Thought (ECoT) [4] reasoning augments this reactive paradigm by supervising the model to generate a structured sequence of N intermediate reasoning steps $R^t = \{ r_i^t \mid i = 1, 2, \ldots, N \}$, e.g. rephrased task, high-level plan, grounded sub-task, low-level move command, visual features, before emitting the final action A^t. The reasoning steps are separated into high- and low-level steps, as clarified in Fig. 1 and appendix A.1.

3.2 Continuous Batching

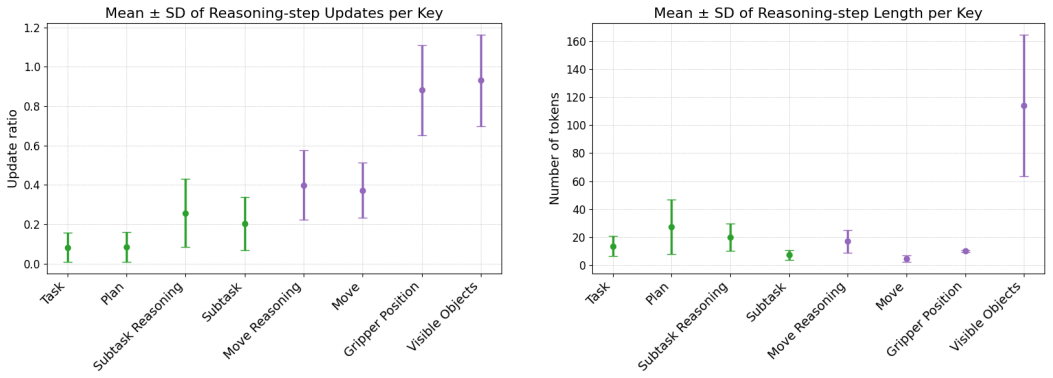
In autoregressive models, batching improves throughput, but static batching can be inefficient when sequence lengths vary—shorter sequences must wait for longer ones, wasting compute on padding. Continuous batching [22] addresses this by dynamically replacing completed sequences with new ones, maintaining high GPU utilization and minimising idle time. This strategy, used in systems like Tensorrt-LLM [23] and vLLM [24], has shown 2–4× throughput gains in LLM serving.

4 Method

4.1 Inference Characteristics of ECoT Reasoning

Unlike traditional chain-of-thought approaches in language modelling [25, 26] and vision-language modelling [27, 28], which yield diverse, dynamic reasoning patterns that rarely *recur*, ECoT consistently follows a structured, periodically recurring workflow: planning, sub-task identification, motion reasoning, and visual-feature processing, before predicting subsequent robot actions. To analyse this behaviour quantitatively, we sample all 1,110 episodes containing reasoning from the Bridge V2 dataset [5] and compute the average and the standard deviation of the update ratio (percentage of reasoning content updated at the next time step) and token length for each reasoning step. As depicted in Fig. 2a, higher-level reasoning components in ECoT (e.g., planning and subtask reasoning) remain relatively similar across multiple time steps within an episode[1]. For example,

[1]While profiling the update ratios, we observed substantial variability in the visible detection module. This counterintuitive result stems from the instability inherent in the visual reasoning module of ECoT and the open-vocabulary nature of object labels, whereby identical objects may be assigned varying labels over time.



(a) Average update ratio of reasoning steps, highlighting that high-level reasoning components remain relatively stable, whereas lower-level steps require more frequent updates.

(b) Average length and standard deviation of different reasoning steps, showing significant disparities in token counts.

Figure 2: Statistics illustrating the pattern of ECoT reasoning steps under Bridge V2 [5].

the planning module exhibits an average update ratio of only 8.4%, meaning 91.6% of its reasoning content is unchanged and can be reused in subsequent inference steps without autoregressive regeneration. Leveraging this temporal locality, we propose to cache ECoT reasoning for reuse in successive time steps, thus potentially enabling the parallel generation of each reasoning step.

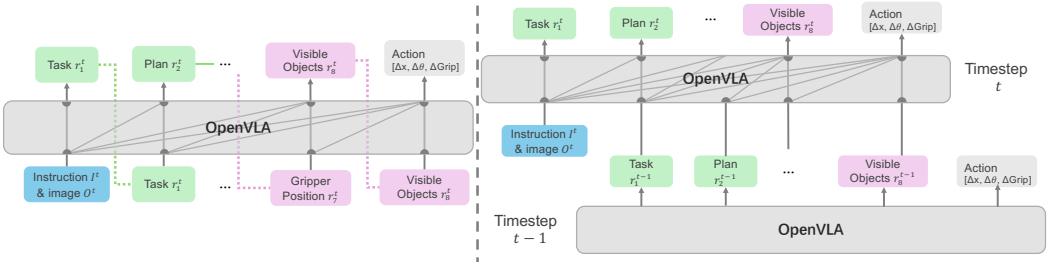


Figure 3: Comparison between ECoT **(left)** and our proposed `Fast ECoT` **(right)**. Both decompose reasoning into fixed stages (e.g., task, plan, object grounding), but ECoT generates these sequentially at every step, while Fast ECoT enables parallel generation and reuses cached high-level reasoning from previous timesteps. The dotted lines coloured in green/magenta represent token copying.

4.2 Efficient Parallelisation of ECoT Reasoning and Action Generation

Compared to the original ECoT—which generates the full reasoning sequence sequentially and autoregressively at every timestep (see Fig. 3 left)—our Fast ECoT reformulates each reasoning step r_n^t as a standalone generation task. For each step, we construct the input by combining the current observation O^t, instruction I^t, and the previously generated reasoning steps $R^t = \{r_i^t \mid i = 1, 2, \ldots, n-1\}$ from the last timestep as prefix context. This allows all reasoning steps and the action at timestep t to be generated independently and in parallel, rather than waiting for preceding components to finish (see Fig. 3 right).

While conceptually straightforward, this strategy introduces performance overhead. Since each reasoning module prepends a growing context of prior thoughts, the resulting input lengths vary significantly—early steps have short prompts, while later ones accumulate more history. Additionally, output lengths also differ across reasoning steps: low-level steps like gripper commands may require fewer than 20 tokens, while object-grounding reasoning can exceed 120 (see Fig. 2b). Traditional

static batching [22] (see Fig. 4) handles such variability by padding all sequences in a batch to match the longest, which leads to substantial inefficiency on modern accelerators. This results in wasted compute on padding tokens, poor GPU utilisation, and offsets the gains from parallel generation.

To address inefficiencies from padding, we adopt continuous batching [22], a dynamic scheduling strategy used in modern LLM serving engines[24, 23]. Instead of padding all sequences to a fixed length, continuous batching maintains a queue where completed sequences are immediately replaced by new ones, allowing variable-length inputs to be processed efficiently. This minimises wasted computation on padding tokens, significantly improves GPU utilisation, and can reduce the total number of tokens processed by up to 6× (see Fig. 4). We use vLLM [24] as our inference backend, and the full pseudocode for `Fast ECoT` is provided in appendix A.2.

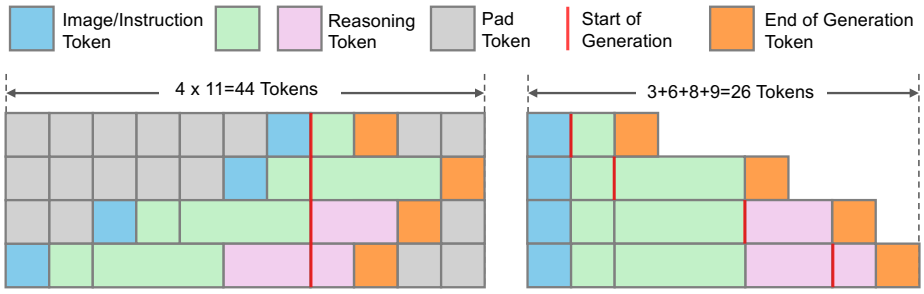


Figure 4: Illustrative example comparing static vs. continuous batching for reasoning generation. **Left:** Static batching pads to the longest sequence, processing 4×11=44 tokens. **Right:** Continuous batching processes only actual tokens (3, 6, 8, 9), adding up to 26 tokens, reducing padding and improving efficiency.

4.3 Optimising Asynchronous Scheduling of Reasoning and Action Updates

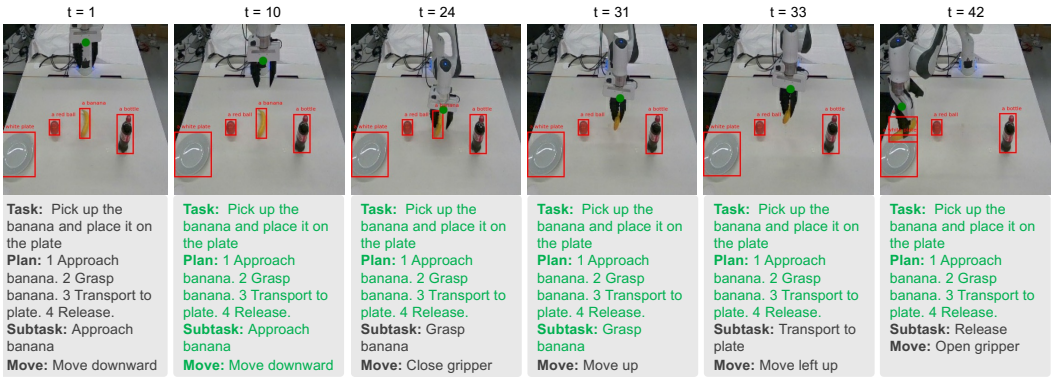


Figure 5: Generated robot rollouts at successive time steps **(top row)** alongside its reasoning **(bottom row)**. Between frames, a large part of the reasoning remains unchanged **(Green)**. At each timestep (t=1, 10, 24, 31, 33, 42), the Subtask updates intermittently, and the low-level Move command adapts continuously as it picks up the banana and places it on the plate.

As discussed in Sec. 4.2, we formulate reasoning and action generation at each time step as batched requests that can be processed in parallel. However, reasoning traces typically span hundreds of tokens, while action decoding involves only a few (around 7) tokens. In a synchronised setup, this mismatch causes unnecessary latency: the agent must wait for all reasoning steps to complete before it can act.

While the original ECoT framework couples reasoning and action tightly for interpretability and causal grounding, we argue that this coupling need not be time-synchronous. In practice, high-level reasoning elements—such as Task and Plan—remain stable across multiple steps, and their

semantic influence on action decisions persists even when updated less frequently. This is evident in the rollout shown in Fig. 5, where reasoning traces and visible objects stay largely consistent despite continuously evolving actions. Motivated by this temporal stability, we adopt an asynchronous design that reuses cached high-level reasoning across timesteps, avoiding unnecessary regeneration. This is particularly beneficial in robotic settings, where action frequencies are faster than reasoning updates. By decoupling reasoning from action generation and updating the cached reasoning trace R^c in the background, our approach enables low-latency action decoding without compromising interpretability or task performance. The asynchronous scheduling mechanism separately handles reasoning and action requests, returning actions immediately upon decoding, while reasoning is refreshed opportunistically. Full pseudocode is provided in Alg. 1.

Algorithm 1 Asynchronous Parallel Embodied Chain-of-Thought

Require: Time step t, Instruction I^t, Observation O^t, History Reasoning R^c
$c^t \leftarrow \text{encode}(I^t, O^t)$
for $i = 1$ **to** $N + 1$ **concurrently do**
    Lock R^c and autoregressively sample $r_i^t \sim P\big(r_i^t \mid c^t, R^c[:i]\big)$
    if $i = N + 1$ **then**
        $A^t = r_i^t$
    else
        Lock R^c and update R^c with r_i^t
    end if
end for
wait until A^t is finished
$A^t \leftarrow \text{decode}(A^t)$
return R^c, A^t

5 Experimental Results

In this section, we conduct experiments to evaluate the effectiveness of `Fast ECoT`. Our evaluation focuses on the following key questions: (1) To what extent does parallel reasoning generation improve computational efficiency? (2) Does the proposed method preserve task performance comparable to the sequential baseline? (3) What is the impact of reasoning step parallelization on the overall quality of reasoning?

5.1 LIBERO Simulation Experimental Setup

Task Setup. We conduct experiments using a Franka Emika Panda robotic arm within the LIBERO environment [29], a widely adopted benchmark for evaluating generalizable robotic policies. To comprehensively assess policy generalisation, we select four diverse task suites—LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long—each targeting distinct challenges, including spatial configuration, object manipulation, goal specification, and long-horizon task execution.

Training Data and Training Recipe. ECoT models require fine-tuning on each task for optimal performance [1, 4]. Training data is generated following the ECoT pipeline [4], with modifications for reproduction: we integrate Janus [30] for automated scene description and Deepseek-Reasoner [26] for generating CoT reasoning trajectories from successful demonstrations. We initialise ECoT models from the open-sourced checkpoint [4], which was pre-trained on Bridge V2 [5] and OXE dataset [6]. Then we apply LoRA [31] with rank 32 and train for $200,000$ gradient steps using a batch size of 1 distributed across 4 NVIDIA A6000 GPUs. See appendix B.1 for other hyperparameters.

Baselines. We compare our proposed method, `Fast ECoT`, against four baselines: ECoT, the original ECoT model that autoregressively generates the full reasoning chain; ECoT (5-step), a variant that updates low-level reasoning at every timestep but updates high-level reasoning only every 5 timesteps; ECoT (async), a variant originally designed to use two GPUs to asynchronously compute high-level and low-level reasoning, which we adapt to run entirely on a single GPU for fair

comparison; and ECoT (Quant), a post-training quantized version of ECoT utilizing Huggingface's Bitsandbytes [32], selected due to its best acceleration performance compared to other quantization approaches tested [19, 20]. For fairness, we exclude methods requiring additional training or architectural changes, such as speculative decoding [15, 16], and action chunking [18, 33].

5.2 LIBERO Simulation Experimental Results

Method	LIBERO-Object SR (%) ↑	LIBERO-Spatial SR (%) ↑	LIBERO-Goal SR (%) ↑	LIBERO-Long SR (%) ↑	Average SR (%) ↑	Latency Per Step (ms) ↓
ECoT	77	84	75	45	70.3	4997 ± 691
ECoT (5-step)	79	75	72	40	66.5	3514 ± 969
ECoT (Async)	70	83	80	51	71.0	3655 ± 773
ECoT (Quant)	82	82	**84**	53	75.3	2180 ± 207
`Fast ECoT`	**83**	**85**	83	**69**	**80.0**	2156 ± 353
`Fast ECoT (Async)`	75	83	83	**69**	77.5	$\mathbf{686 \pm 412}$

Table 1: LIBERO simulation experimental results. SR = Success Rate. All results are averaged over 100 trials per suite and run on a single NVIDIA RTX 4090 GPU. Best results are in **bold**.

Inference Speedup. As shown in the last column of Tab. 1, our parallel reasoning strategy significantly accelerates inference latency. Specifically, `Fast ECoT` achieves a latency reduction to 2156 ± 353 ms per step, representing over a 2.3x speed-up compared to the original ECoT baseline (4997 ± 691 ms). The asynchronous variant, `Fast ECoT (Async)`, further reduces latency to 686 ± 412 ms, nearly a 7x improvement. Notably, both variants maintain relatively stable latency compared to baselines proposed in ECoT [4].

Task Success Rate. Despite the significant reduction in latency, our method preserves or enhances task performance relative to baseline methods (see Tab. 1). `Fast ECoT` achieves the highest overall performance, with an average success rate (SR) of 80.0%, exceeding the baseline ECoT in all LIBERO suites. We hypothesise that our parallel reasoning strategy—which incorporates prior reasoning steps—helps smooth temporal inconsistencies and contributes to this performance gain. Notably, `Fast ECoT (Async)` also achieves a comparable overall performance, with an average SR of 77.5%, including the best result on LIBERO-Long (69%). The largest decline compared with normal `Fast ECoT` happens at LIBERO-Object (from 83% to 75%), where the testing scenes focus on diverse layouts of the object. The positions of visible objects lead to a greater impact on overall control performance, and the larger temporal mismatch between reasoning components and actions of `Fast ECoT (Async)` may cause this degradation. Surprisingly, ECoT (Quant) also shows decent performance improvements compared to the original `ECoT`, which can be understood as regularising model behaviour in testing environments [34].

5.3 Real-world Experiments

Setup. We validate `Fast ECoT` on a physical Franka Emika Panda robotic arm equipped with a RealSense D455 camera, providing third-person RGB-D observations. We design six manipulation tasks representative of common household scenarios. The evaluation includes both in-distribution and out-of-distribution tasks featuring unseen objects and instructions, detailed in appendix B.2. We collect 50 expert demonstrations via Droid teleoperation pipeline [35] to fine-tune ECoT models. We follow the same training data generation process, training recipe, and baselines used in Sec. 5.1.

Results. Tab. 2 summarises the performance of our method against baseline approaches on real-world tasks. `Fast ECoT` and `Fast ECoT (Async)` both demonstrate substantial latency reductions while preserving or even improving task success rates. Specifically, `Fast ECoT (Async)` delivers the best performance, with an average success rate of 70% across in-distribution (ID), out-of-distribution (OOD) objects, and OOD instruction tasks, alongside the lowest latency of 716 ± 529 ms per step—a 7.5x speedup over the original ECoT baseline (5556 ± 384 ms). This result reinforces the benefits of asynchronous parallel reasoning in real-world deployment scenarios, where low-variance and rapid inference are critical.

Method	ID SR (%) ↑	OOD Objects SR (%) ↑	OOD Instruction SR (%) ↑	Average SR (%) ↑	Latency per Step (ms) ↓
ECoT	78.3	73.3	40.0	64.0	5556 ± 384
ECoT (5-step)	56.6	51.7	35.0	47.8	4327 ± 619
ECoT (Async)	76.6	70.0	41.7	62.8	4206 ± 323
ECoT (Quant)	75.0	63.3	48.3	62.2	2437 ± 171
`Fast ECoT`	76.6	66.7	46.7	63.3	2479 ± 520
`Fast ECoT (Async)`	**83.3**	**76.7**	**50.0**	**70.0**	**716 ± 529**

Table 2: Real-world experimental results on selected household tasks. SR = Success Rate. ID = In distribution. OOD = Out of distribution. All results are averaged over 180 trials and run on a single NVIDIA RTX 4090 GPU. Best results are in **bold**.

5.4 Action Faithfulness in ECoT Reasoning

To validate that the generated chain-of-thought (CoT) not only improves performance but also *faithfully explains* the model's decision-making process, we build on the faithfulness criteria for language models [36] and introduce a novel quantitative metric - *action faithfulness (AF)* - to measure the faithfulness of the CoT reasoning for robotic tasks. fFormally, given a complete CoT consisting of N intermediate reasoning steps that culminate in a final robot action A, we enforce the model to *directly* predict an action A_i after generating only the first i reasoning steps, where $i = 0, 1, \ldots, N$. We then compute the L1 distance between A_i and A as the faithfulness score $AF_i = \|A_i - A\|_1$. Higher L1 distances indicate a greater dependence on subsequent reasoning steps, thereby suggesting higher CoT faithfulness.

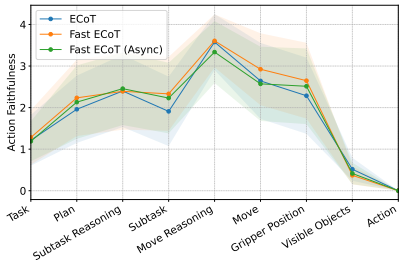


Figure 6: Action faithfulness (AF) on LIBERO tasks. All graphs are plotted with mean and standard deviation shading across 1000 steps.

We plot action faithfulness scores of `Fast ECoT` and ECoT for all reasoning steps $\{AF_i\}$ in Fig. 6. `Fast ECoT` preserves the faithfulness of the base ECoT model during parallel reasoning generation for most reasoning steps. The action faithfulness without predicting "Visible Objects" is slightly lower for `Fast ECoT`, since the high update ratio and one-step delay in visual features might increase its chances to be post-hoc. Meanwhile, the asynchronous variant of `Fast ECoT` generally exhibits a lower faithfulness, probably due to the larger temporal mismatch introduced in async reasoning. Faithfulness is notably low when no reasoning steps are generated (i.e., at "Task"), likely due to fine-tuning from the OpenVLA [1] checkpoint, which did not require reasoning steps. We provide more CoT reasoning examples in appendix C.1.

6 Conclusion

We present `Fast ECoT`, an inference-time acceleration method for Embodied Chain-of-Thought (ECoT) reasoning in VLA models. By exploiting structural and temporal locality, `Fast ECoT` enables (1) reuse of cached high-level reasoning and (2) parallel generation of modular steps. An asynchronous scheduler further decouples reasoning from action decoding, enhancing real-time responsiveness. Requiring no architectural changes or retraining, `Fast ECoT` integrates easily into existing VLA pipelines. Across simulations and real-world tests, it reduces inference latency by up to 7.5× while maintaining task performance and interpretability, advancing ECoT toward real-time deployment.

7 Limitations

While our work demonstrated that ECoT reasoning can be significantly accelerated through the reuse of previously computed thoughts, there remain several limitations. First, we still update all reasoning steps at once without identifying and prioritising the more vital steps for frequent updates. This approach can lead to excessive computation, and more sophisticated methods could dynamically determine which reasoning steps require updating based on their relative importance to task success. Second, the effectiveness of thoughts reuse inherently relies on the assumption that at least part of the reasoning remains stable over a short time window. In scenarios where reasoning steps frequently change between consecutive time steps, this stability assumption breaks down, potentially degrading the performance gains offered by our method. Third, our current approach assumes a fixed structure in reasoning chains, thus limiting its applicability to tasks or environments requiring highly dynamic or adaptive reasoning patterns. Extending thoughts reuse to accommodate more diverse and adaptive reasoning structures remains an open challenge. Addressing these limitations represents promising avenues for future research to further enhance the efficiency and generality of embodied reasoning in robotic control.

Acknowledgments

If a paper is accepted, the final camera-ready version will (and probably should) include acknowledgments. All acknowledgments go at the end of the paper, including thanks to reviewers who gave useful comments, to colleagues who contributed to the ideas, and to funding agencies and corporate sponsors that provided financial support.

References

[1] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An Open-Source Vision-Language-Action Model. In *8th Annual Conference on Robot Learning*, volume abs/2406.09246, 2024.

[2] 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, 2024. URL https://arxiv.org/abs/2410.24164.

[3] NVIDIA, 2025. URL https://d1qx31qr3h6wln.cloudfront.net/publications/GR00T%20N1%20Whitepaper.pdf.

[4] M. Zawalski, W. Chen, K. Pertsch, O. Mees, C. Finn, and S. Levine. Robotic Control via Embodied Chain-of-Thought Reasoning. In *8th Annual Conference on Robot Learning*, volume abs/2407.08693, 2024.

[5] H. Walke, K. Black, A. Lee, M. J. Kim, M. Du, C. Zheng, T. Zhao, P. Hansen-Estruch, Q. Vuong, A. He, V. Myers, K. Fang, C. Finn, and S. Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023.

[6] E. Collaboration, A. O'Neill, A. Rehman, A. Gupta, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain, A. Tung, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Gupta, A. Wang, A. Kolobov, A. Singh, A. Garg, A. Kembhavi, A. Xie, A. Brohan, A. Raffin, A. Sharma, A. Yavary, A. Jain, A. Balakrishna, A. Wahid, B. Burgess-Limerick, B. Kim, B. Schölkopf, B. Wulfe, B. Ichter, C. Lu, C. Xu, C. Le, C. Finn, C. Wang, C. Xu, C. Chi, C. Huang, C. Chan, C. Agia, C. Pan, C. Fu, C. Devin, D. Xu, D. Morton, D. Driess, D. Chen, D. Pathak, D. Shah, D. Büchler, D. Jayaraman, D. Kalashnikov, D. Sadigh, E. Johns, E. Foster, F. Liu, F. Ceola, F. Xia, F. Zhao, F. V. Frujeri, F. Stulp,

G. Zhou, G. S. Sukhatme, G. Salhotra, G. Yan, G. Feng, G. Schiavi, G. Berseth, G. Kahn, G. Yang, G. Wang, H. Su, H.-S. Fang, H. Shi, H. Bao, H. B. Amor, H. I. Christensen, H. Furuta, H. Bharadhwaj, H. Walke, H. Fang, H. Ha, I. Mordatch, I. Radosavovic, I. Leal, J. Liang, J. Abou-Chakra, J. Kim, J. Drake, J. Peters, J. Schneider, J. Hsu, J. Vakil, J. Bohg, J. Bingham, J. Wu, J. Gao, J. Hu, J. Wu, J. Wu, J. Sun, J. Luo, J. Gu, J. Tan, J. Oh, J. Wu, J. Lu, J. Yang, J. Malik, J. Silvério, J. Hejna, J. Booher, J. Tompson, J. Yang, J. Salvador, J. J. Lim, J. Han, K. Wang, K. Rao, K. Pertsch, K. Hausman, K. Go, K. Gopalakrishnan, K. Goldberg, K. Byrne, K. Oslund, K. Kawaharazuka, K. Black, K. Lin, K. Zhang, K. Ehsani, K. Lekkala, K. Ellis, K. Rana, K. Srinivasan, K. Fang, K. P. Singh, K.-H. Zeng, K. Hatch, K. Hsu, L. Itti, L. Y. Chen, L. Pinto, L. Fei-Fei, L. Tan, L. J. Fan, L. Ott, L. Lee, L. Weihs, M. Chen, M. Lepert, M. Memmel, M. Tomizuka, M. Itkina, M. G. Castro, M. Spero, M. Du, M. Ahn, M. C. Yip, M. Zhang, M. Ding, M. Heo, M. K. Srirama, M. Sharma, M. J. Kim, N. Kanazawa, N. Hansen, N. Heess, N. J. Joshi, N. Suenderhauf, N. Liu, N. D. Palo, N. M. M. Shafiullah, O. Mees, O. Kroemer, O. Bastani, P. R. Sanketi, P. T. Miller, P. Yin, P. Wohlhart, P. Xu, P. D. Fagan, P. Mitrano, P. Sermanet, P. Abbeel, P. Sundaresan, Q. Chen, Q. Vuong, R. Rafailov, R. Tian, R. Doshi, R. Mart'in-Mart'in, R. Baijal, R. Scalise, R. Hendrix, R. Lin, R. Qian, R. Zhang, R. Mendonca, R. Shah, R. Hoque, R. Julian, S. Bustamante, S. Kirmani, S. Levine, S. Lin, S. Moore, S. Bahl, S. Dass, S. Sonawani, S. Tulsiani, S. Song, S. Xu, S. Haldar, S. Karamcheti, S. Adebola, S. Guist, S. Nasiriany, S. Schaal, S. Welker, S. Tian, S. Ramamoorthy, S. Dasari, S. Belkhale, S. Park, S. Nair, S. Mirchandani, T. Osa, T. Gupta, T. Harada, T. Matsushima, T. Xiao, T. Kollar, T. Yu, T. Ding, T. Davchev, T. Z. Zhao, T. Armstrong, T. Darrell, T. Chung, V. Jain, V. Kumar, V. Vanhoucke, W. Zhan, W. Zhou, W. Burgard, X. Chen, X. Chen, X. Wang, X. Zhu, X. Geng, X. Liu, X. Liangwei, X. Li, Y. Pang, Y. Lu, Y. J. Ma, Y. Kim, Y. Chebotar, Y. Zhou, Y. Zhu, Y. Wu, Y. Xu, Y. Wang, Y. Bisk, Y. Dou, Y. Cho, Y. Lee, Y. Cui, Y. Cao, Y.-H. Wu, Y. Tang, Y. Zhu, Y. Zhang, Y. Jiang, Y. Li, Y. Li, Y. Iwasawa, Y. Matsuo, Z. Ma, Z. Xu, Z. J. Cui, Z. Zhang, Z. Fu, and Z. Lin. Open x-embodiment: Robotic learning datasets and rt-x models, 2024. URL https://arxiv.org/abs/2310.08864.

[7] H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning, 2023. URL https://arxiv.org/abs/2304.08485.

[8] S. Karamcheti, S. Nair, A. Balakrishna, P. Liang, T. Kollar, and D. Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models, 2024.

[9] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. Chi, Q. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

[10] X. Ning, Z. Lin, Z. Zhou, Z. Wang, H. Yang, and Y. Wang. Skeleton-of-thought: Prompting llms for efficient parallel generation, 2024. URL https://arxiv.org/abs/2307.15337.

[11] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, D. Ho, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, E. Jang, R. J. Ruano, K. Jeffrey, S. Jesmonth, N. Joshi, R. C. Julian, D. Kalashnikov, Y. Kuang, K.-H. Lee, S. Levine, Y. Lu, L. Luu, C. Parada, P. Pastor, J. Quiambao, K. Rao, J. Rettinghouse, D. Reyes, P. Sermanet, N. Sievers, C. Tan, A. Toshev, V. Vanhoucke, F. Xia, T. Xiao, P. Xu, S. Xu, and M. Yan. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances. In *Conference on Robot Learning*, pages 287–318, 2022.

[12] A. Zeng, M. Attarian, B. Ichter, K. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. Ryoo, V. Sindhwani, J. Lee, V. Vanhoucke, and P. Florence. Socratic models: Composing zero-shot multimodal reasoning with language, 2022. URL https://arxiv.org/abs/2204.00598.

[13] O. Mees, J. Borja-Diaz, and W. Burgard. Grounding language with visual affordances over unstructured data, 2023. URL https://arxiv.org/abs/2210.01911.

[14] Q. Sun, P. Hong, T. D. Pala, V. Toh, U.-X. Tan, D. Ghosal, and S. Poria. Emma-x: An embodied multimodal action model with grounded chain of thought and look-ahead spatial reasoning, 2024. URL https://arxiv.org/abs/2412.11974.

[15] Y. Leviathan, M. Kalman, and Y. Matias. Fast inference from transformers via speculative decoding, 2023. URL https://arxiv.org/abs/2211.17192.

[16] S. Kim, K. Mangalam, S. Moon, J. Malik, M. W. Mahoney, A. Gholami, and K. Keutzer. Speculative decoding with big little decoder, 2023. URL https://arxiv.org/abs/2302.07863.

[17] W. Song, J. Chen, P. Ding, H. Zhao, W. Zhao, Z. Zhong, Z. Ge, J. Ma, and H. Li. Accelerating vision-language-action model integrated with action chunking via parallel decoding, 2025. URL https://arxiv.org/abs/2503.02310.

[18] M. J. Kim, C. Finn, and P. Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. *arXiv.org*, abs/2502.19645, 2025.

[19] J. Lin, J. Tang, H. Tang, S. Yang, W.-M. Chen, W.-C. Wang, G. Xiao, X. Dang, C. Gan, and S. Han. Awq: Activation-aware weight quantization for llm compression and acceleration, 2024. URL https://arxiv.org/abs/2306.00978.

[20] G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han. Smoothquant: Accurate and Efficient Post-Training Quantization for Large Language Models. In *International Conference on Machine Learning (ICML)*, pages 38087–38099, 2023.

[21] huggingface. Bitsandbytes, 2025. URL https://huggingface.co/docs/bitsandbytes/main/en/index.

[22] G.-I. Yu, J. S. Jeong, G.-W. Kim, S. Kim, and B.-G. Chun. Orca: A distributed serving system for Transformer-Based generative models. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 521–538, Carlsbad, CA, July 2022. USENIX Association. ISBN 978-1-939133-28-1. URL https://www.usenix.org/conference/osdi22/presentation/yu.

[23] n. Nvidia. Nvidia/tensorrt-llm: Tensorrt-llm provides users with an easy-to-use python api to define large language models (llms) and support state-of-the-art optimizations to perform inference efficiently on nvidia gpus. tensorrt-llm also contains components to create python and c++ runtimes that orchestrate the inference execution in performant way., 2025. URL https://github.com/NVIDIA/TensorRT-LLM.

[24] W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica. Efficient memory management for large language model serving with pagedattention, 2023. URL https://arxiv.org/abs/2309.06180.

[25] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

[26] DeepSeek-AI, D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi, X. Zhang, X. Yu, Y. Wu, Z. F. Wu, Z. Gou, Z. Shao, Z. Li, Z. Gao, A. Liu, B. Xue, B. Wang, B. Wu, B. Feng, C. Lu, C. Zhao, C. Deng, C. Zhang, C. Ruan, D. Dai, D. Chen, D. Ji, E. Li, F. Lin, F. Dai, F. Luo, G. Hao, G. Chen, G. Li, H. Zhang, H. Bao, H. Xu, H. Wang, H. Ding, H. Xin, H. Gao, H. Qu, H. Li, J. Guo, J. Li, J. Wang, J. Chen, J. Yuan, J. Qiu, J. Li, J. L. Cai, J. Ni, J. Liang, J. Chen, K. Dong, K. Hu, K. Gao, K. Guan, K. Huang, K. Yu, L. Wang, L. Zhang, L. Zhao, L. Wang, L. Zhang, L. Xu, L. Xia, M. Zhang, M. Zhang, M. Tang, M. Li, M. Wang, M. Li, N. Tian, P. Huang, P. Zhang, Q. Wang, Q. Chen, Q. Du, R. Ge, R. Zhang, R. Pan, R. Wang, R. J. Chen, R. L. Jin, R. Chen, S. Lu, S. Zhou, S. Chen, S. Ye,

S. Wang, S. Yu, S. Zhou, S. Pan, S. S. Li, S. Zhou, S. Wu, S. Ye, T. Yun, T. Pei, T. Sun, T. Wang, W. Zeng, W. Zhao, W. Liu, W. Liang, W. Gao, W. Yu, W. Zhang, W. L. Xiao, W. An, X. Liu, X. Wang, X. Chen, X. Nie, X. Cheng, X. Liu, X. Xie, X. Liu, X. Yang, X. Li, X. Su, X. Lin, X. Q. Li, X. Jin, X. Shen, X. Chen, X. Sun, X. Wang, X. Song, X. Zhou, X. Wang, X. Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. Zhang, Y. Xu, Y. Li, Y. Zhao, Y. Sun, Y. Wang, Y. Yu, Y. Zhang, Y. Shi, Y. Xiong, Y. He, Y. Piao, Y. Wang, Y. Tan, Y. Ma, Y. Liu, Y. Guo, Y. Ou, Y. Wang, Y. Gong, Y. Zou, Y. He, Y. Xiong, Y. Luo, Y. You, Y. Liu, Y. Zhou, Y. X. Zhu, Y. Xu, Y. Huang, Y. Li, Y. Zheng, Y. Zhu, Y. Ma, Y. Tang, Y. Zha, Y. Yan, Z. Z. Ren, Z. Ren, Z. Sha, Z. Fu, Z. Xu, Z. Xie, Z. Zhang, Z. Hao, Z. Ma, Z. Yan, Z. Wu, Z. Gu, Z. Zhu, Z. Liu, Z. Li, Z. Xie, Z. Song, Z. Pan, Z. Huang, Z. Xu, Z. Zhang, and Z. Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[27] P. Lu, B. Peng, H. Cheng, M. Galley, K.-W. Chang, Y. N. Wu, S.-C. Zhu, and J. Gao. Chameleon: Plug-and-play compositional reasoning with large language models, 2023. URL https://arxiv.org/abs/2304.09842.

[28] G. Xu, P. Jin, H. Li, Y. Song, L. Sun, and L. Yuan. Llava-cot: Let vision language models reason step-by-step, 2025. URL https://arxiv.org/abs/2411.10440.

[29] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023.

[30] X. Chen, Z. Wu, X. Liu, Z. Pan, W. Liu, Z. Xie, X. Yu, and C. Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025.

[31] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

[32] huggingface. Bitsandbytes, 2025. URL https://huggingface.co/docs/bitsandbytes/main/en/index.

[33] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023. URL https://arxiv.org/abs/2304.13705.

[34] S. Krishnan, M. Lam, S. Chitlangia, Z. Wan, G. Barth-maron, A. Faust, and V. J. Reddi. QuaRL: Quantization for fast and environmentally sustainable reinforcement learning. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL https://openreview.net/forum?id=xwWsiFmUEs.

[35] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, P. D. Fagan, J. Hejna, M. Itkina, M. Lepert, Y. J. Ma, P. T. Miller, J. Wu, S. Belkhale, S. Dass, H. Ha, A. Jain, A. Lee, Y. Lee, M. Memmel, S. Park, I. Radosavovic, K. Wang, A. Zhan, K. Black, C. Chi, K. B. Hatch, S. Lin, J. Lu, J. Mercat, A. Rehman, P. R. Sanketi, A. Sharma, C. Simpson, Q. Vuong, H. R. Walke, B. Wulfe, T. Xiao, J. H. Yang, A. Yavary, T. Z. Zhao, C. Agia, R. Baijal, M. G. Castro, D. Chen, Q. Chen, T. Chung, J. Drake, E. P. Foster, J. Gao, V. Guizilini, D. A. Herrera, M. Heo, K. Hsu, J. Hu, M. Z. Irshad, D. Jackson, C. Le, Y. Li, K. Lin, R. Lin, Z. Ma, A. Maddukuri, S. Mirchandani, D. Morton, T. Nguyen, A. O'Neill, R. Scalise, D. Seale, V. Son, S. Tian, E. Tran, A. E. Wang, Y. Wu, A. Xie, J. Yang, P. Yin, Y. Zhang, O. Bastani, G. Berseth, J. Bohg, K. Goldberg, A. Gupta, A. Gupta, D. Jayaraman, J. J. Lim, J. Malik, R. Martín-Martín, S. Ramamoorthy, D. Sadigh, S. Song, J. Wu, M. C. Yip, Y. Zhu, T. Kollar, S. Levine, and C. Finn. Droid: A large-scale in-the-wild robot manipulation dataset, 2025. URL https://arxiv.org/abs/2403.12945.

[36] Q. Lyu, S. Havaldar, A. Stein, L. Zhang, D. Rao, E. Wong, M. Apidianaki, and C. Callison-Burch. Faithful chain-of-thought reasoning. *ArXiv*, abs/2301.13379, 2023. URL https://api.semanticscholar.org/CorpusID:256416127.

[37] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Łukasz Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Łukasz Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O'Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. de Avila Belbute Peres, M. Petrov, H. P. de Oliveira Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

A Extra Algorithm Details

A.1 Algorithm on Embodied Chain-of-Thought

Algorithm 2 Embodied Chain-of-Thought

Require: Time step t, Instruction I^t, Observation O^t
$c^t \leftarrow \text{encode}(I^t, O^t)$
$R^t \leftarrow [\,]$
for $i = 1$ **to** $N + 1$ **do**
    Autoregressively sample $r_i^t \sim P\big(r_i^t \mid c^t, R^t\big)$
    Add r_i^t to R^t
end for
$R^t, A^t \leftarrow \text{decode}(R^t[:-1], R^t[-1])$
return R^t, A^t

A.2 Algorithm on Parallel Embodied Chain-of-Thought

Algorithm 3 Parallel Embodied Chain-of-Thought

Require: Time step t, Instruction I^t, Observation O^t, Last reasoning steps R^{t-1}
$c^t \leftarrow \text{encode}(I^t, O^t)$
$R^t \leftarrow [\,]$
for $i = 1$ **to** $N + 1$ **concurrently do**
    Autoregressively sample $r_i^t \sim P\big(r_i^t \mid c^t, R^{t-1}[:i]\big)$
end for
Synchronize
$R^t, A^t \leftarrow \text{decode}(R^t[:-1], R^t[-1])$
return R^t, A^t

B Extra Experimental Setup

B.1 Finetuning Hyperparameters

Hyperparameter	Value
Base Model	`ecot-openvla-7b-oxe`
Number of GPUs	4
Batch Size	1
Gradient Accumulation Steps	1
Learning Rate	5e-4
Learning Rate Scheduler	constant LR
Optimizer	AdamW
AdamW Betas	(0.9, 0.999)
AdamW Epsilon	1e-8
Training Steps	200,000
Image Augmentation	
Reasoning Dropout	0.0
LoRA Rank	32
LoRA Alpha	16
LoRA Dropout	0.0

Table 3: Fine-tuning hyperparameters used for fine-tuning EcoT model.

B.2 Real-World Experimental Setup

Out-of-distribution Instructions. We use GPT-4o [37] to rephrase instructions from the training set. These rephrasings modify both the verbs and nouns to create linguistically novel yet semantically equivalent instructions, allowing us to test generalization to unseen language.

Original Instruction	OOD Rephrased Instruction
Pick up the gray spanner and put it in the basket	Grab the silver wrench and drop it into the bin
Pick up the blue hammer and put it in the basket	Take the cobalt mallet and place it in the container
Pick up the blue squash ball and put it in the basket	Retrieve the navy rubber ball and set it in the basket
Pick up the white baseball and put it in the basket	Grab the pale softball and store it in the bucket
Pick up the banana and put it on the plate	Take the yellow fruit and lay it on the dish
Pick up the lemon and put it in the cup	Seize the citrus and drop it into the mug

Table 4: Original instructions and their corresponding out-of-distribution (OOD) rephrasings generated by GPT-4o.

In-distribution Objects.

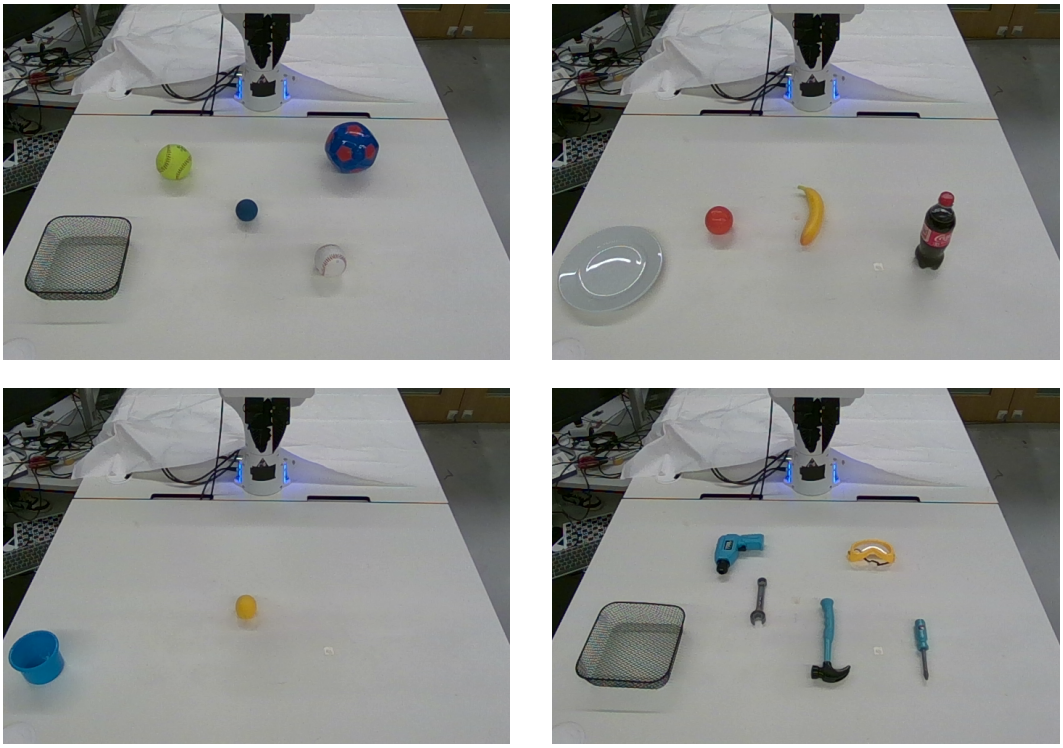


Figure 7: Qualitative examples of in-distribution (ID) objects used for evaluation.

Out-of-distribution objects.

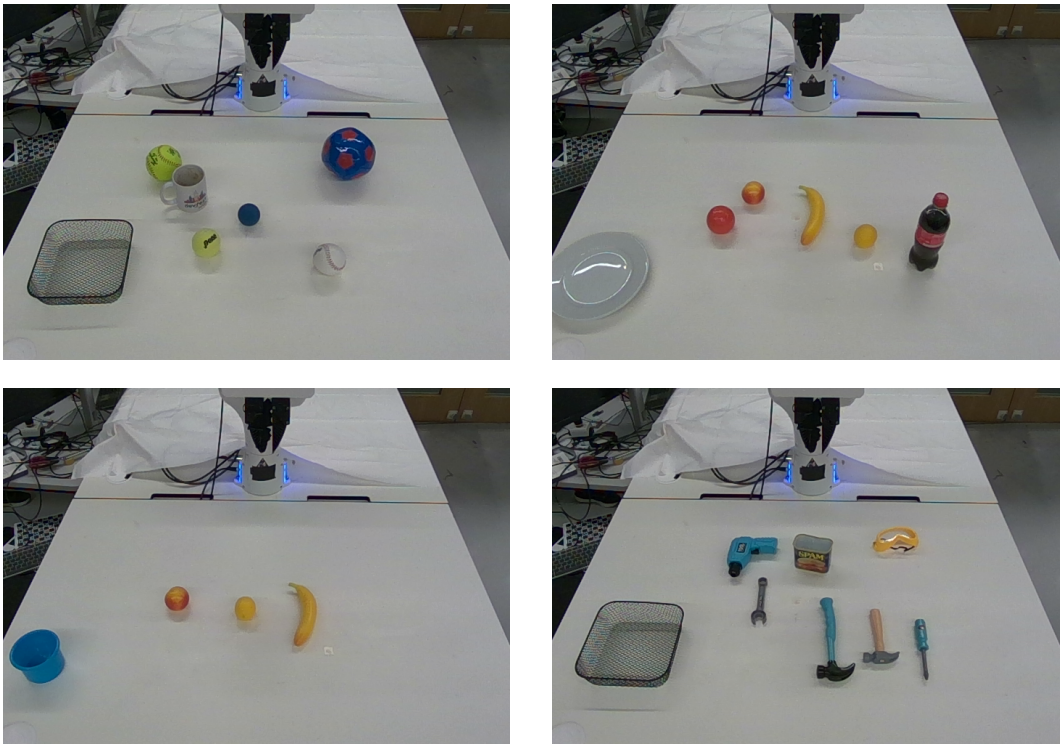
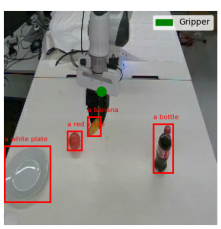


Figure 8: Qualitative examples of out-of-distribution (OOD) objects used for evaluation.

C Extra Experimental Results

C.1 CoT Reasoning Examples



INSTRUCTION: pick up the banana and put it on the plate

TASK: Transport the banana to the plate and release it.

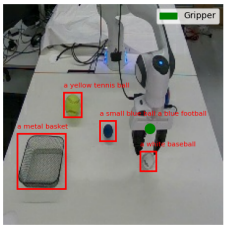
PLAN: 1. Release banana. 2. Retract arm.

SUBTASK REASONING: Descending vertically to place the banana on the plate.

SUBTASK: Lower banana onto plate.

MOVE REASONING: Upward tilt to level the banana for stable placement.

MOVE: move forward left, tilt up, rotate counterclockwise



INSTRUCTION: pick up the white baseball and put it in the basket

TASK: Pick up the white baseball and place it in the basket on the left.

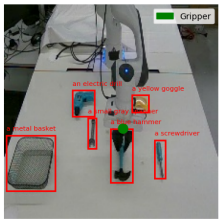
PLAN: 1. Lift. 2. Transport left. 3. Release.

SUBTASK REASONING: Begin ascending to clear the table.

SUBTASK: Lift the ball

MOVE REASONING: Final stability check before lifting.

MOVE: stop



INSTRUCTION: pick up the blue hammer and put it in the basket

TASK: Pick up the blue hammer and place it in the basket

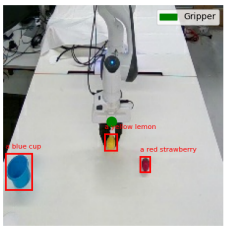
PLAN: 1. Align with the hammer. 2. Descend and open gripper. 3. Grasp and lift. 4. Transport to basket. 5. Release.

SUBTASK REASONING: Continue rotation to avoid colliding with the hook.

SUBTASK: Adjust gripper orientation

MOVE REASONING: Refine orientation while descending.

MOVE: move up, tilt down, open gripper



INSTRUCTION: pick up the lemon and put it in the cup

TASK: Pick up the lemon and place it in the cup.

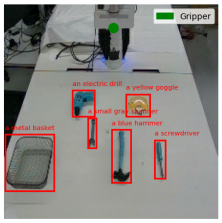
PLAN: Lift → Transport → Release → Retract.

SUBTASK REASONING: Continuing ascent to clear the lemon above table obstacles.

SUBTASK: Lift

MOVE REASONING: Ensuring sufficient clearance from the red object to the right.

MOVE: move up, tilt down, open gripper



INSTRUCTION: pick up the gray spanner and put it in the basket

TASK: Pick up the gray spanner and place it in the basket.

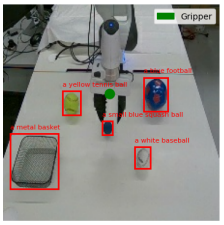
PLAN: 1. Approach spanner. 2. Grasp spanner. 3. Transport to basket. 4. Release and retract.

SUBTASK REASONING: Confirming the spanner's location relative to the yellow semi-circular object and basket.

SUBTASK: Initialize positioning

MOVE REASONING: Waiting for path clearance between the microscope and drill.

MOVE: stop



INSTRUCTION: pick up the blue squash ball and put it in the basket

TASK: Pick up the blue squash ball and place it in the basket.

PLAN: 1. Approach the ball; 2. Position gripper; 3. Grasp the ball; 4. Move to the basket; 5. Release the ball

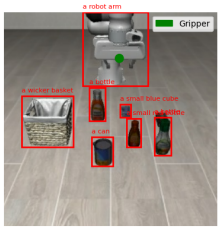
SUBTASK REASONING: Fine-tune alignment to prevent collision with the small blue sphere in the center.

SUBTASK: Approach the ball

MOVE REASONING: Descend while maintaining gripper alignment.

MOVE: move down

Figure 9: Qualitative examples of ECOT reasoning in real-world experiments.



INSTRUCTION: pick up the bbq sauce and place it in the basket

TASK: Pick up the BBQ sauce and place it in the basket

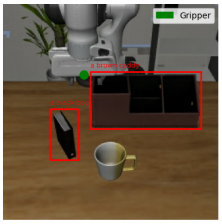
PLAN: 2. Move forward to BBQ sauce; 3. Lower gripper; 4. Close gripper; 5. Lift and rotate toward basket; 6. Move right to basket; 7. Lower and release

SUBTASK REASONING: The BBQ sauce is positioned ahead of the arm. Forward motion brings the gripper into alignment.

SUBTASK: Move forward to BBQ sauce

MOVE REASONING: Begin horizontal approach to minimize vertical adjustments later.

MOVE: move forward



INSTRUCTION: pick up the book and place it in the back compartment of the caddy

TASK: Pick up the book and place it in the caddy.

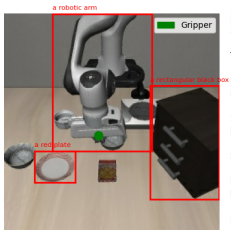
PLAN: 1. Tilt down to approach. 2. Move down to grip. 3. Close gripper. 4. Retract upward. 5. Move left to caddy. 6. Tilt for placement. 7. Move down to release.

SUBTASK REASONING: The gripper must lower vertically to avoid colliding with the book's edges. Tilting adjusts the end-effector angle for a perpendicular grasp.

SUBTASK: Tilt down to approach the book

MOVE REASONING: Continuous tilting ensures gradual alignment with the book's surface.

MOVE: tilt down



INSTRUCTION: pick up the black bowl from table center and place it on the plate

TASK: Place the bowl on the white plate.

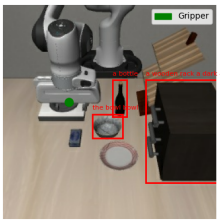
PLAN: Grasp bowl, ascend, move to plate, position, release, retract.

SUBTASK REASONING: The gripper is in contact with the bowl; closing secures it without slippage.

SUBTASK: Grasp bowl.

MOVE REASONING: Apply gradual pressure to grip the bowl firmly.

MOVE: stop



INSTRUCTION: put the cream cheese in the bowl

TASK: Put cream cheese in the bowl

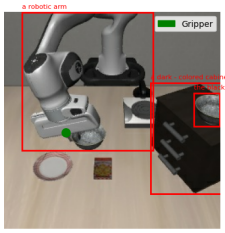
PLAN: 1. Rotate clockwise to face cream cheese. 2. Move forward. 3. Descend and grasp.

SUBTASK REASONING: The cream cheese is positioned to the left of the stove; clockwise rotation orients the gripper toward it.

SUBTASK: Rotate clockwise to align

MOVE REASONING: Begin aligning the arm's base joint to face the target.

MOVE: move down, rotate counterclockwise



INSTRUCTION: pick up the black bowl on the cookie box and place it on the plate

TASK: Place the bowl on the plate.

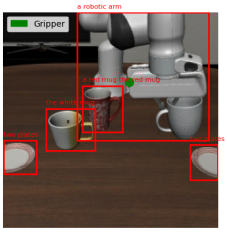
PLAN: 5. Lift and transport to the plate. 6. Lower and release.

SUBTASK REASONING: The plate is located to the right. Moving right while ascending avoids collisions with the cookie box.

SUBTASK: Transporting to Plate

MOVE REASONING: Diagonal movement balances horizontal travel with maintaining safe altitude.

MOVE: move right



INSTRUCTION: put the white mug on the left plate and put the yellow and white mug on the right plate

TASK: Put the white mug on the left plate and the yellow and white mug on the right plate.

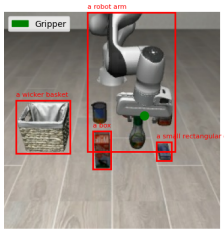
PLAN: [1. Grasp the yellow and white mug, 2. Transport it to the right plate, 3. Place the white mug on the left plate]

SUBTASK REASONING: The gripper must close firmly to prevent slippage during transport.

SUBTASK: Secure the yellow and white mug.

MOVE REASONING: Simultaneous tilt and closure to lock the mug in place.

MOVE: stop



INSTRUCTION: pick up the salad dressing and place it in the basket

TASK: Pick up the salad dressing and place it in the basket.

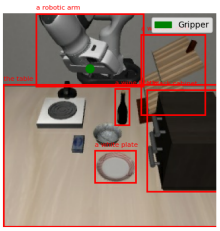
PLAN: 1. Continue approach. 2. Descend. 3. Grasp. 4. Lift. 5. Transport. 6. Lower and release.

SUBTASK REASONING: Maintain forward motion to close proximity to the bottle.

SUBTASK: Continue approach

MOVE REASONING: Sustained movement avoids stop-and-go delays.

MOVE: stop



INSTRUCTION: put the wine bottle on the rack

TASK: Approach bottle

PLAN: Maintain tilt to avoid floor

SUBTASK REASONING: Gripper must clear floor obstacles during extension.

SUBTASK: Adjust elevation

MOVE REASONING: Continue extending arm while raising gripper height.

MOVE: tilt up, rotate counterclockwise

Figure 10: Qualitative examples of ECOT reasoning in simulation. As the VLMs used in both the data generation pipeline and the underlying VLA model are pretrained on real-world images, their performance in simulated environments is **less stable**. This can lead to inaccurate bounding boxes, undetected objects, and unusual or incorrect object labels.