TCPO: Thought-Centric Preference Optimization for Effective Embodied Decision-making

Kechen Jiao[1 2*‡], **Zhirui Fang**[1*], **Jiahao Liu**[2], **Bei Li**[2], **Qifan Wang**[5], **Xinyu Liu**[3], **Junhao Ruan**[3], **Zhongjian Qiao**[1], **Yifan Zhu**[4], **Yaxin Xu**[6], **Jingang Wang**[2], **Xiu Li**[1†]
[1]Tsinghua University, [2]Meituan [3]Northeastern University
[4]Beijing University of Posts and Telecommunications
[5]Meta AI, [6]Wuhan University

Abstract

Using effective generalization capabilities of vision language models (VLMs) in context-specific dynamic tasks for embodied artificial intelligence remains a significant challenge. Although supervised fine-tuned models can better align with the real physical world, they still exhibit sluggish responses and hallucination issues in dynamically changing environments, necessitating further alignment. Existing post-SFT methods, reliant on reinforcement learning and chain-of-thought (CoT) approaches, are constrained by sparse rewards and action-only optimization, resulting in low sample efficiency, poor consistency, and model degradation. To address these issues, this paper proposes Thought-Centric Preference Optimization (TCPO) for effective embodied decision-making. Specifically, TCPO introduces a step-wise preference-based optimization approach, transforming sparse reward signals into richer step sample pairs. It emphasizes the alignment of the model's intermediate reasoning process, mitigating the problem of model degradation. Moreover, by incorporating Action Policy Consistency Constraint (APC), it further imposes consistency constraints on the model output. Experiments in the ALFWorld environment demonstrate an average success rate of **26.67%**, achieving a **6%** improvement over RL4VLM and validating the effectiveness of our approach in mitigating model degradation after fine-tuning. These results highlight the potential of integrating preference-based learning techniques with CoT processes to enhance the decision-making capabilities of vision-language models in embodied agents.

1 Introduction

Large Language Models (LLMs) and Large Multi-modal Models (LMMs) have demonstrated exceptional capabilities in natural language understanding and generation (Brown, 2020; Achiam et al., 2023). Recent advances extend their applications to managing AI models for complex multi-modal tasks (Shen et al., 2024; Lu et al., 2024), mastering strategic games like TextWorld (Yao et al., 2022), Handi (Hu and Sadigh, 2023), and Minecraft (Wang et al., 2023a), as well as enabling robotic interactions through physical deployments (Ahn et al., 2022; Driess et al., 2023; Ahn et al., 2022).

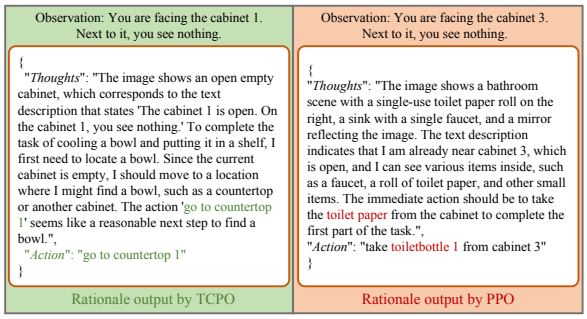


Figure 1: Comparison results of our TCPO and PPO methods. In our TCPO method, we emphasize the logical consistency of actions generated by rationale and incorporate an Action Probability Consistency constraint (APC). In contrast, traditional PPO methods may compromise consistency during training, resulting in the generation of illegal actions, as shown on the right.

Embodied AI research has predominantly concentrated on developing foundational models to augment semantic comprehension and operational capacities of LLMs and multi-modal systems through robotic sensory inputs (Mu et al., 2023; Kim et al., 2024b; Xu et al., 2024). These initiatives aim to bridge the disconnect between pre-trained models' knowledge and physical environments, typically employing supervised fine-tuning (SFT) or LoRA adaption techniques (Hu et al., 2021) to improve visual understanding, planning proficiency, and action strategy generation from multi-modal inputs. However, such static adaptation approaches prove inadequate for dynamic environments, prompting development of two post-

*Equal Contribution
†Corresponding author
‡Work performed while an intern at Meituan.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 9586–9600
November 4-9, 2025

SFT enhancement strategies: dynamic replanning and reinforcement learning integration.

Replanning methodologies address environmental dynamics through chain-of-thought reasoning and task decomposition (Mu et al., 2023; Song et al., 2023), enabling real-time plan updates when environmental states change. This approach introduces adaptive error correction and contingency handling to static planning frameworks. Reinforcement learning extensions further align models with dynamic requirements through various implementations. LLaRP (Szot et al., 2024) integrates policy heads into language models, RL4VLM (Zhai et al., 2024) employs Proximal Policy Optimization (PPO) (Schulman et al., 2017) for decision-making, and TWOSOME (Tan et al., 2024) aligns action probabilities using reinforcement principles (Sutton, 2018). Despite these advancements, practical deployment faces two critical challenges: 1) *the prevalence of sparse environmental rewards that escalate exploration costs*, and 2) *the inherent conflict between reinforcement optimization and linguistic consistency preservation*. As demonstrated in Figure 1, conventional reinforcement paradigms that optimize action probabilities or joint thought-action alignment tend to disrupt internal linguistic coherence of models, ultimately degrading response quality despite improved environmental adaptation.

To address these challenges, we propose that optimization should focus on enhancing the quality of Chain-of-Thought (CoT) reasoning rather than final actions, as strategic decisions inherently emerge from this cognitive process. Our solution leverages a step-wise Direct Preference Optimization (DPO) framework to maximize sample efficiency. Unlike conventional reinforcement learning requiring dense rewards, preference learning effectively utilizes entire trajectories – including zero-return samples through negative pair construction – demonstrating enhanced learning capacity for sparse-reward scenarios and long-horizon tasks. We introduce **Thought-Centric Preference Optimization (TCPO)**, a paradigm prioritizing rationale refinement over action selection to address composite error propagation in multi-step reasoning while strengthening step-wise determinism (see Section 3). Experiments verify the superior capability of TCPO in learning deterministic strategies and resolving credit assignment challenges. Furthermore, we establish the **Action Policy Consistency Constraint (APC)** to preserve the model's intrinsic consistency, ensuring actions strictly derive from CoT processes through constrained policy optimization. Our main contributions can be summarized as following:

- We present **TCPO**, an algorithmic framework employing stepwise alignment methodology to coordinate the CoT process in embodied agents via environmental interactions. The framework strengthens model coherence through strategic determinism optimization while maintaining online adaptability.
- We introduce the novel Action Policy Consistency Constraint (APC), enforcing alignment with the pre-trained model's action conditional distributions to address policy consistency deterioration during online adaptation.
- Our experimental evaluation on GymCards and ALFWorld demonstrate that the proposed approach achieves a **6%** improvement in average task success rate compared to state-of-the-art RL4VLM baselines.

2 Related Work

Embodied Agent with LLMs Recent works highlight the importance of LLMs in interaction and decision-making (Abramson et al., 2020; Karamcheti et al., 2022; Li et al., 2022), and their application in robot navigation (Parisi et al., 2022; Hong et al., 2021; Majumdar et al., 2020) and manipulation (Jiang et al., 2022; Ren et al., 2023; Karamcheti et al., 2022). A growing body of research leverages LLMs to enhance planning and reasoning in embodied agents. SayCan (Ahn et al., 2022) combines LLM probabilities with a value function to assess candidate actions. Zeng et al. (2022) integrate LLMs with visual-language models and pre-trained language-conditioned policies (Shridhar et al., 2022) for open vocabulary tasks. Huang et al. (2022a) show that LLMs can plan and execute household tasks by grounding actions to a predefined list. Inner Monologue (Huang et al., 2022b) extends SayCan with a closed-loop principle, also applied in works like (Yao et al., 2023; Huang et al., 2022b; Kim et al., 2024a; Singh et al., 2023; Liang et al., 2023; Shinn et al., 2023; Wang et al., 2023b) to refine plans based on environment feedback for tasks such as automation and Minecraft. Approaches like (Zheng et al., 2023) use LLMs to generate temporal-abstracted actions, while Dasgupta et al. (2023) employ LLMs for planning and success detection in RL-trained agents.

While these methods show strong results, they depend heavily on powerful LLMs like GPT-4 and PaLM (Chowdhery et al., 2023), which may not be suitable for smaller models like LLaMA-7B with weaker reasoning abilities.

Similarly, GLAM (Carta et al., 2023) uses RL finetuning for grounding LLMs but focuses on simple actions (e.g., turn left, go forward) in toy environments like BabyAI (Chevalier-Boisvert et al., 2018), using a much smaller LLM (Flan-T5-780M). These simple actions, with fewer tokens and less meaningful semantics, underutilize LLM capabilities and fail to address prompt design issues and action space imbalance, leading to instability and poor robustness.

Preference Learning Preference learning has become a key area in machine learning, focusing on developing models that capture human preferences from observational data. Current preference learning methods are typically categorized into pointwise, pairwise, and listwise approaches. Among these, Direct Preference Optimization (DPO) (Rafailov et al., 2024) has emerged as a novel approach, directly optimizing user preferences without intermediary ranking steps. This method enhances alignment with user preferences by constructing loss functions that reflect them directly. Chen et al. (2024) introduces OPTune, an efficient online preference tuning method in RLHF. OPTune improves training speed and model alignment by selectively regenerating low-reward responses and focusing on response pairs with larger reward gaps using a weighted DPO loss.

Recent studies have expanded DPO's applications. Step-DPO (Lai et al., 2024) enhances DPO for tasks requiring long-chain reasoning, like mathematical problem-solving, by optimizing individual reasoning steps and improving both factuality and reasoning in large language models. Pal et al. (2024) advanced DPO's practical applications in sentiment-aware recommendations through DPO-Positive, which integrates sentiment information into the recommendation process, leading to more accurate and user-aligned outcomes.

3 Methodology

Our proposed Thought-Centric Preference Optimization (TCPO) framework employs a replanning-enabled algorithmic architecture to ensure robust adaptability in dynamic environments, comprising two core components. The *Preference-Aware Fine-Tuning* component introduces a stepwise preference learning mechanism that reframes the alignment task as a cross-entropy-guided classification problem, allowing for dense preference supervision and more efficient policy optimization. Sample efficiency is further improved through trajectory repurposing, where typically discarded zero-return trajectories are leveraged to generate auxiliary training pairs via contrastive sampling. The *Action Policy Consistency Constraints* component enforces coherence between intermediate reasoning states and final action outputs, effectively mitigating model degradation observed in conventional chain-of-thought approaches. We present the two components in the following subsections.

3.1 Sample Pairs Construction

Unlike traditional MLP-based policy networks restricted to predefined action spaces, VLM policies exhibit unique advantages through their natural language generation capabilities. This enables explicit CoT reasoning that facilitates systematic environment exploration via intermediate rationalization steps preceding final action selection. However, RL-based fine-tuning of VLM policies π_θ introduces critical challenges arising from sparse reward signals. Specifically, the episodic nature of embodied interactions yields predominantly non-informative state transitions where reward feedback $r_t = 0$ for most timesteps t. Under standard PPO frameworks, these zero-reward transitions provide negligible gradient signals due to their dependence on advantage estimation, resulting in suboptimal sample efficiency during policy adaptation. While conventional approaches often resort to manual reward shaping to mitigate sparsity, our method addresses this through contrastive trajectory pair construction. For any timestep t, we generate preference tuples:

$$\begin{aligned}\mathcal{P}_t &= \langle \tau^t_{\text{win}}, \tau^t_{\text{lose}} \rangle \\ &= \left\langle \left\{ a_t^{(1)}, r_t^{(1)}, \tau_{1:t-1} \right\}, \left\{ a_t^{(2)}, r_t^{(2)}, \tau_{1:t-1} \right\} \right\rangle \end{aligned} \tag{1}$$

where τ^t_{win} denotes the preferred trajectory segment with comparatively higher reward $r_t^{(1)} > r_t^{(2)}$. This construction transforms sparse scalar rewards into relative preference rankings across trajectory segments, enabling three key advancements: (1) Effective utilization of suboptimal transitions where $r_t^{(i)} > 0$ but $r_t^{(i)} \ll r_{\max}$ through contrastive pairings; (2) Amplification of policy update signals via pairwise comparisons rather than absolute reward

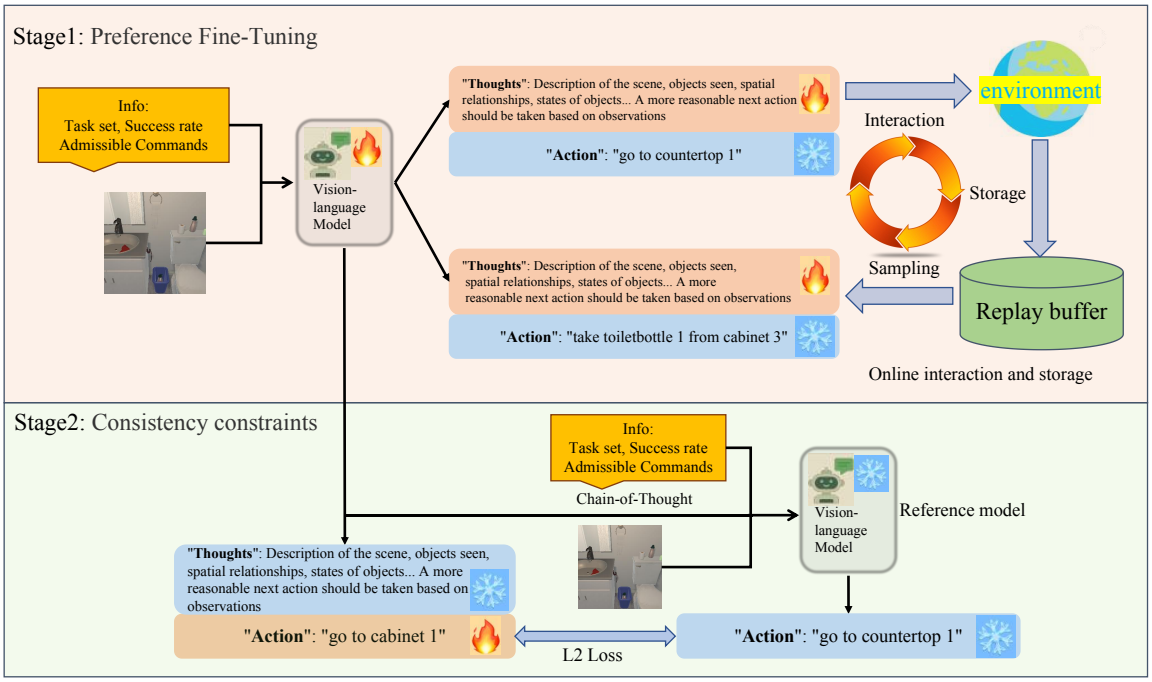


Figure 2: Overview of TCPO framework. The upper stage implements preference-driven CoT fine-tuning: The VLM processes environmental observations through CoT reasoning, generating spatial analyses and executable actions. Online interaction stores decision trajectories in a replay buffer, while contrastive learning with step-wise preference judgments optimizes thought-action distributions. The lower stage enforces APC through L2 loss regularization. This preserves pretrained thought-action mappings while constraining outputs to valid operations, as evidenced by comparative case studies. The 'flame' and 'snowflake' symbols indicate whether gradient backpropagation is applied to corresponding parameters during that training stage.

thresholds; (3) Integration of temporally consistent state-action histories $\tau_{1:t-1}$ to maintain trajectory coherence. By learning from these constructed preference pairs, the policy π_θ develops enhanced discriminative capabilities for identifying and optimizing high-value trajectories, even under sparse environmental feedback conditions.

3.2 Preference Fine-Tuning with CoT

Our visual language model (VLM) processes embodied tasks through structured prompts containing environmental observations and action trajectories. As shown in Figure 2, this framework enables contextual decision-making through chain-of-thought (CoT) reasoning followed by executable actions. The model's response sequence explicitly links cognitive processes with physical actions - each reasoning trajectory concludes with an `action` token denoting the selected operation, maintaining explicit action-rationale alignment.

Critical analysis reveals that conventional fine-tuning methods disproportionately prioritize action optimization while neglecting CoT coherence preservation. This imbalance disrupts the linguistic consistency established during pretraining, ultimately degrading reasoning capabilities. To address this dual challenge of action effectiveness and cognitive integrity, we implement two core constraints. First, we maintain distributional alignment between fine-tuned outputs and the reference model through KL-divergence control. Building on DPO principles, we establish:

$$Q(s,a) = \beta \log \frac{\pi_\theta(a|s)}{\pi_{ref}(a|s)} \tag{2}$$

Second, TCPO explicitly prioritizes reasoning quality over action selection during gradient updates. This approach simultaneously enhances decision reliability while preserving the model's inherent linguistic coherence - crucial for maintaining robust reasoning capabilities in dynamic environments. Then, we can fine-tune the output strategy of VLM by optimizing the Q value, while limiting the output distance between the fine-tuned model and the reference model without fine-tuning by adding a regularization term of KL divergence to the optimization objective, which is as follows:

$$\max_{\pi_\theta} \mathbb{E}_{s\sim D, a\sim\pi_\theta(a|s)}[Q(s,a)] - \beta D_{\mathrm{KL}}[\pi_\theta \parallel \pi_{ref}] \tag{3}$$

Based on this optimization objective, combined with some mathematical derivations of (Yang et al.,

2024), we can derive the following step-wise optimization formula:

$$\mathcal{L} = -\mathbb{E}_\zeta \log \sigma(\beta \log \frac{p(a_1^t|\mathcal{T}_1^t)\pi_\theta(\mathcal{T}_1^t|\tau_1^{t-1})}{\pi_{ref}(a_1^t, \mathcal{T}_1^t|\tau_1^{t-1})} - \beta \log \frac{p(a_2^t|\mathcal{T}_2^t)\pi_\theta(\mathcal{T}_2^t|\tau_2^{t-1})}{\pi_{ref}(a_2^t, \mathcal{T}_2^t|\tau_2^{t-1})}) \tag{4}$$

where we have Λ instead of $\sigma(\hat{Q}_\theta(a_1^t, \mathcal{T}_1^t, \tau_1^{t-1}) - \hat{Q}_\theta(a_2^t, \mathcal{T}_2^t, \tau_2^{t-1}))$. The complete mathematical derivation is provided in Appendix B. Our formulation introduces $\mathcal{T}_i^t$ as the CoT reasoning text at step t, with a denoting the resultant action. The gradient of the objective in Equation 4 is:

$$\nabla_\theta \mathcal{L} = -\beta \mathbb{E}_\zeta [\Lambda [\nabla_\theta \log \pi_\theta(\mathcal{T}_1^t|\tau_1^{t-1}) - \nabla_\theta \log \pi_\theta(\mathcal{T}_2^t|\tau_2^{t-1})]] \tag{5}$$

which reveals the elimination of direct action probability influences. This motivates our practical **TCPO** implementation with Action Probability Weighting (APW):

$$\tilde{\mathcal{L}} = -\mathbb{E}_\zeta \log \sigma[\beta p(a_1^t|\mathcal{T}_1^t) \log \frac{\pi_\theta(\mathcal{T}_1^t|\tau_1^{t-1})}{\pi_{ref}(a_1^t, \mathcal{T}_1^t|\tau_1^{t-1})} - \beta p(a_2^t|\mathcal{T}_2^t) \log \frac{\pi_\theta(\mathcal{T}_2^t|\tau_2^{t-1})}{\pi_{ref}(a_2^t, \mathcal{T}_2^t|\tau_2^{t-1})}] \tag{6}$$

The errors of both components and the feasibility of our approach will be analyzed later. Intuitively, the gradient term of the action probability serves to reinforce the probability of the corresponding thoughts. Actions with higher probabilities following CoT reasoning indicate stronger alignment with the underlying thought process, whereas lower probabilities suggest more randomness in action generation. This weighting mechanism helps suppress the generation of highly random positive samples and promotes the production of more deterministic, thought-aligned samples.

We provide a simple illustration of Equation 6 to demonstrate that the approximation is reasonable. Assuming that the pre-trained model has achieved good alignment, so $p(a|\mathcal{T})$ will be close to 1. We have the following:

$$\begin{aligned}\Delta(p(a_i^t|\mathcal{T}_i^t)) &= \log \frac{p(a_i^t|\mathcal{T}_i^t)\pi_\theta(\mathcal{T}_i^t|\tau_i^{t-1})}{\pi_{ref}(a_i^t, \mathcal{T}_i^t|\tau_i^{t-1})} \\ &- p(a_i^t|\mathcal{T}_i^t) \log \frac{\pi_\theta(\mathcal{T}_i^t|\tau_i^{t-1})}{\pi_{ref}(a_i^t, \mathcal{T}_i^t|\tau_i^{t-1})} \\ &= \log p(a_i^t|\mathcal{T}_i^t) + (1 - p(a_i^t|\mathcal{T}_i^t)) \log \frac{\pi_\theta(\mathcal{T}_i^t|\tau_i^{t-1})}{\pi_{ref}(a_i^t, \mathcal{T}_i^t|\tau_i^{t-1})}\end{aligned} \tag{7}$$

This variable will approach zero as $p(a|\mathcal{T})$ approaches 1. In practice, we have calculated the approximate distribution of action probabilities and demonstrated that our assumption is well-founded, which is illustrated in Figure 5c.

3.3 Action Policy Consistency

In the second optimization stage, we introduce a regularization term to constrain the final action text output. Direct fine-tuning of the reasoning chain may inadvertently modify the model's inherent language generation patterns, potentially inducing catastrophic forgetting. By aligning the action text outputs with those of a reference foundation model, we ensure strict adherence to the prompt's structural requirements while maintaining action validity. This regularization mechanism preserves output integrity without compromising task performance, with empirical effects demonstrated in Figure 1. To strengthen the consistency between the Chain-of-Thought reasoning process and final action generation, we propose augmenting the optimization framework with an additional constraint. Our key insight stems from the observation that pre-trained language models already exhibit well-optimized mappings from reasoning traces to actions. During interactive learning phases, we therefore enforce alignment with these pre-trained behaviors through an **Action Policy Consistency (APC)** constraint, implemented via L2 regularization term:

$$\mathcal{L}_{\text{TCPO}} = \tilde{\mathcal{L}} + \kappa \cdot \mathrm{L}_2\left(\pi_\theta(a_1^t|\mathcal{T}_1^t), \pi_{\text{ref}}(a_1^t|\mathcal{T}_1^t)\right) \tag{8}$$

where κ serves as a tunable hyperparameter controlling constraint intensity, and $\mathrm{L}_2(\cdot) \equiv \|\cdot\|_2$. Unlike the KL divergence typically used in DPO formulations, this constraint specifically targets the thought-to-action mapping process rather than overall output distribution matching.

To evaluate the effectiveness of the APC constraint, we perform comparative experiments examining model outputs at 2k training steps (Figure 1). The constrained model generates coherent CoT reasoning that logically leads to valid actions. In contrast, the unconstrained model often exhibits a disconnect between reasoning and action—despite producing plausible intermediate reasoning, it frequently results in invalid final actions unrelated to the preceding analysis. This divergence underscores the importance of enforcing explicit reasoning-action alignment to preserve decision consistency. The full training procedure for our method is provided in Appendix D.

	GymCards					ALFworld						
	EZP	P24	BJ	NL	Avg.	Pick2	Look	Clean	Heat	Cool	Pick	Avg.
CNN+RL	0	0	38.8	87.1	31.5	0	0	0	0	0	0	0.0
GPT4-V (Yang et al., 2023)	10.5	0	25.5	65.5	25.4	14.6	12.1	18.8	6.7	17.8	38.2	19.4
Gemini (Team, 2024)	2.0	0	30.0	82.5	28.6	12.0	16.7	0	0	0	34.6	13.5
LLaVA-sft (Liu et al., 2024)	23.0	2.6	23.1	24.8	18.4	**28.6**	0	14.4	11.1	0	39.2	17.7
RL4VLM (Zhai et al., 2024)	35.0	7.0	39.3	**89.4**	42.7	20.6	15.1	10.0	17.0	5.6	36.9	20.0
TCPO (Ours)	**50.0**	**11.1**	**40.3**	70.0	**42.9**	27.3	**33.3**	**25.0**	**28.6**	5.9	**41.7**	**26.7**

Table 1: We present results demonstrating that fine-tuning the VLM using TCPO and PPO leads to varying task completion rates and average task completion rates in the GymCards tasks and ALFworld environment. Our findings show that, for most tasks, fine-tuning the model with preference-based methods outperforms reinforcement learning approaches in terms of task performance. Furthermore, we observe that the preference method achieves the same average task completion rate as PPO with fewer interaction steps, highlighting its higher sample efficiency and reduced model degradation during online interaction with the environment.

4 Experiments

To systematically evaluate our proposed framework, we design experiments addressing three core research questions:

- Does TCPO effectively enhance visual-language models' decision-making proficiency in embodied simulation environments?
- Can TCPO stabilize action distributions and prevent policy degradation in interactive learning?
- Does the action probability consistency constraint improve reasoning-action alignment in model outputs?

Experimental Setup Our empirical evaluation utilizes the `gym_cards` environment featuring four core tasks: *Number Line* (**NL**), *Easy Pick* (**EZP**), *Pick-24* (**P24**), and *Blackjack* (**BJ**) and `alfworld` benchmark environment (Shridhar et al., 2020), containing six distinct household task categories: *Pick & Place* (abbreviated as **Pick**), *Pick Two & Place* (**Pick2**), *Clean & Place* (**Clean**), *Cool & Place* (**Cool**), *Heat & Place* (**Heat**), and *Examine in Light* (**Look**). Each task requires agents to process egocentric visual observations and textual instructions for sequential navigation and manipulation. The implementation builds upon the LLaVA-v1.6-Mistral-7B architecture (Liu et al., 2023), extended with our TCPO framework. Visual observations are processed through a structured input pipeline that serializes multi-modal inputs into model-compatible prompts while preserving original instruction-following capabilities. Our analysis focuses on ALFWorld's enhanced complexity, where agents perform multi-step household operations requiring sequential manipulation and spatial reasoning. This environment is prioritized for its comprehensive benchmarking that better reflects real-world challenges compared to foundational GymCards tasks.

Your are an expert in the ALFRED Embodied Environment.
Your task is to ∗ *task name* ∗. You are also given the following text description of the current scene: ∗ *obs* ∗}.
Your admissible actions of the current situation are: [∗ *reformatted admissible actions* ∗].
Your response should be a valid Json file in the following format:
"thoughts": "{first describe what do you see in the image using the text description, then carefully think about which action to complete the task. },
"reflections": "{reflect on your historical trajectory and carefully think about which action to complete the task.}",
"action": "{an admissible action}"
your actions should be based solely on the analysis provided by your thoughts!
your output need to be in 60 words!

Figure 3: Prompt used in ALFWorld tasks.

Prompt Design Our chain-of-thought prompting strategy integrates three core components through natural language instructions. First, we formalize ALFWorld's semantic instructions into structured objectives, mapping paraphrased commands like "examine the pillow with the desklamp" and "look at the pillow under the desklamp" to standardized procedural sequences involving object localization, navigation, and interaction. Second, we explicitly define action space constraints based on environment dynamics, specifying preconditions (e.g., proximity requirements for object interaction) and postconditions (e.g., possession prerequisites for placement actions), with action validity forming a key evaluation metric. Finally, we enforce strict JSON output formatting requiring logically connected `thoughts` and `action` fields, ensuring causal relationships between reasoning traces and final decisions, while rigorously observing output text length constraints. The complete prompt structure with exemplar inputs is visualized in Figure 3.

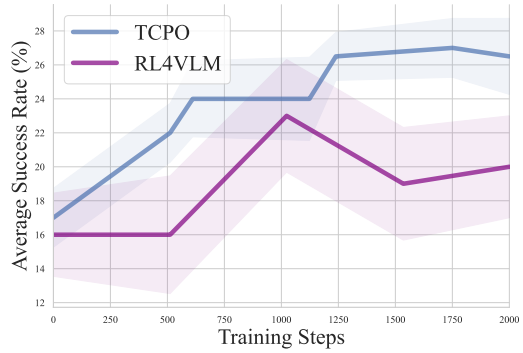


Figure 4: Training curves in the ALFWorld environment. In the first 2k steps, the TCPO method demonstrates superior convergence and efficiency compared to the PPO-based interaction method.

Implementation Our training integrates supervised fine-tuning (SFT) on the LEVI-Project/sft-data corpus (Zhai et al., 2024) containing 45k GPT-4-generated expert trajectories, ensuring structured JSON outputs for action-thread consistency. Subsequent environmental interaction employs online policy optimization with real-time monitoring of action validity and trajectory coherence, dynamically adjusting learning parameters to preserve structured response patterns while enhancing decision-making in interactive scenarios.

4.1 How much better we are at making decisions

The aim of experiments in this section is to validate the performance of TCPO. We first conduct preliminary validation on GymCards' four core tasks (NL, EZP, P24, BJ), where TCPO achieves 42.9% average success rate compared to 42.7% of PPO, demonstrating superior instruction comprehension in constrained scenarios. To evaluate whether the algorithm can consistently generate decisions through the CoT process, we use the success rate of task execution as a reference and select PPO from the RL4VLM (Zhai et al., 2024) framework as the baseline. Our baseline results for CNN+RL, GPT4-V, Gemini, and LlaVA-sft are directly reused from RL4VLM due to the lack of reproduction details. In contrast, the RL4VLM baseline was rigorously reproduced using the original training methods and parameters. In GymCards experiments, we implement task-specific reward shaping where preference scores incorporate both game completion and strategic depth metrics, using Equation 9 with adjusted weights for card-game dynamics. ALFWorld does not provide a reward function during interactions, it only indicates whether the current task is successfully executed and returns the task's progress. Given that such progress updates are sparse in a larger action space, we construct preference criteria for preference learning. The preference score for each trajectory is calculated using Equation 9:

Table 2: Performance comparison (average success rates %) with preference-based learning approaches.

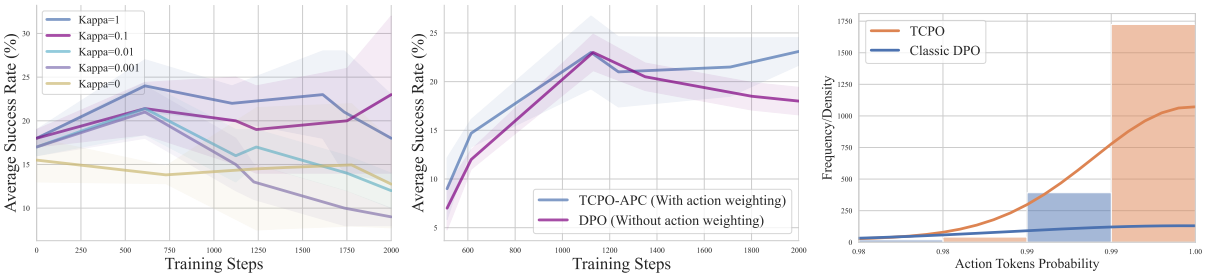
Methods	GymCards	ALFworld
PPO (Schulman et al., 2017)	32.8	20.0
DPO (Rafailov et al., 2024)	31.5	18.8
D3PO (Yang et al., 2024)	35.6	22.1
TCPO (Ours)	**42.9**	**26.7**

$$P = 50 * success\ rate - \mathbb{1}_{\{invalid\}} \tag{9}$$

$$\mathbb{1}_{\{invalid\}} = \begin{cases} 1 & action \notin admissible\ action \\ 0 & \text{otherwise} \end{cases}$$

where $\mathbb{1}_{\{invalid\}}$ represents the rejection of illegal actions given the same success rate. During the exploration phase, the agent collects trajectory data and constructs sample pairs based on the six task types mentioned above. Higher preference scores indicate greater sample preference. In practice, considering the achievement of long-term goals, we calculate preference scores using a method similar to discount factor weighting in reinforcement learning returns. Due to the high randomness of ALFWorld, we set up experimental environments with different seeds and calculated the mean and variance of each result.

We use Equation 8 for the model weight update with $\kappa = 0.1$, measure the agent's performance by the average success rate of each task. The final comparisons are shown in Table 1 and 2. It can be seen that TCPO consistently outperforms all RL-based and preference-based baselines. We further plot the change in the average success rate over training on ALFWorld in Figure 4. TCPO exhibits a more robust growth, and the continuous rise of the curve confirms the improvement in model degradation issues. ALFWorld gives task randomly so we calculate the overall success rate as the weighted average of success rates under all tasks. TCPO shows an improvement in the overall success rate, indicating that our algorithm can learn more efficiently from interactions. In our experiments, we used approximations such as $\log(\pi(a|\mathcal{T},\tau)\pi(\mathcal{T}|\tau)) \approx \pi(a|\mathcal{T},\tau)\log(\pi(\mathcal{T}|\tau))$ when $\pi(a|\mathcal{T},\tau) \to 1$. We calculated the occurrence probability distribution of action tokens in



(a) Impact of Kappa (κ) (b) Comparison of success rates (c) Action token probability

Figure 5: The experimental result of TCPO. (a) Impact of different values of κ in APC. (b) Comparison of the average success rates between TCPO-APC and classic DPO. (c) Action tokens probability distribution.

the experiments to demonstrate that our approximations are reasonable.

4.2 What role does action policy consistency constraint play?

We pointed out that during training, to enhance stability, we introduced the regularization of action token probabilities between finetune model and reference model. This section will explore the impact of regularization on the results and investigate its role. We designed ablation experiments, where we conducted trials with different regularization weight values κ under the same parameter settings, and recorded the average success rate of the agent during training. In this experiment, we use Equation 4 with the regular term as the loss function, with other conditions the same as in Section 4.1.

The results in Figure 5a show that different values of κ significantly impact the success rate. As the parameter increases, the action policy consistency constraint strengthens, leading to improved model performance. This validates the importance of regularization. However, when κ is set to 1, the algorithm's performance declines, indicating that κ should neither be too large nor too small, with a value around 0.1 yielding near-optimal performance. Given the importance of the κ parameter, its optimal value may vary across different environments or tasks. The optimal κ value of 0.1 was determined through comprehensive testing across a three-order-of-magnitude parameter range. The experiment demonstrated exceptional robustness, maintaining consistent performance across diverse conditions without requiring scenario-specific parameter adjustments. Due to space constraints, we do not explore this further in this paper.

Table 3: Effect of joint optimization of TCPO.

	GymCards	ALFworld
APW-only	35.8	23.0
APC-only	34.5	23.1
APC-APW-sequential	37.6	24.5
TCPO	**42.9**	**26.7**

4.3 APW in TCPO

Our analysis in Section 3.2 establishes that action probability weighting (APW) intrinsically reinforces decision determinism through cognitive-behavioral alignment. prioritizing gradient updates for high-probability actions. We validate this through comparative ablation studies between our APW-enhanced TCPO-APC framework (Equation 6) and baseline DPO (Equation 4), using identical experimental configurations. The comparative analysis of experimental results in Figure 5b delineates the performance comparison between the two experimental configurations, while Figure 5c quantitatively characterizes the action probability evolution during the initial 2000 training iterations. As demonstrated by the APW-conditioned results, the action token distribution exhibits predominant clustering near unity (probability ≈ 1), reflecting enhanced decision determinism and policy robustness. Conversely, the non-weighted configuration reveals a broadly distributed action probability spectrum, with notable instances of sub-0.9 probability values. Such dispersion in action selection probabilities suggests reduced policy convergence stability.

4.4 Effect of Joint Optimization

The overall objective of TCPO is a combination of APW and APC, which is jointly optimized to simultaneously enforce consistency between the Thoughts and Actions. To further validate the effectiveness of joint optimization, we present the experimental results in the Table 3, comparing methods

of APW only, APC only and sequential optimization of APW and APC. The results indeed demonstrate the effectiveness of the joint optimization of the Thoughts and Actions.

5 Conclusions

We introduce TCPO, an algorithmic framework for online interactive preference fine-tuning of multimodal models during chain-of-thought reasoning. Built upon LLaVA-7B, TCPO achieves enhanced embodied task execution through dynamic replanning and rigorous CoT-action alignment via APW and APC. Experimental results demonstrate the superiority of TCPO over conventional reinforcement learning baselines in ALFWorld environments, with ablation studies confirming the critical role of APW in gradient prioritization and the contribution of APC to policy robustness.

Limitations

Despite the effectiveness of our TCPO approach, there are two future directions that we'd like to point out. First, the Markovian assumption, as adopted in previous works, restricts the ability to handle complex non-Markovian decision processes in real-world scenarios. However, within our dynamically aligned environment algorithm, this assumption remains viable, as the algorithm inherently learns environmental dynamics. In this framework, redundant historical information may interfere with model judgment. Nevertheless, non-Markovian modeling is an important direction for future research. Moving forward, we plan to develop temporal modeling mechanisms to integrate historical information into TCPO, focusing on long-horizon task dependencies to eliminate the Markovian assumption. Second, empirical validation remains constrained to specific household tasks, necessitating broader domain evaluation to extend our approach to various domains. In the future, we plan to expand our experiments to a wider range of embodied environments such as VirtualHome.

References

Josh Abramson, Arun Ahuja, Iain Barr, Arthur Brussee, Federico Carnevale, Mary Cassin, Rachita Chhaparia, Stephen Clark, Bogdan Damoc, Andrew Dudzik, et al. 2020. Imitating interactive intelligence. *arXiv preprint arXiv:2012.05672*.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. 2023. Grounding large language models in interactive environments with online reinforcement learning. In *International Conference on Machine Learning*, pages 3676–3713. PMLR.

Lichang Chen, Jiuhai Chen, Chenxi Liu, John Kirchenbauer, Davit Soselia, Chen Zhu, Tom Goldstein, Tianyi Zhou, and Heng Huang. 2024. Optune: Efficient online preference tuning. *Preprint*, arXiv:2406.07657.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. 2018. Babyai: A platform to study the sample efficiency of grounded language learning. *arXiv preprint arXiv:1810.08272*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Ishita Dasgupta, Christine Kaeser-Chen, Kenneth Marino, Arun Ahuja, Sheila Babayan, Felix Hill, and Rob Fergus. 2023. Collaborating with language models for embodied reasoning. *arXiv preprint arXiv:2302.00763*.

Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. 2023. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. Vln bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 1643–1653.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

Hengyuan Hu and Dorsa Sadigh. 2023. Language instructed reinforcement learning for human-ai coordination. In *International Conference on Machine Learning*, pages 13584–13598. PMLR.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022a. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International conference on machine learning*, pages 9118–9147. PMLR.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. 2022b. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*.

Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. 2022. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2(3):6.

Siddharth Karamcheti, Megha Srivastava, Percy Liang, and Dorsa Sadigh. 2022. Lila: Language-informed latent actions. In *Conference on Robot Learning*, pages 1379–1390. PMLR.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. 2024a. Language models can solve computer tasks. *Advances in Neural Information Processing Systems*, 36.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. 2024b. Openvla: An open-source vision-language-action model. *Preprint*, arXiv:2406.09246.

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *Preprint*, arXiv:2406.18629.

Shuang Li, Xavier Puig, Chris Paxton, Yilun Du, Clinton Wang, Linxi Fan, Tao Chen, De-An Huang, Ekin Akyürek, Anima Anandkumar, et al. 2022. Pre-trained language models for interactive decision-making. *Advances in Neural Information Processing Systems*, 35:31199–31212.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. 2023. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2024. Improved baselines with visual instruction tuning. *Preprint*, arXiv:2310.03744.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Preprint*, arXiv:2304.08485.

Pan Lu, Baolin Peng, Hao Cheng, Michel Galley, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, and Jianfeng Gao. 2024. Chameleon: Plug-and-play compositional reasoning with large language models. *Advances in Neural Information Processing Systems*, 36.

Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. 2020. Improving vision-and-language navigation with image-text pairs from the web. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 259–274. Springer.

Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhai Wang, Mingyu Ding, Jun Jin, Bin Wang, Jifeng Dai, Yu Qiao, and Ping Luo. 2023. Embodiedgpt: Vision-language pre-training via embodied chain of thought. *Preprint*, arXiv:2305.15021.

Arka Pal, Deep Karkhanis, Samuel Dooley, Manley Roberts, Siddartha Naidu, and Colin White. 2024. Smaug: Fixing failure modes of preference optimisation with dpo-positive. *Preprint*, arXiv:2402.13228.

Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. 2022. The unsurprising effectiveness of pre-trained vision models for control. In *international conference on machine learning*, pages 17359–17371. PMLR.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

Allen Z Ren, Bharat Govil, Tsung-Yen Yang, Karthik R Narasimhan, and Anirudha Majumdar. 2023. Leveraging language for accelerated learning of tool manipulation. In *Conference on Robot Learning*, pages 1531–1541. PMLR.

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

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2024. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Preprint*, arXiv:2303.11366.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. 2022. Cliport: What and where pathways for robotic manipulation. In *Conference on robot learning*, pages 894–906. PMLR.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. *Preprint*, arXiv:1912.01734.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. 2023. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M. Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. *Preprint*, arXiv:2212.04088.

Richard S Sutton. 2018. Reinforcement learning: An introduction. *A Bradford Book*.

Andrew Szot, Max Schwarzer, Harsh Agrawal, Bogdan Mazoure, Walter Talbott, Katherine Metcalf, Natalie Mackraz, Devon Hjelm, and Alexander Toshev. 2024. Large language models as generalizable policies for embodied tasks. *Preprint*, arXiv:2310.17722.

Weihao Tan, Wentao Zhang, Shanqi Liu, Longtao Zheng, Xinrun Wang, and Bo An. 2024. True knowledge comes from practice: Aligning llms with embodied environments via reinforcement learning. *Preprint*, arXiv:2401.14151.

Gemini Team. 2024. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023a. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. 2023b. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*.

Zhiyuan Xu, Kun Wu, Junjie Wen, Jinming Li, Ning Liu, Zhengping Che, and Jian Tang. 2024. A survey on robotics with foundation models: toward embodied ai. *Preprint*, arXiv:2402.02385.

Kai Yang, Jian Tao, Jiafei Lyu, Chunjiang Ge, Jiaxin Chen, Weihan Shen, Xiaolong Zhu, and Xiu Li. 2024. Using human feedback to fine-tune diffusion models without any reward model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2024, Seattle, WA, USA, June 16-22, 2024*, pages 8941–8951. IEEE.

Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. 2023. The dawn of lmms: Preliminary explorations with gpt-4v(ision). *Preprint*, arXiv:2309.17421.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. *Preprint*, arXiv:2210.03629.

Andy Zeng, Maria Attarian, Brian Ichter, Krzysztof Choromanski, Adrian Wong, Stefan Welker, Federico Tombari, Aveek Purohit, Michael Ryoo, Vikas Sindhwani, et al. 2022. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*.

Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, et al. 2024. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. *arXiv preprint arXiv:2405.10292*.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. 2023. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *The Twelfth International Conference on Learning Representations*.

A Training Details

We provide more detailed hyper-parameters during in Table 4. During SFT Phase, we use a dataset of 45k samples, with batch size 4 and 1 training epoch. The gradient accumulation steps is set to 1, with learning rate 2e-5. For the Online Learning Phase, the total sampling steps are 2k to 3k. Parameters are update 1 epoch training per online update cycle.

Our baseline implementation strictly adheres to the reproduction methodology and parameters from the open-source RL4VLM project, as documented in their GitHub repository: https://github.com/RL4VLM/RL4VLM. The task completion rate is obtained through the data statistics of the interface feedback of the task completion rate in the environment. The environment determines task completion through its internal graph structure and returns a binary signal (0 for failure, 1 for success).

Table 4: Hyper-parameters for TCPO.

Hyperparameter	Value
Seed	5 random seeds
Learning Rate	3e-4
Mini Batch Size	1
Grad Accum Steps	256
Max New Tokens	1024
Temperature	0.2
Discount Factor (γ)	0.99
Preference Weight (κ)	0.1
Start Training Samples Nums	1,000

B Derivation of Formulas

We provide a simple derivation of Equation 4. During the RL phase with reward model, the object of training is to maximize returns. Following prior works the optimization is formulated as:

$$\max_{\pi_\theta} \mathbb{E}_{s\sim D, a\sim\pi_\theta(a|s)}[Q(s,a)] - \beta \mathrm{D}_{\mathrm{KL}}[\pi_\theta \parallel \pi_{ref}] \tag{10}$$

which can be rewritten as:

$$\begin{aligned}
&\max_{\pi_\theta} \mathbb{E}_{s\sim D, a\sim\pi_\theta(a|s)}[Q(s,a)] - \beta \mathrm{D}_{\mathrm{KL}}[\pi_\theta \parallel \pi_{ref}] \\
&= \max_{\pi_\theta} \mathbb{E}[Q(s,a) - \beta \log \frac{\pi(a|s)}{\pi_{ref}(a|s)}] \\
&= \min_{\pi_\theta} \mathbb{E}[\log \frac{\pi(a|s)}{\pi_{ref}(a|s)} - \frac{1}{\beta} Q(s,a)] \\
&= \min_{\pi_\theta} \mathbb{E}[\log \frac{\pi(a|s)}{\pi_{ref}(a|s)\exp\left(\frac{1}{\beta}Q(s,a)\right)}] \\
&= \min_{\pi_\theta} \mathbb{E}_{s\sim D}[\mathrm{D}_{\mathrm{KL}}[\pi(a|s) \parallel \tilde{\pi}(a|s)]]
\end{aligned}$$

where $\tilde{\pi}(a|s) = \pi_{ref}(a|s)\exp\left(\frac{1}{\beta}Q(s,a)\right)$. KL-divergence is minimized at zero if and only if the two distributions are identical. Therefore, in the case of the optimal solution we get:

$$\pi(a|s) = \tilde{\pi}(a|s) = \pi_{ref}(a|s)\exp\left(\frac{1}{\beta}Q(s,a)\right)$$

A simple transformation yields:

$$Q(s,a) = \beta \log \frac{\pi(a|s)}{\pi_{ref}(a|s)} \tag{11}$$

We can know from Yang et al. (2024) that the Q-value form of Bradley-Terry preference distribution can be expressed as:

$$\begin{aligned}
&p(\tau_1 > \tau_2 | a_i^t, s_i^t, a_i^{t-1} ..., s_i^0)_{i\in\{1,2\}} \\
&= \frac{\exp(Q(s_1^t, a_1^t))}{\sum_{i\in\{1,2\}} \exp\left(Q(s_i^t, a_i^t)\right)}
\end{aligned} \tag{12}$$

Combining Eq. 11 and Eq. 12, replacing s_i^t with τ_i^{t-1} and a_i^t with $(a_i^t, \mathcal{T}_i^t)$, we derive the following loss function:

$$\begin{aligned}
\mathcal{L} = -\mathbb{E}_\zeta \log \sigma[\beta \log \frac{\pi_\theta(a_1^t, \mathcal{T}_1^t | \tau_1^{t-1})}{\pi_{ref}(a_1^t, \mathcal{T}_1^t | \tau_1^{t-1})} \\
-\beta \log \frac{\pi_\theta(a_2^t, \mathcal{T}_2^t | \tau_2^{t-1})}{\pi_{ref}(a_2^t, \mathcal{T}_2^t | \tau_2^{t-1})}]
\end{aligned} \tag{13}$$

which is similar to Eq. 4

Table 5: Impact of κ on ALFWorld.

task	κ = 0.001	κ =0.01	κ=0.1	κ=1
Pick	25.7%	27.5%	41.7%	35.0%
Pick2	16.7%	25.0%	27.3%	18.5%
Clean	11.0%	22.2%	25.0%	10.0%
Look	25.0%	26.0%	33.3%	26.2%
Heat	6.8%	11.0%	28.6%	13.0%
Cool	1.0%	5.8%	5.9%	5.1%
Avg.	13.3%	20.8%	26.7%	18.8%

C Parameter study of κ

To further understand the impact of the parameter κ, we've conducted parameter experiments to observe the effects of varying κ values under different tasks. The experimental results are shown in the Table 5. It can be seen that $kappa$=0.1 achieves the best performance across all tasks in ALFWorld.

D Pseudo Code of TCPO

We present the pseudo code of TCPO below for better understanding of our approach.

Algorithm 1 Training pipeline of embodied VLMs thorogh TCPO

Require: Reference policy network π_{ref}, finetune policy network π_θ, current observation o_t, past trajectories wises buffer τ_i^{t-1}, $\{i = 1 \dots N\}$

for $i = 1, 2, ..., N$ **do**

Randomly sample past trajectory τ_i^{t-1}. The current chain of reasoning thoughts $\pi_\theta(\mathcal{T}_i^t|\tau_i^{t-1})$ and probabilities of output actions $p(a_i^t|\mathcal{T}_i^t)$ are generated through previous trajectories and fine-tuned models. $\pi_{ref}(a_i^t, \mathcal{T}_i^t|\tau_i^{t-1})$ is also obtained through the reference model.

for j in past trajectories buffer which can be paired with τ_i^{t-1}. **do**

The current chain of reasoning thoughts $\pi_\theta(\mathcal{T}_j^t|\tau_j^{t-1})$ and probabilities of output actions $p(a_j^t|\mathcal{T}_j^t)$ are generated through previous trajectories and fine-tuned models. $\pi_{ref}(a_j^t, \mathcal{T}_j^t|\tau_j^{t-1})$ is also obtained through the reference model.

Calculating the loss in Equtation 6.

end for

Compute the Regularization loss $\mathrm{L}_2\left(\pi_\theta(a_1^t|\mathcal{T}_1^t), \pi_{\text{ref}}(a_1^t|\mathcal{T}_1^t)\right)$ and obtain the $\mathcal{L}_{\text{TCPO}}$.

end for

The parameters θ are updated by backpropagation through the loss function.

Table 6: Sample Efficiency Results.

method	success rate: 18%	success rate: 20%
PPO	650 (avg steps)	810
DPO	580	670
TCPO	400	620

E Effect of Sample Efficiency

To better demonstrate the sample efficiency described in Table 1, we have conducted additional experiments to validate the sample. The training steps required by different methods to achieve varying average success rates during training are shown in Table 6. As shown in the table, our algorithm achieves the same success rate while requiring fewer iteration steps through preference learning.

F Design Details and Discussions

Motivation of using L_2 term in APC We choose L_2 loss for three primary reasons. First, computational efficiency - L_2 term operates with O(n) computational complexity and offers simple implementation. Second, optimization stability - the linear gradients of L_2 ensure higher stability in online algorithms and mini-batch optimization. Third, numerical robustness - L_2 inherently avoids the need for log(0) protection mechanisms. In our experiments, we also tested KL divergence but observed inferior convergence performance compared to L_2 term.

Sample pair construction scheme Our preference sample pairs are generated through complete trajectory sampling. During experiments, we impose a trajectory length constraint by setting a maximum sampling step of 50. The preference score comprises two components: i) Final task success rate of the trajectory (0 or 1 in hard mode), and ii) Proportion of legal actions (trajectories with more legal actions are preferred under equivalent conditions). We estimate step-wise preference scores using a γ weighting factor for credit assignment along the trajectory.

Success rate settings and discount factor weighting The success rate signal is binary. The discount factor here serves for credit assignment. Since our preference construction is step-wise, it requires allocating contributions to each step within the same trajectory.

Train/test split Our method employs online interaction for iterative updates, thus eliminating the need for train/test dataset split. During the online sampling process, we simultaneously calculate the agent's average success rate across all tasks and plot the corresponding success rate curve as shown in Figure 4. Specifically, our evaluation approach involves computing the average success rate once during subsequent sampling after each model update to assess current model capability. All tasks (the four Gymcards tasks individually, and all ALFWorld tasks collectively) include experimental data from at least 5 different seeds, with both mean values and variance displayed in the curves.

Table 7: An example of the prompt and image in our tasks.

Inputs:

You are an expert in the ALFRED Embodied Environment. You are also given the following text description of the current scene: ‘You arrive at loc 0. The cabinet 1 is open. On the cabinet 1, you see a pan 1, a kettle 1, a winebottle 1, a apple 1, a stoveknob 1, a stoveknob 2, a stoveknob 3, a stoveknob 4, a knife 1, a saltshaker 1, and a bread 1.‘. Your task is to put a cool mug in cabinet. Your admissible actions of the current situation are: [‘go to countertop 1’, ‘go to cabinet 2’, ‘go to countertop 2’, ‘go to stoveburner 1’, ‘go to drawer 1’, ‘go to drawer 2’, ‘go to drawer 3’, ‘go to stoveburner 2’, ‘go to stoveburner 3’, ‘go to stoveburner 4’, ‘go to drawer 4’, ‘go to cabinet 3’, ‘go to cabinet 4’, ‘go to microwave 1’, ‘go to cabinet 5’, ‘go to cabinet 6’, ‘go to cabinet 7’, ‘go to sink 1’, ‘go to sinkbasin 1’, ‘go to fridge 1’, ‘go to toaster 1’, ‘go to coffeemachine 1’, ‘go to cabinet 8’, ‘go to drawer 5’, ‘go to drawer 6’, ‘go to drawer 7’, ‘go to drawer 8’, ‘go to shelf 1’, ‘go to shelf 2’, ‘go to countertop 3’, ‘go to shelf 3’, ‘go to drawer 9’, ‘go to garbagecan 1’, ‘open cabinet 1’, ‘close cabinet 1’, ‘take pan 1 from cabinet 1’, ‘take kettle 1 from cabinet 1’, ‘take winebottle 1 from cabinet 1’, ‘take apple 1 from cabinet 1’, ‘take stoveknob 1 from cabinet 1’, ‘take stoveknob 2 from cabinet 1’, ‘take stoveknob 3 from cabinet 1’, ‘take stoveknob 4 from cabinet 1’, ‘take knife 1 from cabinet 1’, ‘take saltshaker 1 from cabinet 1’, ‘take bread 1 from cabinet 1’, ‘inventory’, ‘look’, ‘examine cabinet 1’]. Your response should be a valid json file in the following format:

{
"thoughts": "first describe what do you see in the image using the text description, then carefully think about which action to complete the task.",
"action": "an admissible action"
}

Responses:

{
"thoughts": "Based on the text description, I am at loc 0 with an open cabinet 1 in front of me. Items such as a pan, kettle, wine bottle, apple, several stoveknobs, a knife, saltshaker, and bread are visible on the cabinet 1. To find a cool mug to put in the cabinet, I may need to first locate a mug, which might be in one of the cabinets or drawers. Since the task does not involve any items currently in cabinet 1, and the mug is not mentioned in the description, my best course of action would be to start by searching the other cabinets or drawers for the mug.",
"action": "go to cabinet 2"
}

Weighted average of success rates under all tasks The ALFWorld environment contains 6 major task categories with over 5,000+ predefined task instructions, all of which are randomly assigned. Therefore, during sampling, we calculate the overall weighted average success rate by using the occurrence frequency of different task categories as weights. This approach helps reduce estimation bias - for instance, if tasks in the “Pick" category were only executed once and succeed, their 100% category success rate would significantly impact the arithmetic mean and increase the variance of estimated values.

Validity of the approximation in Equation 6 In Figure 5(c), the X-axis and Y-axis represent Action Token Probability and Sample Density respectively, illustrating the probability distribution of action tokens across all sampled trajectories. Comparing TCPO and DPO, the TCPO method shows a probability distribution of final decision action tokens concentrated around 1, demonstrating that (a) the approximation condition in Section 3.2 is easily satisfied, and (b) TCPO naturally guides the model toward more deterministic action generation during training, supporting the reasonableness of the approximation.

Design of the prompt The detailed description of the prompt used in our experiments is shown in Figure 3. Our approach intentionally requires the agent to verbalize its visual perceptions, fostering deeper situational awareness and more deliberate planning—significantly enhancing contextual comprehension. While direct planning without perceptual descriptions is technically feasible, this design choice strengthens reasoning. Additionally, we explicitly confirm that our framework does not incorporate any environmental descriptions beyond the agent’s own perceptual outputs. A example is shown in Table 7.