GUI-Libra: Data-Efficient Post-Training for Reliable Reasoning-and-Acting in Native GUI Agents

Rui Yang[1†], **Qianhui Wu**[2,*], **Zhaoyang Wang**[3†], **Hanyang Chen**[1], **Ke Yang**[1†], **Hao Cheng**[2]
Huaxiu Yao[3], **Baolin Peng**[2], **Huan Zhang**[1], **Jianfeng Gao**[2], **Tong Zhang**[1]
[1]University of Illinois Urbana-Champaign, [2]Microsoft Research, [3]UNC-Chapel Hill
https://gui-libra.github.io

Abstract

Open-source native GUI agents still lag behind closed-source systems on long-horizon tasks. One reason is the direct reuse of generic post-training pipelines that ignore GUI-specific failure modes: standard supervised fine-tuning (SFT) with long chain-of-thought (CoT) reasoning often degrades grounding, and stronger offline optimization in RL does not necessarily translate to better online performance. This offline-to-online mismatch arises in part from *partial verifiability*: multiple actions may validly advance a task, but supervision typically marks only one demonstrated action as correct, creating reward ambiguity. We present **GUI-Libra**, a data-efficient post-training recipe for reliable reasoning-and-acting in native GUI agents. GUI-Libra combines a construction and filtering pipeline for a curated 81K GUI reasoning dataset, *action-aware supervised fine-tuning* that mixes reasoning-then-action and direct-action supervision with action-aware token reweighting, and KL-constrained RL with success-adaptive scaling to improve offline-to-online predictability under ambiguous rewards. Across web and mobile benchmarks, GUI-Libra consistently improves both step-wise accuracy and end-to-end task completion. GUI-Libra-4B and GUI-Libra-8B improve their base models by +15.6% and +12.2% on AndroidWorld, +4.0% and +8.7% on Online-Mind2Web, and +12.5% and +11.3% on WebArena-Lite-v2. These results show that careful reasoning data curation and tailored post-training can substantially improve long-horizon task solving without costly online data collection. We release our dataset, code, and models to support future research.

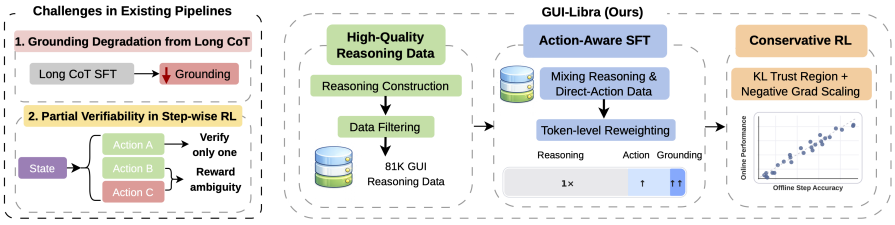


Figure 1: Overview of GUI-Libra. Using existing open-source GUI trajectories, we tackle key limitations of prior training pipelines through action-aligned reasoning data curation, action-aware SFT, and conservative RL, yielding consistent gains on online benchmarks.

1 Introduction

Large vision–language models (VLMs) have become a core building block for graphical user interface (GUI) agents [33, 45, 13, 38, 3], which interpret visual interfaces and output executable actions.

*Corresponding Author. † Part of the work was done during internship at Microsoft. Emails: ry21@illinois.edu, qianhuiwu@microsoft.com

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Table 1: Comparison of existing training recipes from different perspectives. The task type column indicates the training target of the released models: 🔍 denotes GUI grounding, 🧭 denotes GUI navigation (both online and offline), and 🧭 denotes offline step-wise action prediction.

Name	Task	Reasoning	SFT	RL	Open Data	Open Code	Data Size
OS-Atlas [43]	🔍🧭	✗	✓	✗	✓	✗	2.3M
AGUVIS [45]	🔍🧭	short	✓	✗	✓	✓	1.3M
UGround [13]	🔍	✗	✓	✗	✓	✓	1.3M
ScaleCUA [24]	🔍🧭	long	✓	✗	✓	✓	$>$ 170K
OpenCUA [40]	🔍🧭	long	✓	✗	✓	✓	$>$ 420K
UI-TARS [33]	🔍🧭	short	✓	✓	✗	✗	Unknown
GLM-4.1-V [16]	🔍🧭	long	✓	✓	✗	✗	Unknown
Ferret-UI Lite [50]	🔍🧭	long	✓	✓	✗	✗	29M
UI-R1 [27]	🔍🧭	short	✗	✓	✓	✓	136
GUI-R1 [28]	🔍🧭	short	✗	✓	✓	✓	3K
GUI-Libra (Ours)	🔍🧭	long	✓	✓	✓	✓	81K

Among them, native GUI agents [33] use a single end-to-end model to map user instructions and observations directly to actions, without external planners or separate grounding modules. Recent open-source native agents have made strong progress in visual grounding and low-level action execution [45, 38, 24, 40], but they still lag behind proprietary systems on long-horizon tasks that require both reasoning and precise actions.

Further progress depends on post-training GUI-centric VLMs, but current approaches face two bottlenecks: scarce action-aligned reasoning data and the direct reuse of generic post-training recipes that ignore GUI-specific failure modes. Existing GUI navigation datasets [20, 56, 45] often lack explicit rationales, provide only short or weakly grounded reasoning traces, or contain noisy action labels. Meanwhile, most open-source pipelines rely on either supervised fine-tuning (SFT) on brief rationales [43, 45] or reinforcement learning (RL) aimed primarily at grounding accuracy [28, 27, 59, 49]. In practice, these pipelines exhibit two recurring failures. First, incorporating chain-of-thought (CoT) often degrades grounding, creating a persistent tension between reasoning and action accuracy. Second, stronger offline optimization in RL does not necessarily yield better online task success. Motivated by the success of RL from verifiable rewards (RLVR) in domains such as mathematical reasoning [37, 51], recent work [16, 50] has explored step-wise RL for GUI agents, but these methods overlook a key property of GUI interaction: **partial verifiability, where multiple actions may correctly advance a task while offline supervision verifies only one demonstrated action**. As a result, alternative valid actions are ambiguously treated as failures, biasing gradients, destabilizing training, and weakening offline-to-online predictability.

To address these challenges, we propose GUI-Libra, a unified post-training framework for stronger decision making in native GUI agents. GUI-Libra is guided by three insights: (i) high-quality rationales and careful filtering are essential for data-efficient learning; (ii) during SFT, CoT tokens can dominate the loss and interfere with grounding, so learning must explicitly prioritize action and grounding tokens; and (iii) under partially verifiable rewards, GUI agents benefit from moderate KL-regularized RL to improve robustness and offline–online alignment.

Guided by these insights, GUI-Libra integrates action-aware supervised fine-tuning (ASFT) with conservative reinforcement learning. We develop a scalable construction and filtering pipeline and release a curated 81K GUI reasoning dataset with improved alignment between reasoning traces and executable actions. In ASFT, GUI-Libra mixes reasoning-then-action and direct-action supervision together with action-aware token reweighting to reduce the grounding degradation caused by long CoT traces. In RL, GUI-Libra optimizes policies with GRPO [37] under moderate KL regularization and introduces success-adaptive negative gradient scaling to reduce bias from ambiguously "negative" outcomes. The full pipeline uses only existing open-source data and avoids costly online interaction.

Extensive experiments across web and mobile benchmarks show that the GUI-Libra series (3B–8B) consistently improves offline step-wise accuracy and online task completion on AndroidWorld [34], WebArena-Lite-v2 [24], and Online-Mind2Web [46]. Notably, GUI-Libra-4B and GUI-Libra-8B improve their base models by +15.6% and +12.2% on AndroidWorld, and +4.0% and +8.7% on Online-Mind2Web. Ablations confirm the roles of action-aware supervision and conservative regularization, and we further analyze data filtering, explicit reasoning, and the reasoning-grounding trade-off during RL. We hope these findings and open-source resources encourage further work on data-efficient, reliable post-training for native GUI agents.

2 Related Work

GUI data and supervised post-training. GUI-agent training draws on grounding corpora [8, 13, 43], multi-step trajectories [35, 9, 56, 5], and reasoning-augmented demonstrations [55, 45, 54, 40]. Prior work such as AGUVIS, ScaleCUA, and OpenCUA uses curated visual interaction data to develop grounding and navigation capabilities [45, 24, 40]. These advances make the quality of supervision increasingly important: rationales must explain the intended action without weakening its grounding. GUI-Libra augments existing public trajectories with action-aligned reasoning and filters inconsistent samples. Alongside data curation, we study how long reasoning traces interfere with grounding and use mixed supervision and action-aware token weighting to preserve grounding accuracy.

Reinforcement learning for GUI agents. RL has been applied to GUI grounding and step-wise action prediction [27, 28, 59, 49], as well as interactive navigation [2, 38, 58, 1]. Step-wise RLVR offers a scalable training signal from existing demonstrations, while online interaction provides feedback along the agent's own trajectories at greater collection and infrastructure cost. Building on step-wise RL methods such as GUI-R1 [28], we investigate when offline action-matching gains transfer to closed-loop task completion. Our focus is partial verifiability: a non-matching action may still be valid. GUI-Libra combines ASFT initialization with KL-regularized GRPO and success-adaptive negative gradient scaling to improve online navigation using fixed offline data, without collecting new trajectories during training.

Offline learning and distribution shift. Imitation learning studies error accumulation when actions change future observations [36]; offline RL further emphasizes coverage and distribution shift between logged data and the learned policy [18]. Behavior regularization limits policy deviation [42], while conservative Q-learning addresses value overestimation through pessimistic estimation [17]. Our analysis builds on this distribution-shift perspective and separates occupancy mismatch from the unobserved probability of valid-but-uncredited actions under single-demonstration verification. This distinction connects conservative policy updates to the reliability of offline GUI evaluation, complementing the empirical study of KL regularization across navigation checkpoints. Additional related work on RLVR and VLM-based agents is provided in Appendix A.

3 Preliminaries

VLM-based GUI agents. We model GUI interaction as a goal-conditioned partially observable decision process with instruction space $\mathcal{L}$. Given an instruction $\ell \in \mathcal{L}$, the agent observes o_t (typically a screenshot) and conditions on the interaction history $h_t = (o_0, a_0, \ldots, o_{t-1}, a_{t-1})$ to produce a structured action a_t via a VLM policy $\pi_\theta(a_t \mid \ell, h_t, o_t)$. Episodes terminate upon success or at horizon T, and the objective is expected task success $J(\pi) \triangleq \Pr_{\tau\sim\pi}\big(\text{success}(\tau) = 1\big)$, where τ denotes the resulting interaction trajectory. We distinguish *low-level* tasks solved by a single atomic action from *high-level* tasks requiring multi-step interaction. Grounding is treated as a low-level decision problem that localizes a target element from (ℓ, o_t), while our focus is high-level navigation.

RL from Verifiable Rewards. Prior work also applies RL from verifiable rewards to optimize step-wise action/coordinate correctness [28, 27, 59, 49]. For each context $x = (\ell, h_t, o_t) \sim D$, we sample G actions from $\pi_{\theta_{\text{old}}}$, score them with a rule-based reward $r_k = \mathcal{R}(x, a_k)$ based on action type, text value, and coordinates, and update the policy with GRPO:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}_{x,\{a_k\}}\Big[\frac{1}{G}\sum_{k=1}^{G}\min\big(\rho_k\hat{A}_k,\ \text{clip}(\rho_k, 1-\epsilon, 1+\epsilon)\hat{A}_k\big) - \beta\,\text{KL}(\pi_\theta\|\pi_{\text{ref}})\Big],$$

where $\rho_k = \pi_\theta(a_k \mid x)/\pi_{\theta_{\text{old}}}(a_k \mid x)$ and $\hat{A}_k$ is the group-normalized advantage. Recent RLVR-style work often removes the explicit KL term (i.e., sets $\beta = 0$) to improve performance [51, 25, 59, 49]. However, for high-level GUI tasks, step-wise rewards can be ambiguous, making rule-based matching an imperfect proxy for online task completion.

4 Reasoning Data Curation for GUI Agents

As shown in Table 2, existing open-source GUI datasets typically provide only short rationales, often fewer than 100 thought tokens per step. Rather than collecting costly new online data, we augment

Table 2: Comparison of our GUI reasoning dataset with previous open-source datasets in web and mobile domains. Token statistics are computed using the Qwen2.5-VL-3B-Instruct tokenizer.

Dataset	Avg Thought Token Per Step	Total Steps	#Traj
MM-Mind2Web [56]	0	8K	1K
AndroidControl [20]	11	75K	14K
GUIAct [7]	0	17K	2.5k
AMEX [5]	0	35K	3K
GUI-Net-1M [54]	37	4M	1M
ScaleCUA [24]	0	170K	19K
AGUVIS [45] Stage2 L2 (All sources)	56	300K	35K
AGUVIS [45] Stage 2 L3 (All sources)	85	300K	35K
GUI-LIBRA-81K (Ours)	210	81K	9K

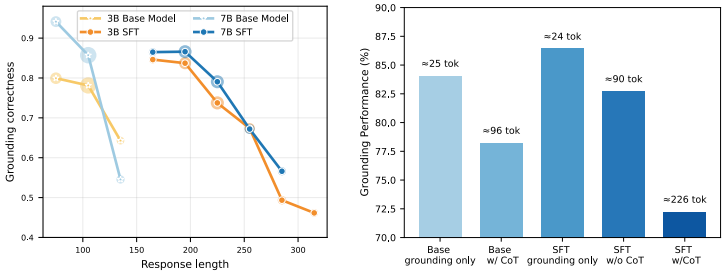


Figure 2: (a) Grounding accuracy on ScreenSpot-v2 versus response length for base models and CoT-SFT models. (b) Average grounding accuracy under different SFT strategies, where excessively long reasoning traces lead to a substantial drop.

existing web and mobile trajectories with richer CoT reasoning and filter out misaligned samples, yielding GUI-LIBRA-81K. Detailed processing is in Appendix B.

Data Format. Each sample contains the current GUI context as input and a reasoning trace plus a structured action as output, represented in a unified action format for both web and mobile control. Full format details are deferred to Appendix B.

Action-aligned Reasoning Curation. We identify two factors that limit the quality of reasoning generated from AGUVIS [45]: underspecified prompts and sensitivity to the generator model. We therefore add GUI-specific instructions for structured reasoning (observation, reflection, and planning), strengthen action-centric requirements, and use GPT-4.1 as the generator. Because the original annotations can be noisy, we treat the original action as a reference rather than a hard constraint: the generator may propose a different action when justified. It outputs the reasoning trace and action fields (`action_description`, `action_type`, `action_target`, `value`) while reusing the original `point_2d`, after which we filter out reasoning–action and action–coordinate inconsistencies. We then apply a two-stage filtering pipeline for action agreement and target–coordinate alignment, yielding **81K** SFT samples. For RL, we construct a more balanced **40K** subset by downsampling biased portions of the data. More details and statistics are deferred to Appendix B.

5 GUI-Libra

In this section, we introduce GUI-Libra, a native GUI agent for reliable reasoning and acting. Using GUI-LIBRA-81K, we first study how long chain-of-thought (CoT) SFT affects grounding, and then analyze how step-wise RLVR-style objectives relate to online navigation performance.

5.1 SFT with Long CoT Hurts GUI Grounding

Prior work [27, 28] suggests that removing CoT can improve grounding in GUI agents, but **systematic evidence remains limited, especially for *long* CoT traces because most existing datasets contain only short rationales.** Using GUI-LIBRA-81K, we study how response length correlates with grounding performance on ScreenSpot-v2 [43]; detailed evaluation settings are deferred to Appendix B. Figure 2(a) shows a clear negative correlation between response length and grounding accuracy for both base and CoT-SFT models. CoT-based SFT also widens the response-length distribution, producing many outputs longer than 250 tokens that suffer particularly severe drops. To isolate the source of this degradation, we compare three SFT variants on Qwen2.5-VL-3B-Instruct using GUI-LIBRA-81K: full reasoning-then-action outputs (*SFT with CoT*), action-only outputs

(*SFT without CoT*), and coordinate prediction from the action-target description (*Grounding-only*). Figure 2(b) shows that grounding-only SFT yields a modest gain, SFT without CoT slightly degrades performance, and long-CoT SFT causes a substantial drop, indicating that excessively long reasoning is the main source of grounding degradation.

5.2 Action-Aware Supervised Fine-Tuning

We seek to preserve reasoning while mitigating the grounding degradation caused by long CoT. To this end, we propose *action-aware supervised fine-tuning* (ASFT), which combines mixed supervision with token-level reweighting to balance reasoning, action prediction, and grounding.

Mixed reasoning and direct-action supervision. ASFT trains on a mixture of data with and without explicit reasoning traces. The direct-action variant removes the reasoning trace and keeps only the structured action between `<answer>` and `</answer>`, yielding two supervision modes: (i) reasoning-then-action and (ii) direct action prediction. In our setting, this dual-mode supervision strengthens action-centric learning while reducing reliance on verbose intermediate reasoning.

Action-aware reweighting. In addition to mixed supervision, ASFT assigns higher weights to action and grounding tokens. Although grounding is part of the action output, semantic components (e.g., action description, action type, and value) and spatial components (coordinates) can be weighted separately. Concretely, we treat tokens inside `<answer>...</answer>` as the action output and split them into *action tokens* (all tokens except `point_2d`) and *grounding tokens* (tokens associated with `point_2d`). Let c_t, a_t, and g_t denote the reasoning, action, and grounding tokens at step t, let $x_t = (\ell, h_t, o_t)$ denote the context, and let D_{mix} contain both reasoning-then-action and direct-action samples, with c_t empty for the latter. The ASFT objective is

$$\mathcal{L}_{\text{ASFT}}(\theta) = -\mathbb{E}_{(x_t,c_t,a_t,g_t)\sim D_{\text{mix}}} \frac{\log \pi_\theta(c_t \mid x_t) + \alpha_a \log \pi_\theta(a_t \mid x_t, c_t) + \alpha_g \log \pi_\theta(g_t \mid x_t, c_t, a_t)}{|c_t| + \alpha_a |a_t| + \alpha_g |g_t|}, \quad (1)$$

where α_a and α_g control the relative importance of action and grounding tokens. By adjusting them, ASFT recovers several common training strategies: (i) $\alpha_a = \alpha_g = 1$ reduces to standard SFT; (ii) $\alpha_a = \alpha_g \gg 1$ emphasizes action/grounding tokens and approximates CoT-free SFT; and (iii) $\alpha_g \gg \alpha_a$ with $\alpha_g \gg 1$ further reduces to grounding-only SFT. Overall, ASFT provides a flexible mechanism for balancing reasoning, action, and grounding during supervised fine-tuning of native GUI agents.

5.3 Reinforcement Learning from Partial Verifiable Rewards

Prior work has applied RLVR-style training to optimize step-wise action or coordinate correctness for GUI agents [28, 27, 59, 49]. However, multi-step GUI navigation differs from standard RLVR settings in two ways: (i) *errors accumulate and induce distribution shift*, since small step mistakes compound over time; and (ii) *rewards are partially verifiable*, since multiple actions may correctly advance a task while offline supervision verifies only one demonstrated action. Both factors are crucial for understanding when offline step-wise metrics can reliably predict online task success.

Setup. We model GUI navigation as an H-step decision process in which each state s may admit multiple valid actions $\mathcal{A}^*(s)$, but offline supervision verifies only one demonstrated action $\tilde{a}(s) \in \mathcal{A}^*(s)$. The induced step-wise reward $\tilde{r}(s,a) = \mathbf{1}\{a = \tilde{a}(s)\}$ is therefore *partially verifiable*: positive rewards are reliable, but zero reward is ambiguous because valid off-demo actions are not credited. Let μ denote the offline behavior policy and d_μ its population state distribution; μ need not be optimal and is not queried during training. Let $M_{\text{off}}(\pi) = \mathbb{E}_{s\sim d_\mu}[\pi(\tilde{a}(s) \mid s)]$ denote offline matching and $J(\pi)$ online task success. Their relationship depends on the occupancy mismatch $C(\pi)$ and the off-demo validity mass $\bar{\eta}_\pi$, which measures the probability of valid but uncredited actions averaged over d_μ. Formal definitions and the data assumptions are given in Appendix F.

Theorem 5.1: Offline-to-online bound under partial verifiability

Assume that any failed episode contains at least one invalid action, and that, for all $t \in [H]$, $d_{\pi,t}(s) > 0$ implies $d_\mu(s) > 0$ (i.e., $\text{supp}(d_{\pi,t}) \subseteq \text{supp}(d_\mu)$). This condition ensures the occupancy ratio $C(\pi)$ is well-defined. Then the online success probability satisfies

$$J(\pi) \;\geq\; 1 \;-\; H \cdot C(\pi) \cdot \Big(1 - M_{\text{off}}(\pi) - \bar{\eta}_\pi\Big). \quad (2)$$

Under a common coverage bound and fixed $\bar{\eta}_\pi$, the success lower bound increases with $M_{\text{off}}(\pi)$.

Takeaway. Theorem 5.1 separates two obstacles to offline-to-online alignment: state-distribution shift and unobserved valid alternatives. Coverage is an explicit assumption on the population distributions, not a property guaranteed by a finite corpus. A success lower bound does not determine checkpoint rankings or correlation; we evaluate this relationship empirically. Appendix F gives the proof, a version allowing uncovered states, and the finite-data interpretation. **Standard RLVR is often easier to predict because it is typically single-step ($H = 1$) and fully verifiable ($\eta_\pi \equiv 0$)**, so one-step matching directly reflects correctness and there is no error accumulation. In contrast, multi-step GUI agents face both distribution shift and partial verifiability, weakening offline-to-online alignment.

5.3.1 KL Regularization Improves Predictability

While many RLVR pipelines omit KL regularization for efficiency [51, 25, 59, 49], we find it important for multi-step GUI navigation. A KL trust region limits policy-induced changes in state visitation and in the probability assigned to valid alternatives, providing a theoretical rationale for the improved offline–online alignment observed in our experiments.

KL-induced bounds for occupancy mismatch and off-demo validity mass (informal). Let π_{ref} be a reference policy (e.g., the SFT initialization trained on demonstrations) and assume a per-state KL constraint $\mathrm{KL}(\pi(\cdot \mid s) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid s)) \le \varepsilon$ for all s. In the discrete-state setting, assume the offline distribution has minimum positive mass $\rho \triangleq \inf_{s:d_\mu(s)>0} d_\mu(s) > 0$ and both $d_{\pi,t} \ll d_\mu$ and $d_{\pi_{\mathrm{ref}},t} \ll d_\mu$ for all $t \in [H]$. Then $C(\pi) \le C(\pi_{\mathrm{ref}}) + \frac{H\sqrt{2\varepsilon}}{\rho}$. If the reference policy is demo-concentrated, i.e., $\pi_{\mathrm{ref}}(\tilde{a}(s) \mid s) \ge 1 - \delta(s)$, then the off-demo validity mass satisfies $\bar{\eta}_\pi \le \bar{\delta} + \sqrt{\varepsilon/2}, \quad \bar{\delta} \triangleq \mathbb{E}_{s\sim d_\mu}[\delta(s)]$. Full statements and proofs are provided in Appendix F.2. More directly, $|\bar{\eta}_\pi - \bar{\eta}_{\pi_{\mathrm{ref}}}| \le \sqrt{\varepsilon/2}$, so a trust region limits changes in the unobserved ambiguity mass without requiring demo concentration. The small-ambiguity bound above is an additional consequence when the reference concentrates on demonstrations. Our training uses an empirical KL penalty, which encourages this behavior but does not directly enforce a uniform per-state constraint.

5.4 Success-adaptive Negative Gradient Scaling

Under partial verifiability, positive rewards are reliable but $\tilde{r}(s,a) = 0$ is ambiguous, since it conflates truly invalid actions with valid-but-uncredited alternatives. Treating all non-matches as equally negative can therefore produce biased and overly aggressive updates. To address this issue, we propose *success-adaptive negative gradient scaling (SNGS)*, which conservatively downweights gradients induced by ambiguous "negative" outcomes rather than suppressing them uniformly.

Concretely, GRPO samples a group of G candidate actions for the same state and computes group-relative advantages. Let $\{(a_k, \tilde{r}_k)\}_{k=1}^{G}$ denote a GRPO group at state s, where $\tilde{r}_k = \mathbf{1}\{a_k = \tilde{a}(s)\} \in \{0,1\}$, and define the empirical group success rate as $\hat{p}_g(s) \triangleq \frac{1}{G}\sum_{k=1}^{G} \tilde{r}_k$.

We introduce a scaling factor $\lambda_g(s)$ that rescales only the *negative* advantages. $\lambda_g(s) \triangleq \min(\lambda_0 + \kappa\, \hat{p}_g(s),\ 1)$. Here λ_0 is an offset and κ controls how λ_g varies with $\hat{p}_g$. With $\kappa > 0$, λ_g increases with $\hat{p}_g$, so as the policy becomes more concentrated on $\tilde{a}(s)$, SNGS downweights negative gradients less and gradually recovers standard GRPO. Let A_k denote the GRPO advantage for sample k. SNGS modifies only the negative advantages: $\tilde{A}_k \triangleq \begin{cases} A_k, & A_k \ge 0, \\ \lambda_g(s)\, A_k, & A_k < 0. \end{cases}$ We then replace the advantage term in the GRPO objective with $\tilde{A}_k$. SNGS thus preserves positive learning signals from reliably verified matches while attenuating updates driven by ambiguous negatives, reducing over-penalization of valid alternatives.

Why retain single-demonstration rewards? Our approach controls how an imperfect reward drives learning rather than attempting to enumerate every valid action. KL limits drift from the reference policy, while SNGS reduces the pressure to suppress valid-but-uncredited alternatives. SNGS does not identify which negatives are valid; it attenuates their influence through group-level success. An alternative is to replace matching with a learned judge. In Appendix E.6, this direct approach produces rapidly increasing judge reward alongside collapsing held-out matching accuracy, motivating our use of rule-based rewards with conservative updates.

Table 3: Performance on the online benchmark **AndroidWorld** in 20 steps. * denotes numbers reported by [24]. Left: *Native Models* (single VLM). Right: *Agent Frameworks* ($\geq$2 VLM modules).

(a) Native Models

Model	Acc.
Qwen2.5-VL-32B	29.6
Qwen2.5-VL-72B	32.2
Qwen3-VL-32B	34.8
Qwen2.5-VL-3B (Baseline)	3.5
GUI-Libra-3B (Ours)	**25.2**
Qwen2.5-VL-7B (Baseline)	7.8
GUI-Libra-7B (Ours)	**29.6**
Qwen3-VL-4B (Baseline)	27.0
GUI-Libra-4B (Ours)	**42.6**
Qwen3-VL-8B (Baseline)	30.4
GUI-Libra-8B (Ours)	**42.6**

(b) Agent Frameworks ($\geq$2 VLM Modules)

Model	Additional Module	Acc.
Qwen2.5-VL-3B	Step-wise Summary	7.0
Qwen2.5-VL-7B	Step-wise Summary	15.7
Qwen3-VL-4B	Step-wise Summary	36.5
Qwen3-VL-8B	Step-wise Summary	39.1
ScaleCUA-3B*	Step-wise Summary	23.7
ScaleCUA-7B*	Step-wise Summary	27.2
ScaleCUA-32B*	Step-wise Summary	30.6
UITARS-1.5-7B*	Step-wise Summary	31.6
GPT-4o	UGround-v1-7B	42.6
GPT-4.1	UGround-v1-7B	37.4
GPT-5-mini	UGround-v1-7B	40.9
GPT-5	UGround-v1-7B	48.7

5.5 Overall Training Framework for GUI-Libra

Based on our augmented and filtered datasets, GUI-Libra has two stages. **In the SFT stage**, we apply ASFT to equip the base model with *action-aligned reasoning* while mitigating long-CoT grounding degradation through mixed supervision and action-aware reweighting. **In the RL stage**, we further optimize the policy with a conservative GRPO objective under partially verifiable step-wise rewards. Overall, GUI-Libra promotes step-wise improvements that are more *behaviorally meaningful*: gains in offline action matching are more likely to translate into better decisions along the policy's own trajectories. By leveraging partially verifiable offline feedback, it also enables scalable optimization on large static datasets without costly online interaction during training.

6 Experiments

We evaluate on diverse offline and online GUI navigation benchmarks, study the effects of our key SFT and RL design choices, and examine when offline step-wise metrics predict online task success.

Experimental Setups. We train GUI-Libra from Qwen2.5-VL-3B/7B-Instruct [4] and Qwen3-VL-4B/8B-Instruct [3], using GUI-Libra-81K for SFT and a downsampled 40K subset for RL. Our SFT corpus is much smaller than those of recent GUI models [50, 24] and excludes grounding-only data, focusing on GUI reasoning and multi-step navigation. Unless otherwise noted, ablations use KL-regularized GRPO. Training details are deferred to Appendix C. For **offline evaluation**, we refine AndroidControl [20] and MM-Mind2Web [56] into AndroidControl-v2 and Multimodal-Mind2Web-v2 (MM-Mind2Web-v2), and report step success together with Pass@1 and Pass@4. For **online evaluation**, we use AndroidWorld [34], WebArena-Lite-v2 [24], and Online-Mind2Web [46], following the official protocols and reporting task success rate. Additional benchmark details are deferred to Appendix D. We compare GUI-Libra against strong native GUI agents, proprietary models paired with grounding modules following UGround [13], and reported/reproduced ScaleCUA results on online benchmarks [24]. Because evaluation pipelines can materially affect performance, we use a unified open-source protocol whenever possible.

6.1 Performance on Offline and Online GUI Navigation Benchmarks

Offline Benchmarks. Table 10 summarizes step-wise accuracy on AC-v2 and MM-M2W-v2. On both benchmarks, GUI-Libra achieves the **best Pass@1**, and consistently improves over its base models by large margins. For example, GUI-Libra-3B improves Pass@1 over Qwen2.5-VL-3B by $+20.9/+14.8$ on AC-v2 (high/low) and by $+19.3$ on MM-M2W-v2. For **Pass@4**, very large models (e.g., Qwen2.5-VL-72B) can be competitive, but require far more parameters. GUI-Libra remains more parameter-efficient and consistently outperforms similarly sized models, with smaller but still substantial gains on stronger Qwen3 backbones.

Online Benchmarks. Tables 3, 14, and 4 report task success rates on AndroidWorld and Online-Mind2Web. Because evaluation frameworks can materially affect performance, we distinguish true native models from agent frameworks with extra modules such as step-wise summaries.

On **AndroidWorld** (Table 3), GUI-Libra improves all backbones substantially: GUI-Libra-3B rises from 3.5 to 25.2, and GUI-Libra-8B from 30.4 to 42.6. GUI-Libra-4B/8B (42.6) surpass several

Table 4: Comparison on **Online-Mind2Web** in 30 steps using o4-mini/WebJudge-7B as judge.

Model \ Judge	o4-mini Easy	o4-mini Medium	o4-mini Hard	o4-mini Overall	WebJudge-7B Easy	WebJudge-7B Medium	WebJudge-7B Hard	WebJudge-7B Overall	Avg. Overall
Agent Framework									
GPT-4o + UGround-v1-7B	38.8	14.0	6.5	18.7	45.0	27.3	19.5	30.0	24.3
GPT-4.1 + UGround-v1-7B	41.3	21.7	5.2	22.7	47.5	35.0	28.6	**36.7**	**29.7**
GPT-5 + UGround-v1-7B	40.0	24.5	14.3	**26.0**	45.0	31.5	24.7	33.3	**29.7**
Native Models									
Qwen2.5-VL-32B	12.5	7.7	1.3	7.3	28.8	14.7	19.5	19.7	13.5
Qwen3-VL-32B	33.8	17.5	7.8	19.3	45.0	31.5	28.6	34.3	26.8
ScaleCUA-3B	30.0	4.9	2.6	11.0	37.5	18.2	11.7	21.7	16.3
ScaleCUA-7B	33.8	14.7	3.9	17.0	47.5	27.3	18.2	30.3	23.7
ScaleCUA-32B	31.3	14.7	6.5	17.0	43.8	29.4	16.9	30.0	23.5
Qwen2.5-VL-3B	3.8	0.0	1.3	1.3	16.3	7.7	1.3	8.3	4.8
GUI-Libra-3B (Ours)	28.8	9.8	5.2	13.7	47.5	21.0	24.7	29.0	21.3
Qwen2.5-VL-7B	22.5	7.7	0.0	9.7	36.3	18.9	13.0	22.0	15.8
GUI-Libra-7B (Ours)	36.3	15.4	2.6	17.7	47.5	30.8	23.4	33.3	25.5
Qwen3-VL-4B	33.8	10.5	6.5	15.7	43.8	23.8	18.2	27.7	21.7
GUI-Libra-4B (Ours)	36.3	18.2	6.5	**20.0**	45.0	30.1	19.5	31.3	25.7
Qwen3-VL-8B	23.8	9.8	0.0	11.0	43.8	23.1	19.5	27.7	19.3
GUI-Libra-8B (Ours)	31.3	17.5	10.4	19.3	42.5	37.8	28.6	**36.7**	**28.0**

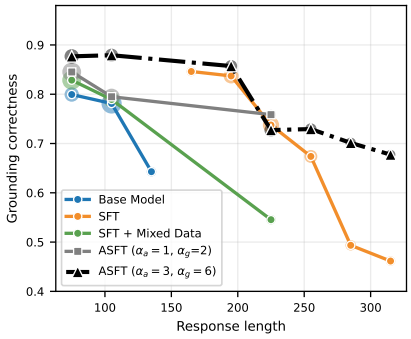


Figure 3: Grounding accuracy under different response lengths. Action-aware SFT strategies, including mixing direct-action data and weighted objectives, help preserve grounding accuracy under long CoT outputs.

Model Name	Inference Mode	Average Tokens	Grounding Accuracy (%)
SFT-3B	Reason	223.4	73.4
SFT 3B + Mixed Data	No-Reason	77.3	79.4
SFT 3B + Mixed Data	Reason	200.6	73.8
ASFT 3B	No-Reason	67.7	81.0
ASFT 3B	Reason	200.2	76.2
GUI-Libra-3B (ASFT+RL)	No-Reason	59.0	83.2
GUI-Libra-3B (ASFT+RL)	Reason	206.5	83.4
SFT-7B	Reason	218.2	79.0
SFT 7B + Mixed Data	No-Reason	69.4	85.6
SFT 7B + Mixed Data	Reason	168.1	81.4
ASFT 7B	No-Reason	76.3	86.8
ASFT 7B	Reason	169.6	83.4
GUI-Libra-7B (ASFT+RL)	No-Reason	124.4	88.5
GUI-Libra-7B (ASFT+RL)	Reason	176.1	89.3

Table 5: Grounding accuracy and average response tokens across different models and inference modes. "Reason" and "No-Reason" indicate whether explicit reasoning mode is encouraged through prompting.

much larger native models and matches or outperforms multi-module agent frameworks, while remaining a single-VLM system. On **Online-Mind2Web** (Table 4), evaluated on live websites, GUI-Libra again improves all base models. GUI-Libra-8B raises the average score from 19.3 to 28.0, the best among native models, and GUI-Libra-4/7/8B all exceed ScaleCUA. Even at 3B scale, GUI-Libra-3B reaches 21.3 versus 4.8 for Qwen2.5-VL-3B. Overall, GUI-Libra yields consistent gains across mobile and web environments while remaining highly data-efficient.

6.2 Action-aware SFT and RL Mitigate Grounding Performance Degradation

Figure 3 analyzes grounding performance as a function of response length on ScreenSpot-v2. As response length increases, both the base model and standard SFT exhibit pronounced degradation, indicating that long-form reasoning interferes with precise action execution. In contrast, ***action-aware SFT substantially mitigates this degradation across all response lengths***, and stronger weighting strategies preserve grounding accuracy even beyond 250 tokens.

Table 5 reports overall grounding accuracy and average response length across models and inference modes. Across both 3B and 7B scales, ***mixed-data training and action-aware weighting improve grounding in both reasoning and no-reasoning modes***, but ***ASFT alone does not fully close the gap between them***. RL largely removes this gap: GUI-Libra reaches comparable, and sometimes better, grounding accuracy in reasoning mode despite longer responses, even though RL uses only step-wise navigation data rather than direct grounding supervision [28, 27]. This shows that ***RL further aligns reasoning with grounding and complements ASFT.***

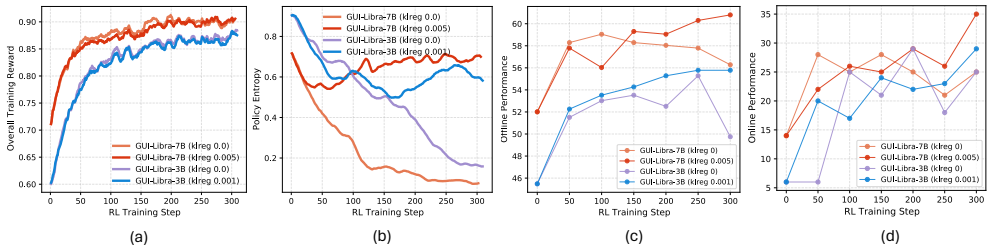


Figure 4: Comparison of training and evaluation metrics with and without KL regularization: (a) training reward, (b) policy entropy during training, (c) offline evaluation performance on AndroidControl-High, and (d) online evaluation performance on AndroidWorld.

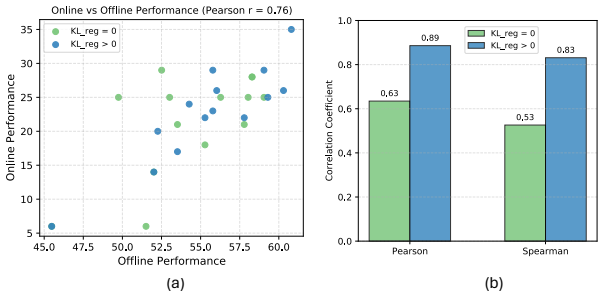


Figure 5: (a) Correlation between offline and online performance. (b) Comparison of Pearson and Spearman correlations with and without KL regularization.

6.3 On the Effectiveness of KL Regularization for RL

Section 5.3.1 motivates KL as a control on distributional drift and changes in ambiguity mass. We assess its effect on offline-to-online predictability empirically. We test this on AndroidControl-High and AndroidWorld in Figure 4, comparing RL runs with and without KL regularization from the same initialization. Training rewards are similar across settings, but evaluation behavior differs markedly: without KL, offline and online performance degrade despite increasing reward, and policy entropy collapses; with a small KL penalty, entropy is more stable and evaluation performance is substantially better. Figure 5 further plots offline and online scores for intermediate 3B and 7B checkpoints. The relationship is approximately linear overall, but much tighter with KL regularization: KL-regularized checkpoints achieve Pearson/Spearman correlations of 0.89 and 0.83, versus 0.63 and 0.53 without KL. Thus, ***KL regularization improves both training stability and offline-to-online alignment.***

6.4 Ablations

Table 6: Ablations with a shared Qwen2.5-VL-3B backbone and curated data pool. Rows isolate SFT variants, RL from base model, KL regularization, and CoT.

Model	MM-Mind2Web-v2		AC-v2 (High)		AC-v2 (Low)		AndroidWorld
	Pass@1	Pass@4	Pass@1	Pass@4	Pass@1	Pass@4	
Base Model	23.4	28.3	36.4	50.8	71.1	79.2	3.5
SFT	28.5	36.9	45.7	59.8	73.1	83.2	5.2
SFT+ Mixed Data	30.2	42.0	45.5	64.8	72.6	85.4	11.3
ASFT	32.0	41.3	44.5	64.6	75.4	86.9	13.0
GUI-Libra w/o ASFT (KL_reg=0.0)	40.9	45.1	50.5	54.8	78.6	81.9	17.4
GUI-Libra w/o ASFT (KL_reg=0.001)	41.9	48.0	57.0	63.6	86.2	90.0	20.9
GUI-Libra (KL_reg=0.0)	**43.8**	49.2	49.8	58.0	87.2	91.0	21.7
GUI-Libra (KL_reg=0.001)	42.7	50.2	**55.8**	65.8	**89.5**	91.5	**25.2**
GUI-Libra (KL_reg=0.01)	43.4	50.6	51.5	64.8	85.9	92.0	21.7
GUI-Libra (KL_reg=0.05)	41.4	**51.1**	49.8	**66.1**	87.9	**92.2**	20.0
ASFT w/o CoT	35.2	41.4	40.2	56.0	71.4	87.7	5.2
GUI-Libra (KL_reg=0.001) w/o CoT	43.4	47.1	48.7	55.8	85.2	87.7	12.2
ASFT infer w/o CoT	37.1	43.8	42.7	58.3	75.6	89.2	8.7
GUI-Libra (KL_reg=0.001) infer w/o CoT	42.5	48.7	52.0	59.3	88.2	**92.2**	18.3

Ablations on ASFT and RL for Navigation Tasks. Table 6 shows steady gains from the base model to SFT, mixed-data SFT, and ASFT. Mixed supervision and action-aware weighting raise Pass@1

from 23.4 to 32.0 on MM-Mind2Web-v2, from 36.4 to 44.5 on AndroidControl-v2 (High), and from 3.5 to 13.0 on AndroidWorld. RL then adds another large jump: with $\mathrm{KL} = 0.001$, GUI-Libra improves over ASFT by 10.7/8.9 points on MM-Mind2Web-v2 Pass@1/Pass@4 and from 13.0 to 25.2 on AndroidWorld. ASFT and RL are therefore complementary.

Matched-data comparisons and component roles. Standard SFT, direct RL from the base model, and the full recipe use the same curated data pool and backbone, reaching 5.2, 20.9, and 25.2 on AndroidWorld, respectively. This comparison fixes the data pool and backbone; training budgets vary with the recipe. Within SFT, mixed supervision supplies the largest online gain ($5.2 \rightarrow 11.3$); token reweighting adds 1.7 points and protects grounding under long reasoning traces. RL supplies the largest subsequent gain ($13.0 \rightarrow 25.2$), while removing ASFT initialization costs 4.3 points. The data alone therefore does not explain the result: how the model learns from it determines how well offline gains transfer online.

Ablations of KL Regularization Coefficient. The KL coefficient trades off offline and online performance. Increasing KL generally improves Pass@4 while slightly reducing Pass@1, consistent with stronger entropy preservation. Moderate regularization ($\mathrm{KL} = 0.001$) gives the best overall trade-off, whereas removing KL or using overly large penalties reduces AndroidWorld performance to around 20. This indicates that ***moderate KL best balances distribution shift and reward ambiguity***.

Ablations of Reasoning in Model Training and Inference. Removing CoT during both training and inference substantially hurts performance, especially on AndroidWorld, where GUI-Libra drops from 25.2 to 12.2 and ASFT from 13.0 to 5.2. Removing CoT only at inference is less harmful but still degrades online performance ($13.0 \rightarrow 8.7$ for ASFT and $25.2 \rightarrow 18.3$ for GUI-Libra). Explicit reasoning during both training and inference therefore remains important for online generalization.

Ablation of SNGS. Table 15 shows that SNGS consistently improves online generalization, boosting GUI-Libra-4B from 39.1 to 42.6 on AndroidWorld and from 22.2 to 24.4 on WebArena-Lite-v2. Offline gains are smaller and concentrated on more reasoning-heavy settings such as MM-Mind2Web-v2 and AndroidControl-v2 (High), with minor trade-offs on low-level metrics. This suggests that SNGS reduces overfitting to short-horizon action matching and favors more robust online behavior.

7 Conclusion

We introduce GUI-Libra, a unified framework for training reasoning-capable native GUI agents with our curated GUI-Libra-81K dataset. Our central finding is that competitive long-horizon navigation can be achieved by fully leveraging existing trajectory corpora: action-aware SFT preserves grounding under long reasoning traces, while conservative RL improves decision-making under partially verifiable feedback by controlling policy drift. GUI-Libra achieves strong offline and online performance across diverse mobile and web benchmarks with favorable data and parameter efficiency, without expensive online interaction during training, and makes offline evaluation more reliable and predictive of online success. We hope our findings and released resources encourage further work on data-efficient, reliable learning for real-world GUI agents.

Limitations

Rewards based on a single demonstration remain partially verifiable: our method uses conservative updates to mitigate this issue but does not identify all valid alternatives. Our analysis assumes valid demonstrations and population coverage, while training relies on finite data and an empirical KL penalty. We also do not study continual online learning. Incorporating online interaction data [14] and developing more reliable step-level verification are important directions for future work.

Acknowledgments and Disclosure of Funding

This work was partially supported in part by NSF grants 2416897, 2505932, and 2601666 and ONR grant N000142512318. Computing resources were provided by the National Center for Supercomputing Applications (NCSA) through Delta (NSF OAC-2005572) and DeltaAI (NSF OAC-2320345), as well as Illinois Computes and the NAIRR Pilot (NAIRR250157). Huan Zhang is supported in part by the AI Safety Fund administered by the Meridian Institute and the Frontier Model Forum (FMF).

References

[1] Hao Bai, Alexey Taymanov, Tong Zhang, Aviral Kumar, and Spencer Whitehead. Webgym: Scaling training environments for visual web agents with realistic tasks. *arXiv preprint arXiv:2601.02439*, 2026.

[2] Hao Bai, Yifei Zhou, Jiayi Pan, Mert Cemri, Alane Suhr, Sergey Levine, and Aviral Kumar. DigiRL: Training in-the-wild device-control agents with autonomous reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[3] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025.

[4] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[5] Yuxiang Chai, Siyuan Huang, Yazhe Niu, Han Xiao, Liang Liu, Guozhi Wang, Dingyu Zhang, Shuai Ren, and Hongsheng Li. AMEX: Android multi-annotation expo dataset for mobile GUI agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

[6] Hanyang Chen, Mark Zhao, Rui Yang, Qinwei Ma, Ke Yang, Jiarui Yao, Kangrui Wang, Hao Bai, Zhenhailong Wang, Rui Pan, Mengchao Zhang, Jose Barreiros, Aykut Onol, ChengXiang Zhai, Heng Ji, Manling Li, Huan Zhang, and Tong Zhang. Era: Transforming vlms into embodied agents via embodied prior learning and online reinforcement learning, 2025.

[7] Wentong Chen, Junbo Cui, Jinyi Hu, Yujia Qin, Junjie Fang, Yue Zhao, Chongyi Wang, Jun Liu, Guirong Chen, Yupeng Huo, Yuan Yao, Yankai Lin, Zhiyuan Liu, and Maosong Sun. GUICourse: From general vision language model to versatile GUI agent. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025.

[8] Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. SeeClick: Harnessing GUI grounding for advanced visual GUI agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

[9] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[10] Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. Openvlthinker: Complex vision-language reasoning via iterative sft-rl cycles, 2025.

[11] Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training, 2025.

[12] Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pages 10835–10866. PMLR, 2023.

[13] Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for GUI agents. In *The Thirteenth International Conference on Learning Representations*, 2025.

[14] Yifei He, Rui Yang, Hao Bai, Tong Zhang, and Han Zhao. Pro-cua: Process-reward optimization for computer use agents. *arXiv preprint arXiv:2605.29119*, 2026.

[15] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents, 2023.

[16] Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, et al. Glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025.

[17] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

[18] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[19] Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. Screenspot-pro: Gui grounding for professional high-resolution computer use. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 8778–8786, 2025.

[20] Wei Li, William E Bishop, Alice Li, Christopher Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on UI control agents. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

[21] Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Weixian Lei, Lijuan Wang, and Mike Zheng Shou. Showui: One vision-language-action model for gui visual agent, 2024.

[22] Jijia Liu, Feng Gao, Bingwen Wei, Xinlei Chen, Qingmin Liao, Yi Wu, Chao Yu, and Yu Wang. What can rl bring to vla generalization? an empirical study, 2026.

[23] Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Song XiXuan, Yifan Xu, Shudan Zhang, Hanyu Lai, Jiadai Sun, Xinyue Yang, Yu Yang, Zehan Qi, Shuntian Yao, Xueqiao Sun, Siyi Cheng, Qinkai Zheng, Hao Yu, Hanchen Zhang, Wenyi Hong, Ming Ding, Lihang Pan, Xiaotao Gu, Aohan Zeng, Zhengxiao Du, Chan Hee Song, Yu Su, Yuxiao Dong, and Jie Tang. Visualagentbench: Towards large multimodal models as visual foundation agents. In *The Thirteenth International Conference on Learning Representations*, 2025.

[24] Zhaoyang Liu, Jingjing Xie, Zichen Ding, Zehao Li, Bowen Yang, Zhenyu Wu, Xuehui Wang, Qiushi Sun, Shi Liu, Weiyun Wang, Shenglong Ye, Qingyun Li, Xuan Dong, Yue Yu, Chenyu Lu, YunXiang Mo, Yao Yan, Zeyue Tian, Xiao Zhang, Yuan Huang, Yiqian Liu, Weijie Su, Gen Luo, Xiangyu Yue, Biqing Qi, Kai Chen, Bowen Zhou, Yu Qiao, Qifeng Chen, and Wenhai Wang. Scalecua: Scaling open-source computer use agents with cross-platform data. *arXiv preprint arXiv:2509.15221*, 2025. Preprint.

[25] Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025.

[26] Quanfeng Lu, Wenqi Shao, Zitao Liu, Fanqing Meng, Boxuan Li, Botong Chen, Siyuan Huang, Kaipeng Zhang, Yu Qiao, and Ping Luo. Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices. *arXiv preprint arXiv:2406.08451*, 2024.

[27] Zhengxi Lu, Yuxiang Chai, Yaxuan Guo, Xi Yin, Liang Liu, Hao Wang, Guanjing Xiong, and Hongsheng Li. Ui-r1: Enhancing action prediction of gui agents by reinforcement learning. *arXiv preprint arXiv:2503.21620*, 2025.

[28] Run Luo, Lu Wang, Wanwei He, and Xiaobo Xia. Gui-r1: A generalist r1-style vision-language action model for gui agents. *arXiv preprint arXiv:2504.10458*, 2025.

[29] OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoochian, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, Dane Sherburn, Daniel Kappler, Daniel Levin, Daniel Levy, David Carr, David Farhi, David Mely, David Robinson, David Sasaki, Denny Jin, Dev Valladares, Dimitris Tsipras, Doug Li, Duc Phong Nguyen, Duncan Findlay, Edede Oiwoh, Edmund Wong, Ehsan Asdar, Elizabeth Proehl, Elizabeth Yang, Eric Antonow, Eric Kramer, Eric Peterson, Eric Sigler, Eric Wallace, Eugene Brevdo, Evan Mays, Farzad Khorasani, Felipe Petroski Such, Filippo Raso, Francis Zhang, Fred von Lohmann, Freddie Sulit, Gabriel Goh, Gene Oden, Geoff Salmon, Giulio Starace, Greg Brockman, Hadi Salman, Haiming Bao, Haitang Hu, Hannah Wong, Haoyu Wang, Heather Schmidt, Heather Whitney, Heewoo Jun, Hendrik Kirchner, Henrique Ponde de Oliveira Pinto, Hongyu Ren, Huiwen Chang, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian O'Connell, Ian Osband, Ian Silber, Ian Sohl, Ibrahim Okuyucu, Ikai Lan, Ilya Kostrikov, Ilya Sutskever, Ingmar Kanitscheider, Ishaan Gulrajani, Jacob Coxon, Jacob Menick, Jakub Pachocki, James Aung, James Betker, James Crooks, James Lennon, Jamie Kiros, Jan Leike, Jane Park, Jason Kwon, Jason Phang, Jason Teplitz, Jason Wei, Jason Wolfe, Jay Chen, Jeff Harris, Jenia Varavva, Jessica Gan Lee, Jessica Shieh, Ji Lin, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joanne Jang, Joaquin Quinonero Candela, Joe Beutler, Joe Landers, Joel Parish, Johannes Heidecke, John Schulman, Jonathan Lachman, Jonathan McKay, Jonathan Uesato, Jonathan Ward, Jong Wook Kim, Joost Huizinga, Jordan Sitkin, Jos Kraaijeveld, Josh Gross, Josh Kaplan, Josh Snyder, Joshua Achiam, Joy Jiao, Joyce Lee, Juntang Zhuang, Justyn Harriman, Kai Fricke, Kai Hayashi, Karan Singhal, Katy Shi, Kavin Karthik, Kayla Wood, Kendra Rimbach, Kenny Hsu, Kenny Nguyen, Keren Gu-Lemberg, Kevin Button, Kevin Liu, Kiel Howe, Krithika Muthukumar, Kyle Luther, Lama Ahmad, Larry Kai, Lauren Itow, Lauren Workman, Leher Pathak, Leo Chen, Li Jing, Lia Guy, Liam Fedus, Liang Zhou, Lien Mamitsuka, Lilian Weng, Lindsay McCallum, Lindsey Held, Long Ouyang, Louis Feuvrier, Lu Zhang, Lukas Kondraciuk, Lukasz Kaiser, Luke Hewitt, Luke Metz, Lyric Doshi, Mada Aflak, Maddie Simens, Madelaine Boyd, Madeleine Thompson, Marat Dukhan, Mark Chen, Mark Gray, Mark Hudnall, Marvin Zhang, Marwan Aljubeh, Mateusz Litwin, Matthew Zeng, Max Johnson, Maya Shetty, Mayank Gupta, Meghan Shah, Mehmet Yatbaz, Meng Jia Yang, Mengchao Zhong, Mia Glaese, Mianna Chen, Michael Janner, Michael Lampe, Michael Petrov, Michael Wu, Michele Wang, Michelle Fradin, Michelle Pokrass, Miguel Castro, Miguel Oom Temudo de Castro, Mikhail Pavlov, Miles Brundage, Miles Wang, Minal Khan, Mira Murati, Mo Bavarian, Molly Lin, Murat Yesildal, Nacho Soto, Natalia Gimelshein, Natalie Cone, Natalie Staudacher, Natalie Summers, Natan LaFontaine, Neil Chowdhury, Nick Ryder, Nick Stathas, Nick Turley, Nik Tezak, Niko Felix, Nithanth Kudige, Nitish Keskar, Noah Deutsch, Noel Bundick, Nora Puckett, Ofir Nachum, Ola Okelola, Oleg Boiko, Oleg Murk, Oliver Jaffe, Olivia Watkins, Olivier Godement, Owen Campbell-Moore, Patrick Chao, Paul McMillan, Pavel Belov, Peng Su, Peter Bak, Peter Bakkum, Peter Deng, Peter Dolan, Peter Hoeschele, Peter Welinder, Phil Tillet, Philip Pronin, Philippe Tillet, Prafulla Dhariwal, Qiming Yuan, Rachel Dias, Rachel Lim, Rahul Arora, Rajan Troll, Randall Lin, Rapha Gontijo Lopes, Raul Puri, Reah Miyara, Reimar Leike, Renaud Gaubert, Reza Zamani, Ricky Wang, Rob Donnelly, Rob Honsby, Rocky Smith, Rohan Sahai, Rohit Ramchandani, Romain Huet, Rory Carmichael, Rowan Zellers, Roy Chen, Ruby Chen, Ruslan Nigmatullin, Ryan Cheu, Saachi Jain, Sam Altman, Sam Schoenholz, Sam Toizer, Samuel Miserendino, Sandhini Agarwal, Sara Culver, Scott Ethersmith, Scott Gray, Sean Grove, Sean Metzger, Shamez Hermani, Shantanu

Jain, Shengjia Zhao, Sherwin Wu, Shino Jomoto, Shirong Wu, Shuaiqi, Xia, Sonia Phene, Spencer Papay, Srinivas Narayanan, Steve Coffey, Steve Lee, Stewart Hall, Suchir Balaji, Tal Broda, Tal Stramer, Tao Xu, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Cunninghman, Thomas Degry, Thomas Dimson, Thomas Raoux, Thomas Shadwell, Tianhao Zheng, Todd Underwood, Todor Markov, Toki Sherbakov, Tom Rubin, Tom Stasi, Tomer Kaftan, Tristan Heywood, Troy Peterson, Tyce Walters, Tyna Eloundou, Valerie Qi, Veit Moeller, Vinnie Monaco, Vishal Kuo, Vlad Fomenko, Wayne Chang, Weiyi Zheng, Wenda Zhou, Wesam Manassra, Will Sheu, Wojciech Zaremba, Yash Patil, Yilei Qian, Yongjik Kim, Youlong Cheng, Yu Zhang, Yuchen He, Yuchen Zhang, Yujia Jin, Yunxing Dai, and Yury Malkov. Gpt-4o system card, 2024.

[30] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[31] Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent q: Advanced reasoning and learning for autonomous ai agents, 2024.

[32] Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning, 2025.

[33] Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025.

[34] Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William E Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Kenji Toyama, Robert James Berry, Divya Tyamagundlu, Timothy P Lillicrap, and Oriana Riva. Androidworld: A dynamic benchmarking environment for autonomous agents. In *The Thirteenth International Conference on Learning Representations*, 2025.

[35] Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. Android in the wild: a large-scale dataset for android device control. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, 2023.

[36] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[37] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[38] Haoming Wang, Haoyang Zou, Huatong Song, Jiazhan Feng, Junjie Fang, Junting Lu, Longxiang Liu, Qinyu Luo, Shihao Liang, Shijue Huang, Wanjun Zhong, Yining Ye, Yujia Qin, Yuwen Xiong, Yuxin Song, Zhiyong Wu, Aoyan Li, Bo Li, Chen Dun, Chong Liu, Daoguang Zan, Fuxing Leng, Hanbin Wang, Hao Yu, Haobin Chen, Hongyi Guo, Jing Su, Jingjia Huang, Kai Shen, Kaiyu Shi, Lin Yan, Peiyao Zhao, Pengfei Liu, Qinghao Ye, Renjie Zheng, Shulin Xin, Wayne Xin Zhao, Wen Heng, Wenhao Huang, Wenqian Wang, Xiaobo Qin, Yi Lin, Youbin Wu, Zehui Chen, Zihao Wang, Baoquan Zhong, Xinchun Zhang, Xujing Li, Yuanfan Li, Zhongkai Zhao, Chengquan Jiang, Faming Wu, Haotian Zhou, Jinlin Pang, Li Han, Qi Liu, Qianli Ma, Siyao Liu, Songhua Cai, Wenqi Fu, Xin Liu, Yaohui Wang, Zhi Zhang, Bo Zhou, Guoliang Li, Jiajun Shi, Jiale Yang, Jie Tang, Li Li, Qihua Han, Taoran Lu, Woyu Lin, Xiaokang Tong, Xinyao Li, Yichi Zhang, Yu Miao, Zhengxuan Jiang, Zili Li, Ziyuan Zhao, Chenxin Li, Dehua Ma, Feng Lin, Ge Zhang, Haihua Yang, Hangyu Guo, Hongda Zhu, Jiaheng Liu, Junda Du, Kai Cai, Kuanye Li, Lichen Yuan, Meilan Han, Minchao Wang, Shuyue Guo, Tianhao Cheng, Xiaobo Ma, Xiaojun Xiao, Xiaolong Huang, Xinjie Chen, Yidi Du, Yilin Chen, Yiwen Wang, Zhaojian Li, Zhenzhu Yang, Zhiyuan Zeng, Chaolin Jin, Chen Li, Hao Chen, Haoli Chen, Jian Chen, Qinghao Zhao, and Guang Shi. Ui-tars-2 technical report: Advancing gui agent with multi-turn reinforcement learning, 2025.

[39] Kangrui Wang, Pingyue Zhang, Zihan Wang, Yaning Gao, Linjie Li, Qineng Wang, Hanyang Chen, Chi Wan, Yiping Lu, Zhengyuan Yang, Lijuan Wang, Ranjay Krishna, Jiajun Wu, Li Fei-Fei, Yejin Choi, and Manling Li. Vagen: Reinforcing world model reasoning for multi-turn vlm agents, 2025.

[40] Xinyuan Wang, Bowen Wang, Dunjie Lu, Junlin Yang, Tianbao Xie, Junli Wang, Jiaqi Deng, Xiaole Guo, Yiheng Xu, Chen Henry Wu, Zhennan Shen, Zhuokai Li, Ryan Li, Xiaochuan Li, Junda Chen, Zheng Boyuan, LI PEIHANG, Fangyu Lei, Ruisheng Cao, Yeqiao Fu, Dongchan Shin, Martin Shin, Hu Jiarui, Yuyan Wang, Jixuan Chen, Yuxiao Ye, Danyang Zhang, Yipu Wang, Heng Wang, Diyi Yang, Victor Zhong, Y.Charles, Zhilin Yang, and Tao Yu. OpenCUA: Open foundations for computer-use agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[41] Yufei Wang, Wanjun Zhong, Liangyou Li, Fei Mi, Xingshan Zeng, Wenyong Huang, Lifeng Shang, Xin Jiang, and Qun Liu. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*, 2023.

[42] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[43] Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. OS-ATLAS: Foundation action model for generalist GUI agents. In *The Thirteenth International Conference on Learning Representations*, 2025.

[44] Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials. In *The Thirteenth International Conference on Learning Representations*, 2025.

[45] Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. Aguvis: Unified pure vision agents for autonomous GUI interaction. In *Forty-second International Conference on Machine Learning*, 2025.

[46] Tianci Xue, Weijian Qi, Tianneng Shi, Chan Hee Song, Boyu Gou, Dawn Song, Huan Sun, and Yu Su. An illusion of progress? assessing the current state of web agents. In *Second Conference on Language Modeling*, 2025.

[47] Rui Yang, Ruomeng Ding, Yong Lin, Huan Zhang, and Tong Zhang. Regularizing hidden states enables learning generalizable reward model for llms. *Advances in Neural Information Processing Systems*, 37:62279–62309, 2024.

[48] Rui Yang, Xiaoman Pan, Feng Luo, Shuang Qiu, Han Zhong, Dong Yu, and Jianshu Chen. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. *arXiv preprint arXiv:2402.10207*, 2024.

[49] Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe Huang, Amrita Saha, Zeyuan Chen, Ran Xu, Liyuan Pan, Silvio Savarese, Caiming Xiong, and Junnan Li. Gta1: Gui test-time scaling agent, 2025.

[50] Zhen Yang, Zi-Yi Dou, Di Feng, Forrest Huang, Anh Nguyen, Keen You, Omar Attia, Yuhao Yang, Michael Feng, Haotian Zhang, Ram Ramrakhya, Chao Jia, Jeffrey Nichols, Alexander Toshev, Yinfei Yang, and Zhe Gan. Ferret-ui lite: Lessons from building small on-device gui agents, 2025.

[51] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

[52] Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, and Sergey Levine. Fine-tuning large vision-language models as decision-making agents via reinforcement learning, 2024.

[53] Qiusi Zhan, Hyeonjeong Ha, Rui Yang, Sirui Xu, Hanyang Chen, Liang-Yan Gui, Yu-Xiong Wang, Huan Zhang, Heng Ji, and Daniel Kang. Visual backdoor attacks on mllm embodied decision making via contrastive trigger learning. *arXiv preprint arXiv:2510.27623*, 2025.

[54] Bofei Zhang, Zirui Shang, Zhi Gao, Wang Zhang, Rui Xie, Xiaojian Ma, Tao Yuan, Xinxiao Wu, Song-Chun Zhu, and Qing Li. Tongui: Internet-scale trajectories from multimodal web tutorials for generalized gui agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

[55] Jiwen Zhang, Jihao Wu, Teng Yihua, Minghui Liao, Nuo Xu, Xiao Xiao, Zhongyu Wei, and Duyu Tang. Android in the zoo: Chain-of-action-thought for GUI agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, 2024.

[56] Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v(ision) is a generalist web agent, if grounded. 2024.

[57] Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

[58] Hanzhang Zhou, Xu Zhang, Panrong Tong, Jianan Zhang, Liangyu Chen, Quyu Kong, Chenglin Cai, Chen Liu, Yue Wang, Jingren Zhou, and Steven Hoi. Mai-ui technical report: Real-world centric foundation gui agents, 2025.

[59] Yuqi Zhou, Sunhao Dai, Shuai Wang, Kaiwen Zhou, Qinglin Jia, and Jun Xu. GUI-g1: Understanding r1-zero-like training for visual grounding in GUI agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

A Additional Related Works

Reinforcement Learning from Verifiable Rewards (RLVR) To optimize LLMs, learning a reward model as the training signal is a common practice in reinforcement learning from human feedback (RLHF) [30, 41, 48]. However, reward models are known to suffer from reward hacking and misalignment issues [12, 47]. To address these limitations, recent work has shifted toward *verifiable rewards*, where supervision is derived from ground-truth verifiers, such as exact mathematical answer matching or code execution, rather than from a learned and potentially ambiguous reward model. [37] introduces GRPO, which enables effective policy optimization from such verifiable signals and demonstrates the emergence of complex reasoning behaviors. Building on this framework, DAPO [51] and Dr. GRPO [25] propose simple yet effective techniques, such as more aggressive clipping and dynamic sampling, to stabilize training and mitigate learning bias in GRPO. More recently, GSPO [57] leverages sequence-level importance sampling to further improve training stability, particularly for mixture-of-experts models. Despite their success, we find that directly applying RLVR recipes to step-wise agent training often leads to suboptimal policies, due to distribution shift and ambiguous intermediate rewards.

Post-training for VLM-based Agents. VLMs have demonstrated strong capabilities in visual perception and multimodal reasoning[29, 4, 3, 56]. However, deploying them as agents in visually grounded environments requires moving beyond static understanding toward robust, long-horizon decision-making. To bridge the gap, recent research adopts a two-stage SFT-then-RL paradigm[6, 52, 53]. In the first stage, SFT equips VLMs with essential agentic skills, including visual grounding, structured reasoning, and action prediction through curated dataset[15, 8, 43, 21, 33, 44]. Nevertheless, SFT is inherently limited by the coverage and diversity of demonstrations, therefore prone to compounding errors when encountering out-of-distribution states[6, 22, 10]. RL complements SFT by enabling agents to interact directly with environments. Through exploration, agents can learn from both successes and failures, gradually developing capabilities such as error recovery, self-correction, and long-horizon planning [2, 32, 31, 11, 39]. Under this two-stage paradigm, SFT first establishes a stable foundation of core skills, after which RL enhances long-horizon decision-making via environment interaction and policy optimization.

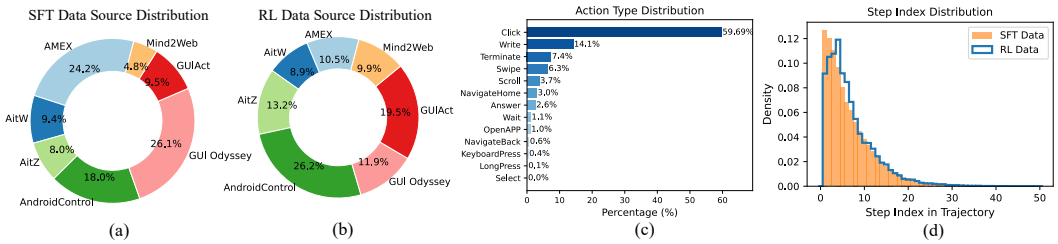


Figure 6: (a)(b) Data source distribution for SFT and RL. (c) Action type distribution of GUI-LIBRA-81K. (d) Comparison of step index distributions between our SFT and RL datasets.

B Detailed Data Processing

B.1 Data Sources

Although large-scale open-source datasets exist for GUI grounding [13, 43], trajectory-based GUI navigation data remain relatively scarce due to the high cost of collecting multi-step interaction traces. Following AGUVIS [45], we therefore aggregate trajectory data from multiple public sources that cover both web and mobile domains, including GUI-Odyssey [26], AMEX [5], AndroidControl [20], AitZ [55], AitW [35], GUIAct [7], and MM-Mind2Web [56]. Compared with the original AGUVIS collection, we additionally include the Chinese subset from GUIAct to broaden multilingual coverage and increase website diversity. Overall, these datasets span diverse applications and websites and include tasks with varying difficulty levels. We apply an initial cleaning stage to remove incomplete trajectories, extremely short or long traces (fewer than 3 steps or more than 50 steps), and steps containing compound actions that cannot be represented in our action space. After cleaning, we obtain 19K trajectories comprising 170K steps.

B.2 Unified Structured Format

Figure 8 summarizes our unified data format. Each sample contains an input and an output. The input includes a system prompt that enumerates available actions, the user instruction, the interaction history (previous actions), and the current screenshot. The output contains (1) a reasoning trace enclosed by `<think> ... </think>` and (2) a structured action enclosed by `<answer> ... </answer>`. The structured action is represented as a JSON object with an `action_type` and corresponding arguments, such as `value` for text entry and `point_2d` for click coordinates. We consider 13 common action types for web and mobile control: `Click`, `Write`, `Terminate`, `Swipe`, `Scroll`, `NavigateHome`, `Answer`, `Wait`, `OpenAPP`, `NavigateBack`, `KeyboardPress`, `LongPress`, and `Select`. The action optionally includes an `action_target`, a natural-language description of the UI element to interact with. In addition, we include an `action_description` that succinctly states the intended operation, which is appended to the interaction history for subsequent steps and captures a brief step-level rationale. Retaining `action_target` further supports downstream filtering by enabling consistency checks between the described target element and the coordinates from the original datasets. Details of the action space are provided in Appendix C.

B.2.1 Data Filtering for SFT

Human-collected and automatically labeled GUI trajectories are inevitably noisy [49, 45], including incorrect action types and inaccurate coordinates. To improve data quality, we curate our SFT data using a two-step automatic filtering pipeline.

(1) Agreement filtering via action re-prediction. We run Qwen3-VL-8B-Instruct for 10 stochastic runs on each input and measure how often the predicted action matches the annotation (e.g., exact match on `action_type` and coordinate proximity for `Click`-like actions). We discard steps with re-prediction accuracy below 0.3, which effectively removes uncertain or low-quality samples.

(2) Coordinate alignment via bounding-box verification. We leverage `action_target` to verify whether the original coordinate actually corresponds to the intended UI element, and to obtain bounding-box supervision for RL. Concretely, we prompt Qwen3-VL-32B-Instruct to predict a bounding box given the current screenshot and `action_target`, and keep a step only if the original

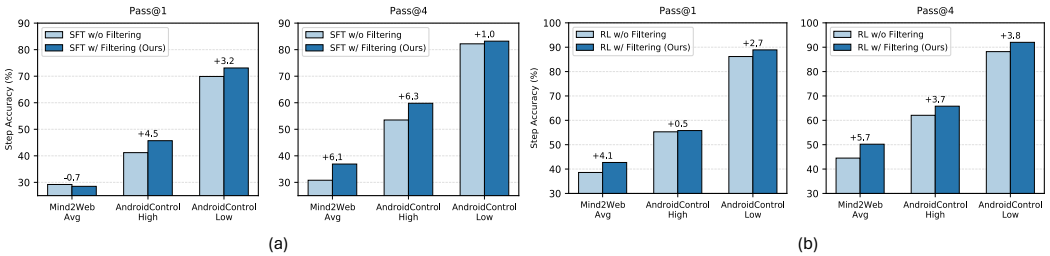


Figure 7: Ablation study of data filtering at the (a) SFT and (b) RL stages. Data filtering consistently improves both Pass@1 and Pass@4 across three benchmarks.

`point_2d` falls inside the predicted box. This filtering removes coordinate errors and reduces reasoning–action mismatch, while also providing reliable bounding-box annotations, often missing from prior datasets [45], for subsequent RL training.

SFT Dataset Statistics. After filtering, we obtain 81K SFT steps originating from 9K trajectories. Figure 6(a) shows the source distribution: most data comes from mobile datasets (e.g., AndroidControl, GUI-Odyssey, AMEX), while only 14.3% comes from the web domain. This reflects the current ecosystem where large-scale open mobile interaction data are more prevalent; scaling high-quality web trajectories remains an important direction. Figure 6(b) shows the action distribution: `Click` accounts for around 60% of steps, followed by `Write`, `Terminate`, and `Swipe`, while `LongPress` and `Select` are rare. This imbalance makes rare actions difficult to learn from SFT alone, motivating a subsequent RL stage.

B.2.2 Data Filtering for RL

For RL, we prioritize a more balanced training set by reducing biases in both step index and domain. Specifically, we address two issues: (i) **early-step bias**, where many trajectories share similar initial screens and actions (e.g., mobile home screens or common web landing pages), and (ii) **domain imbalance**, where mobile trajectories dominate the pool. To mitigate these effects, we downsample early steps (small step indices) and further downsample mobile-domain trajectories, resulting in a **40K**-step dataset for RL training. Figure 6(d) compares the step-index distributions, while Figure 6(a) compares the domain distributions. Overall, the RL subset is substantially more balanced than the SFT dataset.

B.3 Long-CoT Grounding Study Protocol

For the response-length analysis in Section 5, we fine-tune Qwen2.5-VL-3/7B-Instruct models to generate reasoning before grounding on GUI-LIBRA-81K, and evaluate them on ScreenSpot-v2 [43]. To induce a broad range of response lengths, we sample with temperature 1.0, then group outputs into 30-token bins and discard bins with fewer than 20 samples. This protocol lets us estimate how grounding accuracy changes as reasoning becomes longer while avoiding overly noisy bins.

B.3.1 Ablation of Data Filtering

Figure 7 reports data-filtering ablations using a 3B base model. Across most settings, filtering consistently improves performance, with especially large gains in Pass@4. For example, in SFT, filtering improves AndroidControl-High by $+4.5$ Pass@1 and $+6.3$ Pass@4, while in RL it yields an additional $+0.5$ Pass@1 and $+3.7$ Pass@4. These results highlight the importance of data quality in both SFT and RL: a smaller but cleaner and less biased dataset can generalize better than a larger, noisier corpus.

C Implementation Details

Data Example. We show the example data format in Figure 8. Each sample includes the current visual observation (screenshot) and textual context (system prompt, user instruction, and interaction

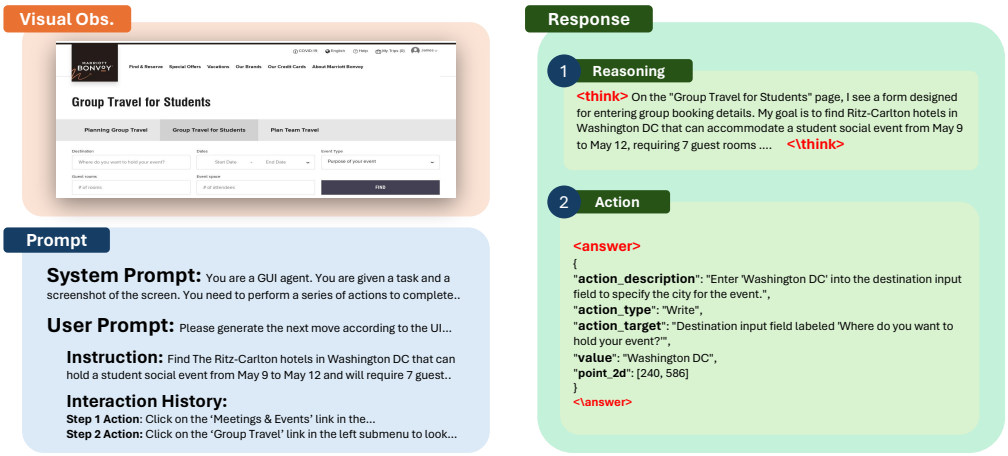


Figure 8: Example data format in GUI-LIBRA-81K.

history/previous actions). The model output is split into (1) a CoT reasoning trace and (2) a structured executable action (JSON), specifying the action type, a brief action description, the target element (if available), and action arguments such as text values or coordinates.

Action Space. We model GUI interaction with a unified action space where each step outputs a structured tuple `(action_type, action_target, value, point_2d)`. Here, `action_type` specifies the operation (e.g., `Click`, `Write`, `Scroll`), `action_target` describes the target UI element when applicable, and `value` provides additional arguments such as input text, scroll/swipe direction, key name, waiting time, or app name. For actions that require spatial grounding (e.g., `Click`, `LongPress`, and optionally `Swipe` or `Write`), `point_2d` records the screen coordinate `[x,y]`; otherwise it is set to `None`. This unified schema supports both web and mobile environments, including device-level controls (e.g., `NavigateBack`, `NavigateHome`, `OpenApp`) and a terminal action `Terminate`. See Table 7 for the full specification. Following the base model's coordinate system, we use absolute pixel coordinates for Qwen2.5-VL-based models, and normalized coordinates in $[0, 1000]$ for Qwen3-VL-based models.

Reward Function Implementation Each rollout produces a structured prediction string y following our output structure:

$$y = \texttt{<think>}\cdots\texttt{</think><answer>}a\texttt{</answer>},$$

where the `<answer>` block contains a structured action a that can be parsed into a JSON object

$$a = \{\texttt{action_type},\ \texttt{action_description},\ \texttt{value},\ \texttt{point_2d}\}.$$

with $\texttt{point_2d} \in \mathbb{R}^2$ (or `"none"` when not applicable). We implement two automated verifiers: a *format verifier* and an *accuracy verifier*. The resulting step-wise reward is a weighted sum of the two:

$$\tilde{r}(s,a) \;=\; w_{\text{fmt}}\, r_{\text{fmt}} \;+\; (1 - w_{\text{fmt}})\, r_{\text{acc}}, \qquad w_{\text{fmt}} \in [0,1], \tag{3}$$

where r_{fmt} checks output validity and r_{acc} scores action correctness. We set $w_{\text{fmt}} = 0.1$ so that our reward mainly focus on action correctness.

Format reward. The format reward r_{fmt} is 1 if the output contains valid `<think>` and `<answer>` tags and the `<answer>` block can be parsed into the required JSON schema; otherwise $r_{\text{fmt}} = 0$.

Accuracy reward. The accuracy reward r_{acc} evaluates semantic correctness of the predicted action: $r_{\text{acc}} = r_{\text{act}} \cdot r_{\text{val}} \cdot r_{\text{g}}$, where each component is computed as follows: (1) **Action-type reward** r_{act} checks whether `action_type` matches the demonstrated action type. (2) **Value reward** r_{val} compares the predicted `value` v with the demonstrated value $v^\star$ using word-level F1, and sets $r_{\text{val}} = 1$ if $\text{F1}(v, v^\star) > 0.5$. (3) **Grounding reward** r_{g} evaluates point grounding by checking whether the predicted point $\mathbf{u}$ falls inside the demonstrated bounding box $b^\star$, i.e., $r_{\text{g}} = \mathbf{1}\{\mathbf{u} \in b^\star\}$.

Together, these verifiers yield a step-wise signal: positive rewards indicate reliably correct predictions, whereas low rewards may arise from either incorrect actions or valid but uncredited alternatives. This design matches the partial-verifiability setting analyzed in Sec. 5.3.

Table 7: Unified action space. Each action is a tuple (`action_type, action_target, value, point_2d`).

action_type	action_target	value	point_2d	details
Answer	None	answer text	`[-100,-100]`	Return the final answer to the user's question.
Click	element description	None	`[x,y]`	Tap/click a specific UI element and provide its coordinates.
Select	element description	option value	`[-100,-100]`	Select an item in a list or dropdown menu.
LongPress	element description	None	`[x,y]`	Press-and-hold on a UI element (mobile only) and provide its coordinates.
Write	element description or None	input text	`[x,y]` or `[-100,-100]`	Enter text into a specific input field; if `point_2d` is `[-100,-100]`, type at the current focus.
KeyboardPress	None	key name (e.g., `enter`)	`[-100,-100]`	Press a specific key on the keyboard.
Scroll	None	direction (`up`/`down`/`left`/`right`)	`[-100,-100]`	Scroll a view/container in the specified direction.
Swipe	element description or None	direction (`up`/`down`/`left`/`right`)	`[x,y]` or `[-100,-100]`	Perform a swipe gesture on a touchscreen in the given direction; provide coordinates if applicable.
Wait	None	seconds	`[-100,-100]`	Pause execution for a specified duration to allow UI updates.
NavigateHome	None	None	`[-100,-100]`	Navigate to the device's home screen.
NavigateBack	None	None	`[-100,-100]`	Press the system "Back" button.
OpenApp	None	app name	`[-100,-100]`	Launch an app by its name (mobile only).
Terminate	None	end-task message	`[-100,-100]`	Signal the end of the current task with a final message.

SFT. We summarize our shared SFT and Action-aware SFT implementation parameters in Table 8. We apply full parameter tuning on Qwen2.5-VL and Qwen3-VL base models from 3B,4B, to 7B and 8B. We use a learning rate of 1×10^{-5} and an effective batch size of 256. We train SFT and ASFT models for either two epochs on GUI-Libra-81K or 1 epoch on mixing reasoning and direct-action data (it doubles data size). For action-aware SFT, we by default use $\alpha_a = 2$ and $\alpha_g = 4$ for ASFT, except for GUI-Libra-4B, where $\alpha_a = \alpha_g = 1$. We use 8 B200 GPUs for approximately 4 hours for Qwen3-VL-4B and 5.5 hours for Qwen3-VL-8B.

Category	Configuration
Backbone	Qwen2.5-VL / Qwen3-VL
Training Strategy	Full-parameter fine-tuning
Epochs	1
Learning Rate	1×10^{-5}
Scheduler	Cosine
Warmup Ratio	0.01
Weight Decay	0
Per-device Batch Size	4
Gradient Accumulation Steps	8
Effective Batch Size	256 (8 GPUs)
Gradient Checkpointing	Enabled

Table 8: SFT configuration used in our experiments.

RL. We adopt GRPO [37] as our RL algorithm, implemented with the `verl` framework[2] and EasyR1[3]. GUI-Libra is initialized from the SFT/ASFT checkpoints. At each iteration, we draw contexts from the fixed offline 40K subset, sample candidate reasoning-and-action responses from the current policy, compute step-wise rewards against the curated annotations, and update with GRPO. These model-generated responses require no environment interaction, new trajectories, or access to the behavior policy. We train for 300 RL iterations with a learning rate of 1×10^{-6}, global batch size

[2] https://github.com/verl-project/verl
[3] https://github.com/hiyouga/EasyR1

128, and rollout group size $n = 8$. We set the KL regularization coefficient to 0.001 by default and increase it to 0.005 for GUI-Libra-7B to improve training stability. For SNGS, we use model-specific (λ_0, κ): $(0.9, 0.5)$ for 3B, $(1.4, -0.5)$ for 7B, $(0.5, 1.5)$ for 4B, and $(0.5, 2.0)$ for 8B. Training is conducted on 8 NVIDIA B200 GPUs and takes approximately 16 hours for Qwen3-VL-4B and 20 hours for Qwen3-VL-8B.

Category	**Configuration**
RL Training Framework	VERL
Distributed Training Backend	FSDP
Inference Engine	vLLM
Backbone	Qwen2.5-VL, Qwen3-VL
Rollout	
Rollout Samples per Prompt	8
Rollout Batch Size	256
Sampling Strategy	Top-p 0.98
Temperature	1.0
Max Prompt Length	8092
Max Response Length	1500
Optimization	
Training Iterations	300
Learning Rate	1×10^{-6}
Optimizer	AdamW (bf16)
Global Batch Size	128
Micro Batch (Update)	4
Micro Batch (Experience)	8
Clip Ratio (ϵ)	0.2
KL Coefficient (β)	0.001 by default, and 0.005 for GUI-Libra-7B
Algorithm	
Advantage Estimator	GRPO, GRPO w/ SNGS
Reward Function	$\tilde{r}(s,a) = w_{\text{fmt}}\, r_{\text{fmt}} + (1 - w_{\text{fmt}})\, r_{\text{acc}}, w_{\text{fmt}} = 0.1$

Table 9: RL configuration for our experiments.

Evaluation. For inference, we use vLLM as the serving backend. We set the temperature to 0.0 and top-p to 1.0, and allow up to 1024 completion tokens by default. The system prompt follows Appendix G. The available action list within the system prompt can be adjusted depending on the deployment environment. For example, if a mobile environment does not support the `OpenAPP` action, it can be removed from the action list; in this case, the model typically resorts to alternative strategies such as scrolling to access the app drawer.

For models trained with mixed direct-action data, we optionally use an explicit instruction prompt to elicit direct action generation without intermediate reasoning. Specifically, we can append the following format instruction after user instruction:

```
The response should be structured in the following format.
Make sure the output between <answer> and </answer> is a valid JSON object.
Regarding the key "point_2d", please provide the coordinates on the screen
↪  where the action
is to be performed; if not applicable, use [-100, -100]:
<answer>
{
    "action\_description": "the description of the action to perform,
    ↪  summarized in one sentence",
    "action\_type": "the type of action to perform. Please follow the
    ↪  system prompt for
                available actions.",
```

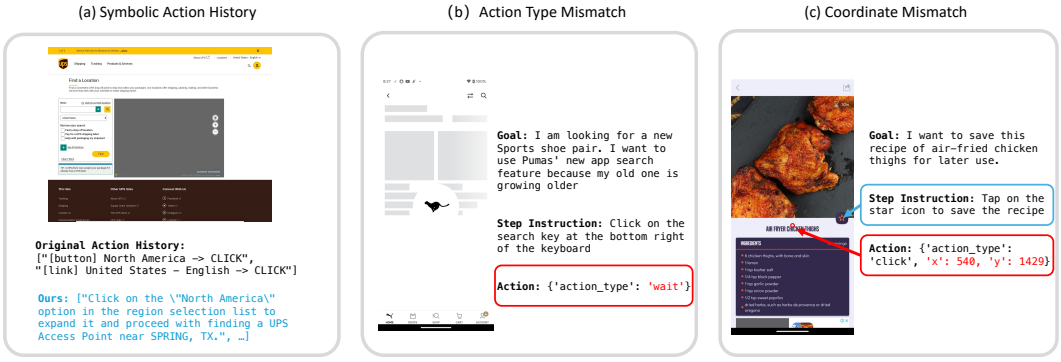


Figure 9: Limitations of current offline benchmarks. (a) Symbolic action history not in natural language in MM-Mind2Web, (b) Action type mismatch and (c) coordinate mismatch in AndroidControl.

```
    "value": "the input text or direction ('up', 'down', 'left', 'right')
    ↪  for the 'scroll' action
           or the app name for the 'openapp' action; otherwise, use
           ↪  'None'",
    "point\_2d": [x, y]
}
</answer>
```

D Benchmark Details

In our experiments, we adopt a diverse of benchmarks for evaluation, mainly focus on GUI navigation benchmarks that measures step-wise success or task-level completion. We also use grounding benchmarks to evaluate the correctness of grounding after long CoT generation. Details about these benchmarks are as follows.

D.1 Limitations in existing Benchmarks

As illustrated in Figure 9, the original MM-Mind2Web [56] uses symbolic action histories that do not reflect real-world usage, while AndroidControl [20] contains roughly 20% errors in action types and coordinates.

D.2 Grounding Benchmarks

We adopt ScreenSpot-V2 [8, 43] and ScreenSpot-Pro [19] to evaluate grounding accuracy. Each task provides a short instruction specifying the target element or intent, together with a screenshot from a digital interface (mobile, desktop, or web). The ground-truth target is given as a bounding box, and we measure success by whether the model's predicted click coordinate falls inside the box. ScreenSpot-V2 corrects labeling errors in the original ScreenSpot benchmark and contains 1,269 tasks, with most screenshots below 2560×1440 resolution. In contrast, ScreenSpot-Pro includes 1,555 tasks and features substantially higher-resolution screenshots (up to 5120×2880), resulting in denser visual content and more fine-grained targets. Overall, ScreenSpot-V2 reflects common UI settings, while ScreenSpot-Pro stresses precise grounding in high-resolution, information-rich interfaces.

D.3 Offline GUI Navigation Benchmarks

Multimodal-Mind2Web-v2 We build our benchmark on *Multimodal-Mind2Web* (MM-Mind2Web) [56], the multimodal extension of Mind2Web [9], to evaluate offline web navigation on realistic user tasks. MM-Mind2Web aligns each step in a human demonstration with a webpage screenshot (and the corresponding HTML/DOM state), forming a golden multi-step trajectory conditioned on a high-level natural-language instruction [9, 56]. The test split spans 100+ websites; all webpages along the golden

trajectories are cached to support fully offline evaluation, and tasks are crowdsourced to reflect real user intents. As shown in Figure 9, MM-Mind2Web represents action history as symbolic records that are neither natural-language descriptions nor aligned with real user interaction. To address this, we use Qwen3-VL-32B-Instruct to rewrite each action into a natural-language description and use these descriptions as the history context. The resulting dataset, *Multimodal-Mind2Web-v2* (MM-Mind2Web-v2), contains three subsets, Cross-Task, Cross-Website, and Cross-Domain, with 1,328, 1,019, and 1,002 samples, respectively. We report **step success rate** as the primary metric, which requires both correct target grounding (*element accuracy*) and correct operation execution. Operation correctness is measured by an exact-match F1 score (F1$=1$) over the serialized action string "`ActionType Value`", where `Value` can be the typed text for `Write` actions or the app/website identifier for `OpenApp` actions.

AndroidControl-v2 *AndroidControl-v2* is based on AndroidControl [20], an offline Android GUI navigation benchmark that pairs step-wise instructions with mobile screenshots and demonstrated actions. However, AndroidControl contains non-trivial annotation noise (about 20% errors in action types and/or coordinates), as illustrated in Figure 9. To improve evaluation reliability, we use Qwen3-VL-32B-Instruct to filter mismatched samples by checking the consistency between each demonstrated action and its oracle low-level step instruction. We start from a sampled set of 500 examples from UGround [13] and obtain a cleaned subset of **398 examples** after filtering. We evaluate under both high-level and low-level instructions and report step accuracy, where a step is counted as successful only if the predicted action type, textual value (when applicable), and target coordinates are all correct. **Note that prior work may use inconsistent evaluation protocols for this benchmark.** For example, OS-Atlas [43] and GUI-R1 [28] treat grounding as a distance-to-target threshold, which can be misleading in practice because UI elements vary greatly in size (so a fixed threshold is not comparable across screens). Instead, we follow UGround' strategy [13]: we use the accessibility tree to map the predicted coordinate to its nearest UI element, and then match that element against the ground-truth target.

D.4 Online GUI Navigation Benchmarks

AndroidWorld We evaluate online mobile agent performance on *AndroidWorld* [34], which runs interactive tasks in Android emulators and scores agents by whether they reach the correct final device states. AndroidWorld contains 116 tasks across 20 real-world apps and covers diverse multi-step workflows (e.g., search, form filling, and settings changes) under realistic UI dynamics. With the official Docker environment, we found that Task #82 (SIMPLESMSREPLYMOSTRECENT) cannot be initialized, so we report results on the remaining 115 tasks. Evaluation uses the benchmark's rule-based completion checker with a maximum horizon of 20 steps. To assess self-verification, we count a task as successful only when (i) the agent explicitly outputs a `Terminate` action and (ii) the environment state satisfies the completion rules. Our agent follows the See-Act-V framework following UGround [13], but **we remove the step-wise reflection and summary module. This design choice isolates the *native* capability of the underlying model, rather than relying on a hand-crafted control structure.** For completeness, we also report baseline results with the summary modules, and show that our native model (without such modules) can surpass agents that depend on these additional components, highlighting the potential of our recipe to reduce human-designed scaffolding.

WebArena-Lite-v2 We evaluate online web agent performance on *WebArena-Lite-v2* [24], a locally deployed website environments upgraded from *WebArena-Lite* [23] consisting of 154 tasks. To setup the website environment, we follow the instruction by Maxime Gasse[4] for a more stable Map setup[5]. Building upon the official WebArena-Lite-v2 implementation, we further improve the robustness of action parsing and execution by: (i) parsing the "response" action as "answer" rather than treating it as illegal; (ii) appending an automated "terminate" action after "action" and "response"; (iii) supporting multi-line answer strings (separated by `\n`) instead of only the first line; (iv) clearing blank content before typing predicted messages. Moreover, we replace `gpt-4o-2024-11-20` with `gpt-5` for more accurate LLM-based fuzzy evaluation as we observed false positives when using GPT-4o. Given

[4] https://github.com/gasse/webarena-setup/tree/main/webarena

[5] We found the container for Map website is brittle when following the official WebArena-Lite-v2 setup: https://github.com/OpenGVLab/ScaleCUA/tree/main/evaluation/WebArenaLiteV2.

Table 10: Step accuracy on AC-v2 (High/Low Level) and MM-M2W-v2 (Average).

Model	AC-v2 (High)		AC-v2 (Low)		MM-M2W-v2 (Avg)	
	Pass@1	Pass@4	Pass@1	Pass@4	Pass@1	Pass@4
Proprietary Models with SeeAct-V Framework						
GPT-4o + UGround-v1-7B	57.0	66.3	78.4	85.4	36.2	39.6
GPT-4.1 + UGround-v1-7B	57.5	63.3	78.4	83.2	40.1	43.6
GPT-5-mini + UGround-v1-7B	52.8	58.8	77.1	83.2	43.5	46.7
GPT-5 + UGround-v1-7B	61.3	69.4	86.2	90.0	47.0	50.3
Open-source Native Models						
GUI-R1-3B	40.0	54.0	55.8	71.9	23.6	38.2
GUI-R1-7B	39.7	56.3	62.3	72.6	36.9	49.0
Aguvis-7B	37.7	43.7	48.0	48.7	35.4	44.9
UI-TARS-1.5-7B	45.2	63.1	48.5	70.9	34.7	46.0
GLM-4.1V-9B-Thinking	37.2	49.0	67.1	73.4	26.2	32.5
Qwen2.5-VL-32B	49.0	66.6	78.4	85.4	44.9	56.5
Qwen2.5-VL-72B	56.5	**72.9**	82.9	90.2	48.0	**57.6**
Qwen3-VL-32B	58.8	69.6	83.9	86.9	47.6	57.2
Qwen2.5-VL-3B (Baseline)	36.4	50.8	71.1	79.2	23.4	28.3
GUI-Libra-3B (Ours)	57.3 (+20.9)	67.1 (+16.3)	85.9 (+14.8)	90.5 (+11.3)	42.7 (+19.3)	50.3 (+22.0)
Qwen2.5-VL-7B (Baseline)	46.5	58.5	67.8	81.7	32.5	45.2
GUI-Libra-7B (Ours)	59.3 (+12.8)	67.3 (+8.8)	85.2 (+17.4)	90.7 (+9.0)	46.5 (+14.0)	53.6 (+8.4)
Qwen3-VL-4B (Baseline)	49.3	63.3	78.9	82.4	41.2	51.3
GUI-Libra-4B (Ours)	62.3 (+13.0)	68.6 (+5.3)	86.4 (+7.5)	**93.0** (+10.6)	50.0 (+8.8)	55.6 (+4.3)
Qwen3-VL-8B (Baseline)	54.8	66.1	77.6	83.2	43.8	52.8
GUI-Libra-8B (Ours)	**64.3** (+9.5)	70.6 (+4.5)	**88.9** (+11.3)	91.7 (+8.5)	**50.5** (+6.7)	55.2 (+2.4)

the high variance in results, we report the average across four runs for all experiments, with the temperature set to 0.0 and top_p to 1.0.

Online-Mind2Web We also include Online-Mind2Web benchmark [46] spanning 136 live real-world websites and covering 300 web agent tasks. We set the maximum interaction steps to 30 for each task. We adopt the proposed WebJudge method backed by either o4-mini [6] or WebJudge-7B [7] models. The LLM-based judge first identifies key points of the task, then selects task-relevant key screenshots from each step. Finally, the judge is provided with task description, agent textual actions, task completion key points and selected key screenshots to make a binary outcome judgment indicating whether all key points are satisfied.

E Additional Results

This section presents additional results to further clarify our approach. We include an auxiliary study on grounding as a single-step verifiable case in Appendix E.3 to contrast with multi-step navigation under partial verifiability, a controlled comparison of reasoning-augmentation models in Appendix E.4, as well as complementary offline metrics (grounding accuracy, action-type accuracy or operation F1) in Appendix E.9.

E.1 Offline Benchmark Results

The results are in Table 10.

Table 11: Effect of mixing grounding data into RL training on ScreenSpot-v2 (SS-v2), ScreenSpot-Pro (SS-Pro), MM-Mind2Web-v2, and AndroidControl-v2 (AC-v2). We report Pass@1 on all benchmarks.

	SS-v2	SS-Pro	MM-Mind2Web-v2	AC-v2 (high)	AC-v2 (low)
Qwen3-VL-4B	91.7	52.8	41.2	49.3	78.9
GUI-Libra-4B	92.3	54.3	49.1	59.8	87.7
GUI-Libra-4B + Mix Grounding 20k	94.6 ↑ (+2.3)	61.4 ↑ (+7.1)	43.9 ↓ (-5.2)	61.3 ↑ (+1.5)	84.9 ↓ (-2.8)
Qwen3-VL-8B	92.1	52.7	43.8	54.8	77.6
GUI-Libra-8B	90.7	54.1	50.3	65.6	88.7
GUI-Libra-8B + Mix Grounding 20k	94.8 ↑ (+4.1)	59.9 ↑ (+5.8)	49.5 ↓ (-0.8)	61.7 ↓ (-3.9)	86.4 ↓ (-2.3)

[6] https://developers.openai.com/api/docs/models/o4-mini
[7] https://huggingface.co/osunlp/WebJudge-7B

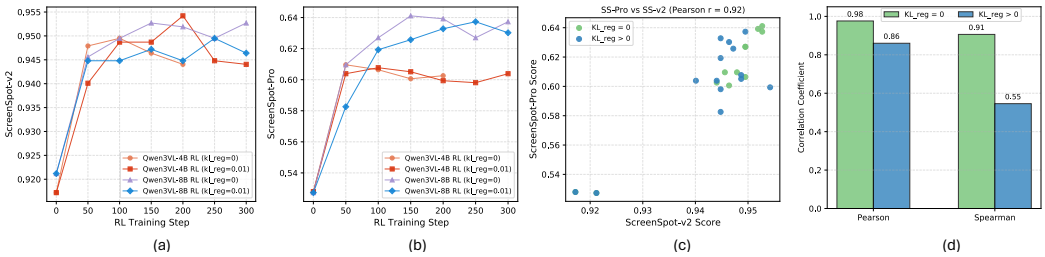


Figure 10: RL for grounding exhibits stable improvements and strong cross-benchmark predictability. (a) ScreenSpot-V2 and (b) ScreenSpot-Pro performance over RL training. (c) Correlation between the two benchmark scores across checkpoints. (d) Pearson and Spearman correlations with and without KL regularization.

E.2 RL with Mixed Navigation and Grounding Data

Prior work [49, 28, 27] adds direct grounding supervision during RL to improve localization. We test this by mixing 20K grounding examples from [49] with our 40K navigation RL set, converting them to our unified action format and keeping the same reward computation. Table 11 shows a clear trade-off: grounding accuracy on ScreenSpot-v2 and ScreenSpot-Pro improves by 2–7 points, but navigation performance generally declines. Thus, **adding direct grounding supervision strengthens spatial alignment but can hurt reasoning-intensive navigation**.

E.3 Grounding as a Single-step Verifiable Setting

Grounding provides a near-ideal *single-step verifiable* setting: each example typically refers to a specific UI element, and we can directly verify correctness by checking whether the predicted coordinate falls inside the annotated bounding box. Since the agent produces only a single action, this setting isolates the single-step, verifiable regime discussed in Section 5.3. To study this regime, we train Qwen3-VL-4B and Qwen3-VL-8B with GRPO on a 40K downsampled grounding dataset from GTA1 [49]. Results are shown in Figure 10.

Unlike navigation, grounding does not exhibit an offline–online evaluation gap; therefore, we assess *predictability* using two grounding benchmarks instead: ScreenSpot-v2 and ScreenSpot-Pro. ScreenSpot-v2 is closer to our training distribution (similar image resolution), while ScreenSpot-Pro contains substantially higher-resolution screenshots, making it a useful test of distribution shift. Figures 10(a)–(b) show that performance on both benchmarks improves steadily and then plateaus after roughly 200 RL steps, without the significant reward-hacking-style drops observed in multi-step navigation. Moreover, the two benchmarks are strongly correlated: Figure 10(c) shows a tight relationship between ScreenSpot-v2 and ScreenSpot-Pro scores, and Figure 10(d) reports very high Pearson and Spearman correlations. Interestingly, removing KL regularization yields even higher correlation (Pearson ≈ 0.98), consistent with our analysis that in single-step, verifiable tasks, RLVR-style training can remain stable and highly predictable even without KL regularization.

E.4 Comparing Different Models for Reasoning Augmentation

In our reasoning augmentation pipeline, we use GPT-4.1 to generate reasoning traces. This choice is motivated by preliminary experiments showing that GPT-4.1 can be prompted to produce richer, more informative rationales for VLM training. With the same prompt, GPT-4o often generates shorter reasoning, while reasoning models such as o4-mini typically provide limited visible traces and hide their true reasoning traces, resulting in similarly short rationales. In contrast, GPT-4.1 more reliably produces detailed reasoning that provide more useful thought and better aligns with the actions.

We further quantify this effect with a controlled comparison. Using the same 30K web samples from AGUVIS, we generate reasoning traces with each model under an identical prompt and fine-tune the same Qwen2.5-VL-3B-Instruct base model on the resulting augmented data. To isolate the impact of reasoning quality, we extract action targets from the model outputs and use the same UGround-7B-v1 [13] model for coordinate prediction across all settings. Figure 11 reports results on the original MM-Mind2Web Cross-Website subset: GPT-4.1 yields the best performance, outperforming GPT-4o

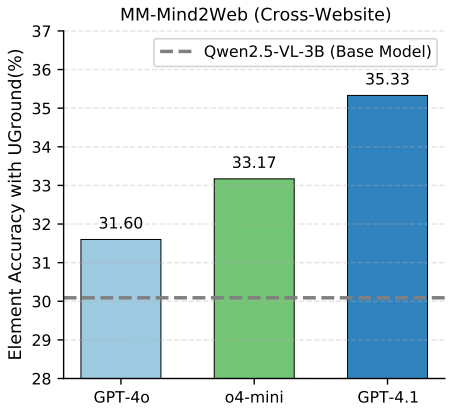


Figure 11: Performance comparison of different models for reasoning generation. All models are used to augment the same 30K web samples from AGUVIS and are fine-tuned on the same 3B base model. This controlled evaluation reveals substantial performance differences among generator models.

by $+3.7$ and o4-mini by $+2.1$. These results indicate that the choice of reasoning generator is an important factor in effective reasoning augmentation.

E.5 Data Quality, Reasoning, and Teacher Contributions

The data pipeline and the training objective play complementary roles. Teachers generate reasoning and textual action fields, while coordinates originate from the source corpora; model-based agreement filtering and bounding-box annotation further shape the training set. Holding this curated data pool fixed, direct RL without ASFT reaches 61.7 average offline Pass@1 and 20.9 AndroidWorld success, compared with 43.6/3.5 for the base model and 62.7/25.2 for the full recipe. Thus RL recovers most of the improvement without a teacher-generated reasoning SFT initialization. This variant still uses model-assisted data curation and reward annotations.

Removing explicit reasoning throughout training and inference is a different intervention: online success drops from 25.2 to 12.2, while average offline Pass@1 drops from 62.7 to 59.1 (Table 6). Reasoning is therefore particularly valuable for online navigation, where its benefit is less visible in step-wise matching. Generator quality also matters: the controlled 30K comparison in Appendix E.4 finds a $+3.7/+2.1$ advantage for GPT-4.1 over GPT-4o/o4-mini. Filtering independently improves data quality (Appendix B.3.1). Together these studies show that stronger reasoning supervision raises performance, while ASFT and RL determine how effectively the model uses it; no single teacher or data-quality effect explains the full gain.

E.6 Replacing Rule-based Rewards with a Learned Judge

We test a direct alternative to single-demonstration matching: Qwen3-VL-8B-Instruct serves as the reward judge for GRPO training of Qwen3-VL-4B-Instruct. The judge receives the step context, the candidate action, and the oracle action, and returns `true` if the candidate is equivalent to the oracle action and `false` otherwise. We map these outputs to rewards of one and zero. All other training settings are unchanged from the rule-based run; only the reward is replaced. We track judge reward on training responses and rule-based action accuracy on held-out examples, keeping the validation criterion independent of the learned training judge.

Table 12 shows a sharp divergence: judge reward approaches one within 200 steps while held-out accuracy collapses. This behavior is consistent with reward exploitation and shows that a learned judge can introduce an optimization target that departs from action correctness. Rule-based matching leaves valid alternatives uncredited, but offers a transparent target; KL and SNGS control the updates driven by that imperfect supervision. The experiment motivates this design choice without ruling out better-calibrated judges or richer verification schemes.

Table 12: Learned-judge reward and held-out rule-based accuracy during GRPO. Values are fractions in $[0, 1]$.

RL step	Training judge reward	Held-out rule-based accuracy
0	0.22	0.371
20	0.69	0.053
100	0.96	0.005
200	0.99	0.000

E.7 Comparing with Uniform Negative Gradient Scaling Strategy

In our method, we use an adaptive negative gradient scaling method to enable adaptive scaling behaviors. Table 13 compares our adaptive negative gradient scaling strategy with a uniform variant that applies a constant scaling factor λ_g across all states. Uniform scaling yields weaker overall performance: both $\lambda_g = 0.75$ and $\lambda_g = 0.9$ perform worse on MM-Mind2Web-v2, AC-v2 (High-/Low), and AndroidWorld. These results suggest that a single global scaling factor cannot capture the heterogeneous difficulty and ambiguity across states and can lead to suboptimal optimization, whereas adaptive scaling provides the flexibility needed to stabilize training and improve decision making.

Table 13: Ablations study of negative scaling strategy on 3B base model.

Model	**MM-Mind2Web-v2**		**AC-v2 (High)**		**AC-v2 (Low)**		**AndroidWorld**
	Pass@1	**Pass@4**	**Pass@1**	**Pass@4**	**Pass@1**	**Pass@4**	
GUI-Libra ($\lambda_g = 0.75$)	42.2	48.5	54.5	64.1	86.7	90.2	19.1
GUI-Libra ($\lambda_g = 0.9$)	**42.7**	48.1	52.3	61.1	87.4	90.7	20.0
GUI-Libra	**42.7**	**50.2**	**55.8**	**65.8**	**89.5**	**91.5**	**25.2**

E.8 Additional Results on Online WebArena-Lite-v2

On **WebArena-Lite-v2** (Table 14), a **locally deployed web benchmark (rather than live websites)**, GUI-Libra shows strong generalization across diverse web tasks despite being trained on only 15K web-related samples, far fewer than the web corpora used by many existing GUI models. Even in this low-data regime, GUI-Libra delivers large gains over its base models: GUI-Libra-7B improves the average success rate from 4.9 to 22.6, and GUI-Libra-8B increases performance from 15.3 to 26.6. These results are competitive with strong proprietary systems such as GPT-4o equipped with UI-TARS and UGround. Moreover, GUI-Libra-8B outperforms large-scale models including ScaleCUA-32B, UI-TARS-72B, and Aguvis-72B, all trained on substantially larger web datasets.

E.9 Additional Metrics on Offline Benchmarks

In our main offline evaluations, we use *step accuracy* as the primary metric. To better understand *where* the gains come from, we also report decomposed metrics that separate grounding from action prediction. Specifically, we report (i) **grounding accuracy** (whether the predicted coordinate lies inside the target element) and **action-type accuracy** on AndroidControl-v2, and (ii) **grounding accuracy** and **operation F1** on MM-Mind2Web-v2. Operation F1 is computed following the protocol in Appendix D.3.

AndroidControl-v2. Tables 16 and 17 show that GUI-Libra remains strong under these more fine-grained metrics. Notably, GUI-Libra-8B achieves the best Pass@1 grounding accuracy on both high-level and low-level tasks, reaching 76.3 and 95.5 Pass@1 on the high-level and low-level settings, respectively. It also outperforms GPT-5 + UGround-7B-v1 and larger open-weight models (e.g., 32B and 72B). The overall trend for action-type accuracy is similar, but the gaps to strong baselines are smaller, suggesting that our improvements are driven primarily by better grounding rather than action-type prediction. This is expected: action types are often inferred reliably from language and prior knowledge, whereas accurate grounding, especially for high-level, goal-directed tasks, is more challenging for GUI tasks.

Table 14: Performance comparison on **WebArena-Lite-v2** in 15 steps. * denotes numbers reported in [24].

	GitLab	MAP	Reddit	Shopping	ShoppingAdmin	Average
Native Models						
Aguvis-72B*	-	-	-	-	-	5.8
Qwen2.5-VL-72B*	-	-	-	-	-	15.6
InternVL3.5-241B-A28B*	-	-	-	-	-	11.7
UI-TARS-1.5-7B*	-	-	-	-	-	20.8
UI-TARS-72B-DPO*	-	-	-	-	-	23.4
ScaleCUA-3B	21.7	7.7	13.2	16.5	23.6	17.2
ScaleCUA-7B	28.3	15.4	27.6	18.8	30.7	23.9
ScaleCUA-32B	34.2	10.6	26.3	16.5	33.6	24.0
Qwen2.5-VL-3B	1.7	0.0	0.0	1.7	0.0	0.8
GUI-Libra-3B (Ours)	25.8	9.6	18.4	17.6	12.1	16.7
Qwen2.5-VL-7B	8.3	1.0	2.6	7.4	2.9	4.9
GUI-Libra-7B (Ours)	25.0	10.6	26.3	26.1	22.9	22.6
Qwen3-VL-4B	17.5	5.8	10.5	13.1	10.7	11.9
GUI-Libra-4B (Ours)	29.2	10.6	34.2	30.1	17.9	24.4
Qwen3-VL-8B	15.0	5.8	17.1	17.0	19.3	15.3
GUI-Libra-8B (Ours)	31.7	17.3	35.5	26.1	25.0	**26.6**
Agent Framework (GPT-4o as the Planner)						
GPT-4o + UI-TARS-1.5-7B [33]*	-	-	-	-	-	22.6
GPT-4o + UGround-V1-7B [13]*	-	-	-	-	-	23.2
GPT-4o + ScaleCUA-7B [24]*	-	-	-	-	-	**28.6**

Table 15: Comparison between GUI-Libra w/ SNGS and w/o SNGS across benchmarks.

Model	MM-Mind2Web-v2		AC-v2 (High)		AC-v2 (Low)		AndroidWorld	WebArena-Lite-v2
	Pass@1	Pass@4	Pass@1	Pass@4	Pass@1	Pass@4		
GUI-Libra-4B (w/o SNGS)	49.1	55.1	59.8	69.9	87.7	92.0	39.1	22.2
GUI-Libra-4B (w/ SNGS)	50.0	55.6	62.3	68.6	86.4	93.0	42.6	24.4

MM-Mind2Web-v2. Tables 18 and 19 report grounding accuracy and operation F1, respectively. For grounding, GUI-Libra-8B achieves the highest Pass@1 across all three subsets, with an average Pass@1 of 57.8, surpassing strong baselines including Qwen3-VL-32B, Qwen2.5-VL-72B, and GPT-5 with UGround. For Pass@4, Qwen2.5-VL-72B is 1.4 points higher than GUI-Libra-8B, indicating that web-domain evaluation can still benefit from additional model capacity and/or web-specific training data. We attribute this gap mainly to domain imbalance in our training set: roughly 85% of our SFT data comes from mobile, with only 15% from web. We expect that scaling high-quality web data would further improve Pass@4, consistent with our strong Pass@1/Pass@4 results on AndroidControl-v2. For operation F1, we report `Best@N` (the maximum over $N \in \{1, 4\}$ samples), since F1 is a continuous score in $[0, 1]$. While larger models achieve strong `Best@4` results, GUI-Libra-7B attains the best `Best@1` performance (85.1), and GUI-Libra-3B remains competitive with 32B–72B baselines.

Overall, these additional metrics show that GUI-Libra improves not only step accuracy but also its underlying components, providing a more complete view of the sources of improvement.

F Proofs for Theoretical Analysis

This section provides detailed proofs for the theoretical results in Sec. 5.3.

For completeness, we formalize GUI navigation as a goal-conditioned multi-step decision process with valid-action set $\mathcal{A}^*(s)$ at each state s, where offline supervision provides a single demonstrated action $\tilde{a}(s) \in \mathcal{A}^*(s)$. The offline reward is $\tilde{r}(s, a) = \mathbf{1}\{a = \tilde{a}(s)\}$, the offline matching score is $M_{\text{off}}(\pi) = \mathbb{E}_{s\sim d_\mu}[\pi(\tilde{a}(s) \mid s)]$, and the online objective is the task success rate $J(\pi)$. We measure distribution shift by $C(\pi) \triangleq \max_{t\in[H]} \sup_{s:\, d_\mu(s)>0} \frac{d_{\pi,t}(s)}{d_\mu(s)}$, and define the off-demo validity mass as $\eta_\pi(s) \triangleq \pi(\mathcal{A}^*(s) \setminus \{\tilde{a}(s)\} \mid s)$ with $\bar{\eta}_\pi = \mathbb{E}_{s\sim d_\mu}[\eta_\pi(s)]$.

Table 16: Grounding Accuracy on AndroidControl-v2.

Model	High Level		Low Level	
	Pass@1	Pass@4	Pass@1	Pass@4
Proprietary Models with SeeAct-V Framework				
GPT-4o + UGround-v1-7B	58.7	67.3	83.4	88.3
GPT-4.1 + UGround-v1-7B	60.5	64.6	83.0	86.1
GPT-5-mini + UGround-v1-7B	61.9	66.8	83.9	87.9
GPT-5 + UGround-v1-7B	74.4	**82.1**	93.7	94.6
Open-source Native Models				
GUI-R1-3B	53.4	66.4	81.2	87.0
GUI-R1-7B	58.3	72.2	87.9	87.9
Aguvis-7B	67.3	78.0	85.7	87.0
UI-TARS-1.5-7B	57.9	71.3	78.5	88.8
GLM-4.1V-9B-Thinking	44.0	55.6	68.2	75.8
Qwen2.5-VL-32B	49.0	66.6	78.4	85.4
Qwen2.5-VL-72B	56.5	72.9	82.9	90.2
Qwen3-VL-32B	70.9	79.4	91.9	93.3
Qwen2.5-VL-3B (Baseline)	42.2	46.6	78.0	80.7
GUI-Libra-3B (Ours)	68.2	77.1	89.2	91.5
Qwen2.5-VL-7B (Baseline)	60.5	69.5	82.5	88.3
GUI-Libra-7B (Ours)	71.8	78.0	91.5	93.7
Qwen3-VL-4B (Baseline)	65.9	76.7	90.6	93.3
GUI-Libra-4B (Ours)	74.0	81.6	91.0	95.1
Qwen3-VL-8B (Baseline)	67.7	79.8	93.3	95.1
GUI-Libra-8B (Ours)	**76.2**	81.6	**95.5**	**96.4**

Table 17: Type Accuracy on AndroidControl-v2.

Model	High Level		Low Level	
	Pass@1	Pass@4	Pass@1	Pass@4
Proprietary Models with SeeAct-V Framework				
GPT-4o + UGround-v1-7B	73.9	82.7	89.2	93.7
GPT-4.1 + UGround-v1-7B	72.9	78.9	89.5	93.0
GPT-5-mini + UGround-v1-7B	70.9	77.4	89.7	93.5
GPT-5 + UGround-v1-7B	**74.1**	78.9	88.4	91.5
Open-source Native Models				
GUI-R1-3B	63.1	76.4	71.1	90.2
GUI-R1-7B	61.8	76.1	76.1	89.2
Aguvis-7B	58.0	59.8	60.6	62.1
UI-TARS-1.5-7B	69.4	**87.4**	70.9	91.0
GLM-4.1V-9B-Thinking	65.8	74.1	89.2	92.5
Qwen2.5-VL-32B	67.3	78.1	85.7	89.7
Qwen2.5-VL-72B	72.9	86.9	89.5	**96.2**
Qwen3-VL-32B	73.6	81.2	90.2	92.2
Qwen2.5-VL-3B (Baseline)	53.0	77.4	80.2	90.0
GUI-Libra-3B (Ours)	70.4	78.1	89.5	93.7
Qwen2.5-VL-7B (Baseline)	62.8	77.4	78.6	91.7
GUI-Libra-7B (Ours)	72.9	78.6	88.9	91.2
Qwen3-VL-4B (Baseline)	63.6	78.4	86.4	88.7
GUI-Libra-4B (Ours)	72.6	76.1	92.5	95.0
Qwen3-VL-8B (Baseline)	68.8	77.6	82.4	86.9
GUI-Libra-8B (Ours)	**74.1**	78.9	**93.7**	95.5

Table 18: Grounding accuracy (%) on Multimodal-Mind2Web-v2.

Model	Cross-Task		Cross-Website		Cross-Domain		Average	
	Pass@1	Pass@4	Pass@1	Pass@4	Pass@1	Pass@4	Pass@1	Pass@4
Proprietary Models with SeeAct-V Framework								
GPT-4o + UGround-v1-7B	42.0	44.0	40.6	42.9	44.9	47.2	42.5	44.7
GPT-4.1 + UGround-v1-7B	47.4	49.3	43.5	45.5	48.7	51.4	46.5	48.7
GPT-5-mini + UGround-v1-7B	52.5	53.1	50.7	51.6	53.6	54.3	52.3	53.0
GPT-5 + UGround-v1-7B	55.3	56.6	53.1	53.4	55.8	57.3	54.7	55.8
Open-source Native Models								
GUI-R1-3B	29.1	44.6	27.2	45.5	29.1	44.9	28.5	45.0
GUI-R1-7B	43.8	56.8	40.6	54.9	45.7	56.3	43.4	56.0
Aguvis-7B	40.5	53.5	34.9	47.4	38.8	48.8	38.1	49.9
UI-TARS-1.5-7B	46.2	59.0	42.2	57.6	46.4	60.2	45.0	58.9
GLM-4.1V-9B-Thinking	36.8	42.8	34.2	39.7	36.9	44.1	36.0	42.2
Qwen2.5-VL-32B	53.0	61.8	50.9	61.6	52.6	62.9	52.2	62.1
Qwen2.5-VL-72B	55.7	**65.0**	53.7	61.5	56.3	**64.4**	55.2	**63.6**
Qwen3-VL-32B	55.9	62.4	52.7	**61.7**	56.7	63.9	55.1	62.7
Qwen2.5-VL-3B (Baseline)	37.9	40.1	35.6	38.6	38.8	43.2	37.4	40.6
GUI-Libra-3B (Ours)	50.1	57.8	50.7	58.6	51.6	58.5	50.8	58.3
Qwen2.5-VL-7B (Baseline)	44.1	54.0	45.5	54.2	46.2	57.4	45.3	55.2
GUI-Libra-7B (Ours)	53.5	59.8	54.2	60.4	53.6	60.7	53.7	60.3
Qwen3-VL-4B (Baseline)	49.9	59.6	47.9	58.2	49.9	57.5	49.2	58.4
GUI-Libra-4B (Ours)	57.2	61.7	56.2	61.4	56.7	62.1	56.7	61.7
Qwen3-VL-8B (Baseline)	52.6	60.0	49.6	59.3	52.4	59.3	51.5	59.5
GUI-Libra-8B (Ours)	**58.6**	62.6	**56.5**	61.2	**58.2**	62.5	**57.8**	62.1

Table 19: Operation F1 score on Multimodal-Mind2Web-v2.

Model	Cross-Task		Cross-Website		Cross-Domain		Average	
	Best@1	Best@4	Best@1	Best@4	Best@1	Best@4	Best@1	Best@4
Proprietary Models with SeeAct-V Framework								
GPT-4o + UGround-v1-7B	70.1	78.9	70.9	79.6	67.7	75.7	69.6	78.1
GPT-4.1 + UGround-v1-7B	72.7	81.6	71.1	81.0	71.1	79.4	71.6	80.7
GPT-5-mini + UGround-v1-7B	80.7	88.8	77.3	86.8	77.9	85.3	78.6	87.0
GPT-5 + UGround-v1-7B	79.7	89.2	79.6	88.9	79.1	88.2	79.5	88.8
Open-source Native Models								
GUI-R1-3B	79.5	87.9	75.4	85.3	79.6	89.4	78.1	87.5
GUI-R1-7B	78.9	89.6	75.2	88.5	79.8	90.7	78.0	89.6
Aguvis-7B	84.4	91.1	80.7	87.9	83.3	91.5	82.8	90.2
UI-TARS-1.5-7B	81.2	85.3	78.5	81.6	81.1	85.0	80.3	84.0
GLM-4.1V-9B-Thinking	73.6	82.0	69.2	80.0	73.7	81.6	72.2	81.2
Qwen2.5-VL-32B	84.2	92.6	81.9	**92.0**	82.6	**93.0**	82.9	92.5
Qwen2.5-VL-72B	85.2	**93.5**	83.3	91.5	84.3	**93.0**	84.3	**92.7**
Qwen3-VL-32B	83.9	91.6	81.3	89.9	83.7	92.2	83.0	91.2
Qwen2.5-VL-3B (Baseline)	59.3	86.5	51.6	83.0	64.7	88.3	58.5	85.9
GUI-Libra-3B (Ours)	84.6	88.0	80.8	85.2	**86.4**	89.2	83.9	87.5
Qwen2.5-VL-7B (Baseline)	68.1	90.9	66.7	90.4	73.3	91.6	69.4	91.0
GUI-Libra-7B (Ours)	85.2	88.1	**84.3**	88.2	85.9	89.4	**85.1**	88.6
Qwen3-VL-4B (Baseline)	81.6	88.9	78.9	88.2	79.5	89.9	80.0	89.0
GUI-Libra-4B (Ours)	85.0	88.9	83.6	88.2	82.8	88.1	83.8	88.4
Qwen3-VL-8B (Baseline)	83.2	91.2	80.0	89.0	82.7	90.2	82.0	90.1
GUI-Libra-8B (Ours)	**85.5**	88.1	81.8	86.9	84.4	87.0	83.9	87.3

Behavior policy and data distribution. Here μ is the data-generating behavior policy, or a mixture of policies for a heterogeneous corpus, and $d_\mu = H^{-1} \sum_{t=1}^{H} d_{\mu,t}$ is its population occupancy distribution. We use a fixed horizon with absorbing terminal states and include the goal and history in the state. Training draws contexts from the fixed offline dataset and candidate actions from the current policy; neither knowledge of $\mu(a \mid s)$ nor additional samples from μ are needed. The reference policy π_{ref} is a separately trained initialization, not μ. The verified action map $\tilde{a}(s)$ can differ from the original behavior policy's choices after curation. Valid demonstrations and the failure condition below are modeling assumptions; they do not assert that the behavior policy maximizes return. The bound concerns task success, rather than regret relative to an optimal policy.

F.1 When does offline step-wise matching predict online success?

Theorem F.1: Offline-to-online bound under partial verifiability (Restatement of Theorem 5.1)

Assume that any failed episode contains at least one invalid action, and that, for all $t \in [H]$, $d_{\pi,t}(s) > 0$ implies $d_\mu(s) > 0$ (i.e., $\mathrm{supp}(d_{\pi,t}) \subseteq \mathrm{supp}(d_\mu)$). This condition ensures the occupancy ratio $C(\pi)$ is well-defined. Then the online success probability satisfies

$$J(\pi) \;\geq\; 1 \;-\; H \cdot C(\pi) \cdot \Big(1 - M_{\mathrm{off}}(\pi) - \bar{\eta}_\pi\Big). \tag{4}$$

If $C(\pi) \leq C_0$ and $\bar{\eta}_\pi = \eta_0$ across the policies considered, Eq. 4 gives $J(\pi) \geq 1 - HC_0(1 - M_{\mathrm{off}}(\pi) - \eta_0)$. Holding C_0 and η_0 fixed, this common affine lower bound increases with offline matching. Thus coverage and ambiguity must be controlled alongside the matching score; the actual success probabilities need not follow the same ordering.

Proof. Let E_t denote the event that the action at step t is invalid:

$$E_t \;\triangleq\; \{a_t \notin \mathcal{A}^*(s_t)\}.$$

By assumption, failure implies that at least one invalid step occurs, i.e.,

$$\{\text{failure}\} \subseteq \bigcup_{t=1}^{H} E_t.$$

Therefore,

$$1 - J(\pi) = \mathbb{P}(\text{failure}) \leq \mathbb{P}\left(\bigcup_{t=1}^{H} E_t\right) \leq \sum_{t=1}^{H} \mathbb{P}(E_t),$$

where the last inequality is the union bound.

Fix any $t \in [H]$. Conditioning on s_t gives

$$\begin{aligned}\mathbb{P}(E_t) &= \mathbb{E}_{s\sim d_{\pi,t}}\Big[\mathbb{P}(a_t \notin \mathcal{A}^*(s) \mid s_t = s)\Big] \\ &= \mathbb{E}_{s\sim d_{\pi,t}}\Big[1 - \pi(\mathcal{A}^*(s) \mid s)\Big].\end{aligned}$$

Define the nonnegative function $f(s) \triangleq 1 - \pi(\mathcal{A}^*(s) \mid s) \geq 0$. By the definition of $C(\pi)$ and the support condition $d_{\pi,t} \ll d_\mu$,

$$\begin{aligned}\mathbb{E}_{s\sim d_{\pi,t}}[f(s)] &= \mathbb{E}_{s\sim d_\mu}\left[\frac{d_{\pi,t}(s)}{d_\mu(s)}\, f(s)\right] \\ &\leq C(\pi) \cdot \mathbb{E}_{s\sim d_\mu}[f(s)].\end{aligned}$$

Next, since the offline dataset verifies only the demonstrated action $\tilde{a}(s) \in \mathcal{A}^*(s)$, we can decompose the true step-wise validity probability as

$$\pi(\mathcal{A}^*(s) \mid s) = \pi(\tilde{a}(s) \mid s) \;+\; \pi(\mathcal{A}^*(s) \setminus \{\tilde{a}(s)\} \mid s) = \pi(\tilde{a}(s) \mid s) \;+\; \eta_\pi(s).$$

Taking expectation over $s \sim d_\mu$ yields

$$\mathbb{E}_{s\sim d_\mu}\big[\pi(\mathcal{A}^*(s) \mid s)\big] = M_{\mathrm{off}}(\pi) \;+\; \bar{\eta}_\pi,$$

and hence

$$\mathbb{E}_{s\sim d_\mu}[f(s)] = 1 - \mathbb{E}_{s\sim d_\mu}\big[\pi(\mathcal{A}^*(s) \mid s)\big] = 1 - M_{\mathrm{off}}(\pi) - \bar{\eta}_\pi.$$

Combining the above bounds, for each t we have

$$\mathbb{P}(E_t) \leq C(\pi) \cdot \big(1 - M_{\mathrm{off}}(\pi) - \bar{\eta}_\pi\big).$$

Summing over $t = 1, \ldots, H$ and using $1 - J(\pi) \leq \sum_{t=1}^{H} \mathbb{P}(E_t)$ gives

$$1 - J(\pi) \leq H \cdot C(\pi) \cdot \big(1 - M_{\mathrm{off}}(\pi) - \bar{\eta}_\pi\big),$$

which is equivalent to Eq 4. This concludes the proof. □

Discussion. Theorem F.1 (Theorem 5.1) clarifies why an offline one-step matching score can be an unreliable proxy for the online success probability $J(\pi)$ in multi-step GUI navigation under partial verifiability. Offline evaluation is computed on a fixed dataset distribution d_μ and credits only the single demonstrated action $\tilde{a}(s)$. In contrast, online success depends on the policy-induced state distributions $\{d_{\pi,t}\}_{t=1}^{H}$ and on choosing *any* valid action in $\mathcal{A}^*(s)$ over a long horizon. The theorem makes this mismatch explicit through two quantities: the occupancy mismatch $C(\pi)$ and the unobserved off-demo validity mass $\bar{\eta}_\pi$. As a result, offline-to-online predictability can break down in several ways.

(i) Distribution shift and error accumulation (large $C(\pi)$). Even if π matches the demonstrator well on states drawn from d_μ, small errors can compound over time and shift the online trajectory distribution away from the offline support. When $C(\pi)$ is large, $M_{\text{off}}(\pi)$ provides limited information about the states that dominate online performance. In this regime, $M_{\text{off}}(\pi)$ may improve while $J(\pi)$ stagnates (or decreases) because failures are driven by states that are rarely or never seen in the offline data.

(ii) Non-identifiability under partial verifiability (unstable $\bar{\eta}_\pi$). Offline matching measures only $\pi(\tilde{a}(s) \mid s)$, whereas the true one-step validity is

$$\pi(\mathcal{A}^*(s) \mid s) = \pi(\tilde{a}(s) \mid s) + \eta_\pi(s),$$
$$\eta_\pi(s) = \pi(\mathcal{A}^*(s) \setminus \{\tilde{a}(s)\} \mid s)\,.$$

Therefore, $M_{\text{off}}(\pi)$ does not determine true step validity unless the off-demo validity mass $\eta_\pi(s)$ is negligible or approximately invariant across the policies being compared. Importantly, a *larger* $\bar{\eta}_\pi$ does *not* imply better predictability: the issue is that η_π is unobserved under offline verification and can vary substantially across policies, so changes in $M_{\text{off}}(\pi)$ may reflect reallocation of probability mass rather than genuine improvements in correctness.

Example F.1: Larger η_π does not mean better predictability

Consider a single decision state s (so $C(\pi) = 1$) with three actions: a demonstrated valid action $a^\star = \tilde{a}(s)$, an alternative valid action a', and an invalid action a_{bad}. Thus $\mathcal{A}^*(s) = \{a^\star, a'\}$ and choosing any action outside $\mathcal{A}^*(s)$ causes failure. Offline matching credits only $a^\star$, so

$$M_{\text{off}}(\pi) = \pi(a^\star \mid s),$$
$$\eta_\pi(s) = \pi(a' \mid s),$$
$$J(\pi) = \pi(a^\star \mid s) + \pi(a' \mid s) = 1 - \pi(a_{\text{bad}} \mid s).$$

Compare two policies:

$$\pi_1(a^\star \mid s) = 0.2,\ \pi_1(a' \mid s) = 0.7,\ \pi_1(a_{\text{bad}} \mid s) = 0.1$$
$$\Rightarrow \quad M_{\text{off}}(\pi_1) = 0.2,\ \eta_{\pi_1}(s) = 0.7,\ J(\pi_1) = 0.9,$$

$$\pi_2(a^\star \mid s) = 0.4,\ \pi_2(a' \mid s) = 0.1,\ \pi_2(a_{\text{bad}} \mid s) = 0.5$$
$$\Rightarrow \quad M_{\text{off}}(\pi_2) = 0.4,\ \eta_{\pi_2}(s) = 0.1,\ J(\pi_2) = 0.5.$$

Here the offline score increases ($0.2 \to 0.4$), while the off-demo validity mass changes dramatically ($0.7 \to 0.1$) and the true success probability drops sharply ($0.9 \to 0.5$). This happens because M_{off} only tracks probability on the single demonstrated action $a^\star$; it cannot distinguish whether probability mass is reallocated from other valid alternatives a' (uncredited) to invalid actions a_{bad}.

(iii) Offline overfitting can reduce online robustness. Example F.1 also illustrates another failure mode: maximizing an offline demo-matching score can encourage demo-specific behavior. Because $M_{\text{off}}(\pi)$ rewards only matching the single demonstrated action $\tilde{a}(s)$, a policy may increase $M_{\text{off}}(\pi)$ by concentrating probability mass on $\tilde{a}(s)$ while reducing exploration of other valid alternatives. This overfitting reduces behavioral diversity and weakens recovery strategies that are essential for interactive agents. In long-horizon GUI navigation, the impact is amplified by error accumulation. A small early mistake can move the agent to states that are poorly covered by the offline data, where the policy has not learned robust correction behaviors. This increases state-distribution shift (larger $C(\pi)$) and can lower the overall success probability $J(\pi)$. Consequently, it is possible for $M_{\text{off}}(\pi)$ to improve while $J(\pi)$ decreases.

Taken together, these failure modes explain why offline one-step matching can be poorly predictive of online success in multi-step GUI navigation: offline evaluation measures demo-matching under d_μ, whereas online success depends on long-horizon validity under $\{d_{\pi,t}\}$ and is confounded by unobserved (and potentially unstable) off-demo validity mass.

Incomplete coverage. The support condition concerns states visited by the evaluated policy, not all possible state–action pairs. To expose the cost of violating it, let $S_\mu = \{s : d_\mu(s) > 0\}$, $\zeta_t = \Pr_{s\sim d_{\pi,t}}(s \notin S_\mu)$, and $C_{S_\mu}(\pi) = \max_t \sup_{s\in S_\mu} d_{\pi,t}(s)/d_\mu(s)$. Without assuming full coverage, the same argument gives

$$J(\pi) \geq 1 - \sum_{t=1}^{H} \zeta_t - HC_{S_\mu}(\pi)\big(1 - M_{\text{off}}(\pi) - \bar{\eta}_\pi\big). \tag{5}$$

Indeed, split each $\mathbb{E}_{d_{\pi,t}}[f(s)]$ into S_μ and its complement. The first part is at most $C_{S_\mu}(\pi)\mathbb{E}_{d_\mu}[f(s)]$, and the second is at most ζ_t because $0 \leq f \leq 1$. Summing over time proves Eq. 5. KL can limit distributional drift, but does not by itself remove the uncovered-state term.

Finite data and imperfect demonstrations. The theorem is a population statement. A finite dataset provides an empirical estimate of M_{off}, not an oracle for d_μ, $C(\pi)$, or $\bar{\eta}_\pi$. In particular, empirical support in screenshot space should not be equated with population coverage. We do not claim a finite-sample or uniform checkpoint-selection guarantee. Our filtering seeks to improve demonstration quality, but the proof assumes $\tilde{a}(s) \in \mathcal{A}^*(s)$; erroneous demonstrations break the reliable-positive premise and require an additional error term. Likewise, the assumption that every failed episode contains an invalid action excludes failures caused solely by uncontrollable environment outcomes or insufficient horizon. These conditions define the scope of the analysis.

Lower bounds and empirical predictability. A rising lower bound does not imply rising task success or positive correlation across policies. Even with bounded $C(\pi)$ and stable ambiguity mass, policies can differ in bound slack. The analysis identifies mechanisms that can disrupt alignment; the reported checkpoint correlations provide the empirical evidence that KL improves alignment in our setting.

F.2 KL regularization improves predictability

Many RLVR pipelines drop the KL regularization term for efficiency [51, 25, 59, 49]. In our setting, however, KL regularization is important because step-wise offline matching is only partially verifiable: matches to the demonstrated action provide reliable positive signal, while non-matches are ambiguous. Below we connect KL regularization (to a reference policy) to the two quantities that govern offline-to-online predictability: the occupancy mismatch $C(\pi)$ and the off-demo validity mass $\bar{\eta}_\pi$.

Lemma F.1: KL regularization controls distribution shift

Let π_{ref} be a reference policy and assume a per-state KL constraint

$$\text{KL}(\pi(\cdot \mid s) \,\|\, \pi_{\text{ref}}(\cdot \mid s)) \leq \varepsilon, \qquad \forall s \in \mathcal{S}.$$

Then Pinsker's inequality implies

$$\text{TV}(\pi(\cdot \mid s), \pi_{\text{ref}}(\cdot \mid s)) \leq \sqrt{\varepsilon/2}, \qquad \forall s \in \mathcal{S}.$$

Consequently, the induced state visitation distributions satisfy

$$\|d_{\pi,t} - d_{\pi_{\text{ref}},t}\|_1 \leq t\sqrt{2\varepsilon}, \qquad \forall t \in [H].$$

Moreover, in a discrete-state setting, if $d_\mu(s) \geq \rho > 0$ on its support and both $d_{\pi,t} \ll d_\mu$ and $d_{\pi_{\text{ref}},t} \ll d_\mu$, then

$$\sup_{s:\, d_\mu(s)>0} \frac{d_{\pi,t}(s)}{d_\mu(s)} \leq \sup_{s:\, d_\mu(s)>0} \frac{d_{\pi_{\text{ref}},t}(s)}{d_\mu(s)} + \frac{\|d_{\pi,t} - d_{\pi_{\text{ref}},t}\|_1}{\rho} \leq C(\pi_{\text{ref}}) + \frac{t\sqrt{2\varepsilon}}{\rho}.$$

In particular, KL regularization bounds how much the occupancy mismatch $C(\pi)$ can grow relative to π_{ref}.

Proof. **Step 1: From KL to per-state TV.** By Pinsker's inequality, for every s,

$$\begin{aligned}\mathrm{TV}(\pi(\cdot \mid s), \pi_{\mathrm{ref}}(\cdot \mid s)) &\le \sqrt{\tfrac{1}{2}\,\mathrm{KL}(\pi(\cdot \mid s) \,\|\, \pi_{\mathrm{ref}}(\cdot \mid s))} \\ &\le \sqrt{\varepsilon/2}.\end{aligned}$$

Equivalently, for every measurable set $A \subseteq \mathcal{A}$,

$$\begin{aligned}\big|\pi(A \mid s) - \pi_{\mathrm{ref}}(A \mid s)\big| &\le \mathrm{TV}(\pi(\cdot \mid s), \pi_{\mathrm{ref}}(\cdot \mid s)) \\ &\le \sqrt{\varepsilon/2}.\end{aligned} \tag{6}$$

Step 2: One-step propagation bound. Let P_π denote the (time-homogeneous) state-transition operator induced by π:

$$(P_\pi \nu)(s') \triangleq \sum_{s \in \mathcal{S}} \nu(s) \sum_{a \in \mathcal{A}} \pi(a \mid s)\, P(s' \mid s, a),$$

and similarly define $P_{\pi_{\mathrm{ref}}}$. For any distribution ν over states, consider $\|\nu P_\pi - \nu P_{\pi_{\mathrm{ref}}}\|_1$. For each s', define

$$\begin{aligned}P_\pi(s' \mid s) &\triangleq \sum_a \pi(a \mid s) P(s' \mid s, a), \\ P_{\pi_{\mathrm{ref}}}(s' \mid s) &\triangleq \sum_a \pi_{\mathrm{ref}}(a \mid s) P(s' \mid s, a).\end{aligned}$$

Then

$$\|\nu P_\pi - \nu P_{\pi_{\mathrm{ref}}}\|_1 = \left\| \sum_s \nu(s) \big(P_\pi(\cdot \mid s) - P_{\pi_{\mathrm{ref}}}(\cdot \mid s)\big) \right\|_1 \le \sum_s \nu(s) \big\|P_\pi(\cdot \mid s) - P_{\pi_{\mathrm{ref}}}(\cdot \mid s)\big\|_1.$$

Moreover, for each fixed s,

$$\begin{aligned}\big\|P_\pi(\cdot \mid s) - P_{\pi_{\mathrm{ref}}}(\cdot \mid s)\big\|_1 &= \left\| \sum_a \big(\pi(a \mid s) - \pi_{\mathrm{ref}}(a \mid s)\big) P(\cdot \mid s, a) \right\|_1 \\ &\le \sum_a \big|\pi(a \mid s) - \pi_{\mathrm{ref}}(a \mid s)\big| \\ &= \|\pi(\cdot \mid s) - \pi_{\mathrm{ref}}(\cdot \mid s)\|_1.\end{aligned}$$

Using $\|\cdot\|_1 = 2\,\mathrm{TV}(\cdot, \cdot)$ for distributions,

$$\big\|P_\pi(\cdot \mid s) - P_{\pi_{\mathrm{ref}}}(\cdot \mid s)\big\|_1 \le 2\,\mathrm{TV}(\pi(\cdot \mid s), \pi_{\mathrm{ref}}(\cdot \mid s)) \le 2\sqrt{\varepsilon/2} = \sqrt{2\varepsilon}.$$

Therefore, for any ν,

$$\|\nu P_\pi - \nu P_{\pi_{\mathrm{ref}}}\|_1 \le \sqrt{2\varepsilon}. \tag{7}$$

Step 3: Telescoping over t steps. Let $d_{\pi,t}$ and $d_{\pi_{\mathrm{ref}},t}$ denote the state distributions at step t under π and π_{ref}, respectively. Using the recursion $d_{\pi,t+1} = d_{\pi,t} P_\pi$ and $d_{\pi_{\mathrm{ref}},t+1} = d_{\pi_{\mathrm{ref}},t} P_{\pi_{\mathrm{ref}}}$, we have

$$\begin{aligned}\|d_{\pi,t+1} - d_{\pi_{\mathrm{ref}},t+1}\|_1 &= \|d_{\pi,t} P_\pi - d_{\pi_{\mathrm{ref}},t} P_{\pi_{\mathrm{ref}}}\|_1 \\ &\le \underbrace{\|d_{\pi,t} P_\pi - d_{\pi,t} P_{\pi_{\mathrm{ref}}}\|_1}_{\le \sqrt{2\varepsilon} \text{ by } (7)} \\ &\quad + \underbrace{\|d_{\pi,t} P_{\pi_{\mathrm{ref}}} - d_{\pi_{\mathrm{ref}},t} P_{\pi_{\mathrm{ref}}}\|_1}_{\le \|d_{\pi,t} - d_{\pi_{\mathrm{ref}},t}\|_1}.\end{aligned}$$

Thus,

$$\|d_{\pi,t+1} - d_{\pi_{\mathrm{ref}},t+1}\|_1 \le \|d_{\pi,t} - d_{\pi_{\mathrm{ref}},t}\|_1 + \sqrt{2\varepsilon}.$$

Iterating this inequality and using $\|d_{\pi,0} - d_{\pi_{\mathrm{ref}},0}\|_1 = 0$ (same initial distribution) yields

$$\|d_{\pi,t} - d_{\pi_{\mathrm{ref}},t}\|_1 \le t\sqrt{2\varepsilon}.$$

Step 4: Bounding occupancy mismatch relative to d_μ. Assume both $d_{\pi,t} \ll d_\mu$ and $d_{\pi_{\text{ref}},t} \ll d_\mu$, with $d_\mu(s) \geq \rho > 0$ on its discrete support. The support condition for π is separate from the KL constraint; otherwise this step bounds only the ratio on covered states, as in Eq. 5. For any s with $d_\mu(s) > 0$,

$$\begin{aligned}\frac{d_{\pi,t}(s)}{d_\mu(s)} &= \frac{d_{\pi_{\text{ref}},t}(s)}{d_\mu(s)} + \frac{d_{\pi,t}(s) - d_{\pi_{\text{ref}},t}(s)}{d_\mu(s)} \\ &\leq \frac{d_{\pi_{\text{ref}},t}(s)}{d_\mu(s)} + \frac{|d_{\pi,t}(s) - d_{\pi_{\text{ref}},t}(s)|}{\rho}.\end{aligned}$$

Taking supremum over s with $d_\mu(s) > 0$ and using $\sup_s |x_s| \leq \sum_s |x_s| = \|x\|_1$ gives

$$\sup_{s:\, d_\mu(s)>0} \frac{d_{\pi,t}(s)}{d_\mu(s)} \leq \sup_{s:\, d_\mu(s)>0} \frac{d_{\pi_{\text{ref}},t}(s)}{d_\mu(s)} + \frac{\|d_{\pi,t} - d_{\pi_{\text{ref}},t}\|_1}{\rho} \leq C(\pi_{\text{ref}}) + \frac{t\sqrt{2\varepsilon}}{\rho},$$

which completes the proof. □

Lemma F.2: KL limits off-demo validity mass under single-demo supervision

Fix a state s and let $\tilde{a} = \tilde{a}(s)$ be the demonstrated action. Assume $\text{KL}(\pi(\cdot \mid s) \,\|\, \pi_{\text{ref}}(\cdot \mid s)) \leq \varepsilon$ and that the reference policy is demo-concentrated: $\pi_{\text{ref}}(\tilde{a} \mid s) \geq 1 - \delta(s)$. Then

$$1 - \pi(\tilde{a} \mid s) \leq \delta(s) + \sqrt{\varepsilon/2}, \qquad \eta_\pi(s) = \pi(\mathcal{A}^*(s) \setminus \{\tilde{a}\} \mid s) \leq 1 - \pi(\tilde{a} \mid s),$$

and therefore

$$\eta_\pi(s) \leq \delta(s) + \sqrt{\varepsilon/2}, \qquad \bar{\eta}_\pi \leq \bar{\delta} + \sqrt{\varepsilon/2}, \quad \bar{\delta} \triangleq \mathbb{E}_{s\sim d_\mu}[\delta(s)].$$

Proof. By Pinsker's inequality,

$$\text{TV}(\pi(\cdot \mid s), \pi_{\text{ref}}(\cdot \mid s)) \leq \sqrt{\varepsilon/2}.$$

Let $E \triangleq \{a \neq \tilde{a}\}$. Using the standard event bound for total variation distance,

$$\begin{aligned}\pi(E \mid s) &\leq \pi_{\text{ref}}(E \mid s) + \text{TV}(\pi(\cdot \mid s), \pi_{\text{ref}}(\cdot \mid s)) \\ &\leq \big(1 - \pi_{\text{ref}}(\tilde{a} \mid s)\big) + \sqrt{\varepsilon/2} \\ &\leq \delta(s) + \sqrt{\varepsilon/2}.\end{aligned}$$

Since $\pi(E \mid s) = 1 - \pi(\tilde{a} \mid s)$, we obtain $1 - \pi(\tilde{a} \mid s) \leq \delta(s) + \sqrt{\varepsilon/2}$.

Next, because $\mathcal{A}^*(s) \setminus \{\tilde{a}\} \subseteq E$, we have

$$\eta_\pi(s) = \pi(\mathcal{A}^*(s) \setminus \{\tilde{a}\} \mid s) \leq \pi(E \mid s) \leq \delta(s) + \sqrt{\varepsilon/2}.$$

Taking expectation over $s \sim d_\mu$ yields

$$\bar{\eta}_\pi = \mathbb{E}_{s\sim d_\mu}[\eta_\pi(s)] \leq \mathbb{E}_{s\sim d_\mu}[\delta(s)] + \sqrt{\varepsilon/2} = \bar{\delta} + \sqrt{\varepsilon/2},$$

which completes the proof. □

Stability without demo concentration. For $B_s = \mathcal{A}^*(s) \setminus \{\tilde{a}(s)\}$, the event characterization of total variation gives

$$|\eta_\pi(s) - \eta_{\pi_{\text{ref}}}(s)| = |\pi(B_s \mid s) - \pi_{\text{ref}}(B_s \mid s)| \leq \sqrt{\varepsilon/2}.$$

Averaging over the same d_μ yields $|\bar{\eta}_\pi - \bar{\eta}_{\pi_{\text{ref}}}| \leq \sqrt{\varepsilon/2}$. Thus KL controls changes in ambiguity even if the reference is not demo-concentrated; two policies in this trust region differ in $\bar{\eta}$ by at most $\sqrt{2\varepsilon}$. This is a stability result, not a claim that valid alternatives should be eliminated. Since $\bar{\eta}_\pi$ enters the success lower bound with a positive sign, its upper bound cannot be substituted to obtain a stronger success lower bound.

Takeaway. The offline-to-online bound shows that predictability depends on (i) distribution shift along the policy's own trajectories (captured by $C(\pi)$) and (ii) the unobserved probability mass on valid-but-uncredited actions under single-demo verification (captured by $\bar{\eta}_\pi$). Lemma F.1 bounds changes in visitation under a uniform KL constraint, while Lemma F.2 and the event bound above control ambiguity mass. These results motivate conservative updates. Population coverage, demo quality, and a uniform trust region remain assumptions; an empirical KL penalty is a practical control rather than a certificate that all of them hold.

G Prompt Templates

G.1 Prompt for Reasoning Augmentation

We use the following prompts with GPT-4.1 to generate reasoning traces, which form our initial reasoning dataset. We draw inspiration from AGUVIS [45] and further refine and adapt the prompt through iterative trial-and-error during our data generation process.

Prompt Template for Web Data

Please generate detailed reasoning and explain your logic according to the UI screenshot, instruction, interaction history and the reference action.

Instruction: {}
Interaction History: {}
Reference Action Description for Current Step: {}
Reference Action Command for Current Step: {}

Note: If there exists a referenced element, it is highlighted by a small red hollow circle in the screenshot.

Guidelines

- Carefully observe and interpret the screenshot to extract any relevant information that can inform your reasoning.
- Reason step by step to accomplish the instruction, aligning your thought process with the overall goal.
- Use the reference action as a guide, but do not simply repeat it. Instead, explain why you would take that action (if it is reasonable) without referring to it.
- Do not rely on the current reference action or the highlighted hollow circle as justification—they are outcomes of hindsight.
- Use a first-person perspective to express your reasoning, as if you are the one navigating and making decisions.
- If the question is primarily in a language other than English, please reason in English but provide the final answer in the corresponding language.
- The reference action may involve both clicking and writing simultaneously. In such cases, describe the action as: first click the target element, then type the desired text. You should specify the action type as Write and indicate the target element where the text should be entered in the json.

Response Format:
Return exactly one `<think></think> + <answer></answer>` pair for one step, with the following structured format:

`<think>`
Explain your reasoning in detail, step by step.
• State what you observe and extract useful information from the screenshot. Keep important information in the thought because you may need to refer to them later.
• Reflect on the instruction and interaction history to understand the context and current

status.
• Decide what action to take next and why, following the guidelines. Do not summarize—show your full chain of thought.
</think>
<answer>

{
"action_description": "Describe the current step action with a clear instruction",
"action_type": "The type of action to take, e.g., Click, Select, Write, KeyBoardPress, Scroll, Terminate or Answer",
"action_target": "Provide a description of the element you want to operate on the screen. (If action_type == Scroll and Answer, this field should be 'None'.) It should include the element's identity, type (button, input field, dropdown menu, tab, etc.), text on it (if have), and location hints if needed to disambiguate. If you want to write text into an input field, specify the input field's identity and necessary information. If you already clicked the input field in last step, you can leave this field as 'None'.",
"value": "Specify the input or selected value. For Write, Select, and Answer actions, provide the exact text or option. For KeyboardPress, specify the key(s). For Scroll, indicate the direction as in the reference action command. For other action types, use 'None'."
}
</answer>

Prompt Template for Mobile Data

Please generate detailed reasoning and explain your logic according to the UI screenshot, instruction, interaction history and the reference action.

Instruction:
{}

Interaction History:
{}

Reference Action Description for Current Step (labeled by human): {}
Reference Action Command for Current Step (labeled by human): {}
Note: If there exists a referenced element, it is highlighted by a small red hollow circle in the screenshot.

Guidelines

- Carefully observe and interpret the screenshot to extract any relevant information that can inform your reasoning.
- Reason step by step to accomplish the instruction, aligning your thought process with the overall goal.
- Use the reference action as a guide, but do not simply repeat it. Instead, explain why you would take that action (if it is reasonable) without referring to it.
- Do not rely on the current reference action or the highlighted hollow circle as justification—they are outcomes of hindsight.
- Use a first-person perspective to express your reasoning, as if you are the one navigating and making decisions.
- If the question is primarily in a language other than English, please reason in English but provide the final answer in the corresponding language.
- The reference action may involve both clicking and writing simultaneously. In such cases, describe the action as: first click the target element, then type the desired text. You should specify the action type as Write and indicate the target element where the text should be entered in the json.

Response Format:
Return exactly one <think></think> + <answer></answer> pair for one step, with the following structured format:

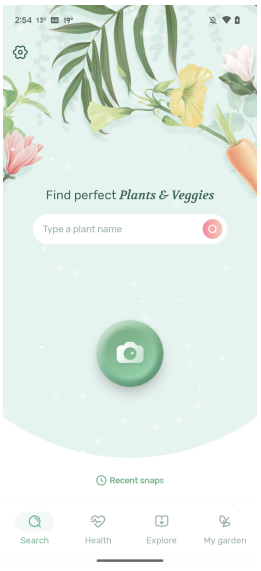
<think>
Explain your reasoning in detail, step by step.
■ State what you observe and extract useful information from the screenshot. Keep important information in the thought because you may need to refer to them later.
■ Reflect on the instruction and interaction history to understand the context and current status.
■ Decide what action to take next and why, following the guidelines.
Do not summarize—show your full chain of thought.
</think>
<answer>

{
"action_description": "Describe the current step action with a clear instruction",
"action_type": "The type of action to take, e.g., Click, Write, LongPress, Scroll, Swipe, NavigateHome, Wait, NavigateBack, Terminate, OpenAPP, or Answer. If the referece action is terminate and you need to answer the question from the user, you should specify the action type as Answer and provide the answer in the value field.",
"action_target": "Provide a description of the element you want to operate on the screen. (If action_type == Scroll, Terminate, or Answer, this field should be 'None'). It should include the element's identity, type (button, input field, dropdown menu, tab, etc.), text on it (if have), and location hints if needed to disambiguate. If you want to write text into an input field, specify the input field's identity and necessary information. If you already clicked the input field in last step, you can leave this field as 'None'.",
"value": "Specify the input or selected value. For Write, Select, and Answer actions, provide the exact text or option. For Scroll and Swipe, indicate the direction. For Wait, specify the wait time in seconds. For OpenAPP, specify the app name. For other action types, use 'None'.
"
}
</answer>

G.2 SFT Data Example

SFT Data Example

Image: 20162/screenshot_1.png



System Prompt:

```
You are a GUI agent. You are given a task and a screenshot of the
screen. You need to perform a series of actions to complete the
task. You need to choose actions from the following list:
action_type: Answer, action_target: None, value: Answer text,
point_2d: None
## Explanation: Return the final answer to the user's question
action_type: Click, action_target: Element description, value:
None, point_2d: [x, y]
## Explanation: Tap or click a specific UI element and provide
its coordinates
action_type: Write, action_target: Element description or None,
value: Text to enter, point_2d: [x, y] or None
## Explanation: Enter text into a specific input field or at the
current focus if coordinate is None
action_type: LongPress, action_target: Element description,
value: None, point_2d: [x, y]
## Explanation: Press and hold on a specific UI element (mobile
only) and provide its coordinates
action_type: Scroll, action_target: None, value: "up" | "down"
| "left" | "right", point_2d: None
## Explanation: Scroll a view or container in the specified
direction
action_type: Swipe, action_target: Optional position or None,
value: "up" | "down" | "left" | "right", point_2d: [x, y] or
None
## Explanation: Perform a swipe gesture on the screen in the
given direction (mobile)
action_type: Wait, action_target: None, value: Number of
seconds, point_2d: None
## Explanation: Pause execution to allow the UI to load or update
action_type: NavigateHome, action_target: None, value: None,
point_2d: None
## Explanation: Navigate to the device's home screen
action_type: NavigateBack, action_target: None, value: None,
point_2d: None
## Explanation: Press the system "Back" button
action_type: OpenApp, action_target: None, value: App name,
point_2d: None
## Explanation: Launch an app by its name (mobile only)
action_type: Terminate, action_target: None, value: End-task
message, point_2d: None
## Explanation: Signal the end of the current task with a final
message
```

Human Instruction and History:

```
Please generate the next move according to the UI screenshot
(origin image size 1080x2400), instruction and previous actions.
Instruction: Using the Blossom app, I would like to know the
botanical name of the ihi plant in order to identify the type of
plant it is.
Interaction History: Step 1 Action: Open the Blossom app to
start searching for the botanical name of the ihi plant.
```

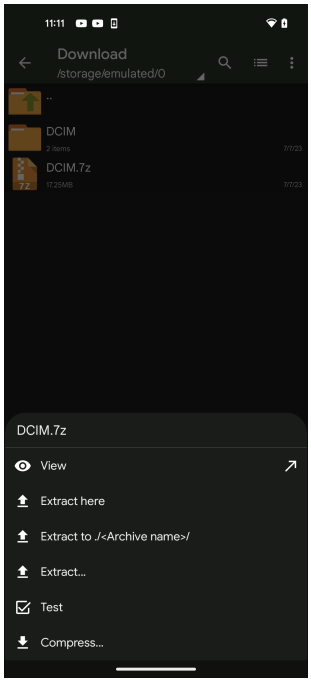
Response:

```
<think>
The screenshot shows that the Blossom app is now open, and I am on the
main search interface. The instruction is to find the botanical name
of the "ihi" plant. There is a prominent search bar near the center
of the screen labeled "Type a plant name," which is intended for
entering the name of a plant to search for information.

Since I need to search for the ihi plant and identify its botanical
name, the next logical step is to activate the search bar so I can
type "ihi" into it. This will allow me to perform a search for the plant
and get the relevant botanical information.
</think>
<answer>
{
  "action_description": "Tap on the search bar labeled
  'Type a plant name' to start entering the plant's name.",
  "action_type": "Click",
  "action_target": "Search bar input field labeled
  'Type a plant name' located below the text
  'Find perfect Plants & Veggies'",
  "value": "None",
  "point_2d": [500, 397]
}
</answer>
```

G.3 RL Data Example

Image:



System Prompt:

```
You are a GUI agent. You are given a task and a screenshot of the
screen. You need to perform a series of actions to complete the
task. You need to choose actions from the following list:
action_type: Answer, action_target: None, value: Answer text,
point_2d: None
## Explanation: Return the final answer to the user's question
action_type: Click, action_target: Element description, value:
None, point_2d: [x, y]
## Explanation: Tap or click a specific UI element and provide
its coordinates
action_type: Write, action_target: Element description or None,
value: Text to enter, point_2d: [x, y] or None
## Explanation: Enter text into a specific input field or at the
current focus if coordinate is None
action_type: LongPress, action_target: Element description,
value: None, point_2d: [x, y]
## Explanation: Press and hold on a specific UI element (mobile
only) and provide its coordinates
action_type: Scroll, action_target: None, value: "up" | "down"
| "left" | "right", point_2d: None
## Explanation: Scroll a view or container in the specified
direction
action_type: Swipe, action_target: Optional position or None,
value: "up" | "down" | "left" | "right", point_2d: [x, y] or
None
## Explanation: Perform a swipe gesture on the screen in the
given direction (mobile)
action_type: Wait, action_target: None, value: Number of
seconds, point_2d: None
## Explanation: Pause execution to allow the UI to load or update
action_type: NavigateHome, action_target: None, value: None,
point_2d: None
## Explanation: Navigate to the device's home screen
action_type: NavigateBack, action_target: None, value: None,
point_2d: None
## Explanation: Press the system "Back" button
action_type: OpenApp, action_target: None, value: App name,
point_2d: None
## Explanation: Launch an app by its name (mobile only)
action_type: Terminate, action_target: None, value: End-task
message, point_2d: None
## Explanation: Signal the end of the current task with a final
message
```

Human Instruction and History:

```
<image>
Please generate the next move according to the UI screenshot,
instruction and previous actions.
Instruction:  In the ZArchive App, decompress the DCIM.7z zip file
and save it to the Pocketbook folder.
Interaction History:
Step 1 Action:  Open the ZArchiver app to begin the decompression
process as required by the instruction.
Step 2 Action:  Tap on the 'DCIM.7z' archive file to open the
extraction options menu.

The response should be structured in the following format.
Make sure the output between <answer> and </answer> is a valid
JSON object.  Regarding the key "point_2d", please provide the
coordinates on the screen where the action is to be performed; if
not applicable, use [-100, -100].

<think> Your step-by-step thought process here... </think>
<answer>
{
  "action_description": "the description of the action to perform,
                         summarized in one sentence",
  "action_type": "the type of action to perform. Please follow the
  ↪  system prompt
                  for available actions.",
  "value": "the input text or direction ('up', 'down', 'left',
  ↪  'right')
                  for the 'scroll' action, if applicable;
                  ↪  otherwise, use 'None'",
  "point_2d": [x, y]
}
</answer>
```

The answer and the additional information used for reward computation are shown below. The `gt_point_2d` coordinates are normalized to the range $[0, 1]$, while `gt_bbox` is further scaled by a factor of 1000.

```
{
    'gt_action': 'Click',
    'gt_input_text': 'None',
    'gt_point_2d': [0.194, 0.843],
    'gt_target': "'Extract...' option in the extraction context menu for
    ↪  DCIM.7z",
    'image_height': 2400,
    'image_width': 1080,
    'gt_bbox': [36, 824, 268, 857]
}
```

H Long-Horizon Trajectory Case Studies

Figures 12 and 13 compare GUI-Libra-7B with its base model, Qwen2.5-VL-7B-Instruct, on AndroidWorld Task 18 (EXPENSEDELETEMULTIPLE). The task requires deleting three specific expenses in Pro Expense: *School Supplies*, *Religious*, and *Flight Tickets*. As shown in Figure 12, GUI-Libra successfully completes this long-horizon task by alternating between iterative reasoning and grounded actions. In contrast, the base model requires more steps to discover how to delete a single item and then fails to remove the second one, highlighting its difficulty in sustaining multi-step progress and demonstrating the advantage of GUI-Libra on long-horizon decision making.

We also include a web navigation case study in Figure 14, where GUI-Libra-4B successfully completes a long-horizon WebArena-Lite-v2 task, further illustrating strong generalization to multi-step web interactions.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, we summarize the contributions in abstract and intro.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: We discussed that we only use offline data from existing datasets without collecting new data.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: In Appendix F

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We add details about implementation with hyperpameters.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer:[Yes]

   Justification: We will open-source our data and code for public use.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We include all details in Appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We do not due to computational limitations.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer:[Yes]

   Justification: We discussed the gpu usage.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.

- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: We follow the code of conduct.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: We do not see any negative impact of our work.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: We do not see any risk of our research.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, we will include a license.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: Maybe in the future.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [No]

    Justification: We do not do this.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [Yes]

    Justification:Yes.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: We use LLM for writing.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.

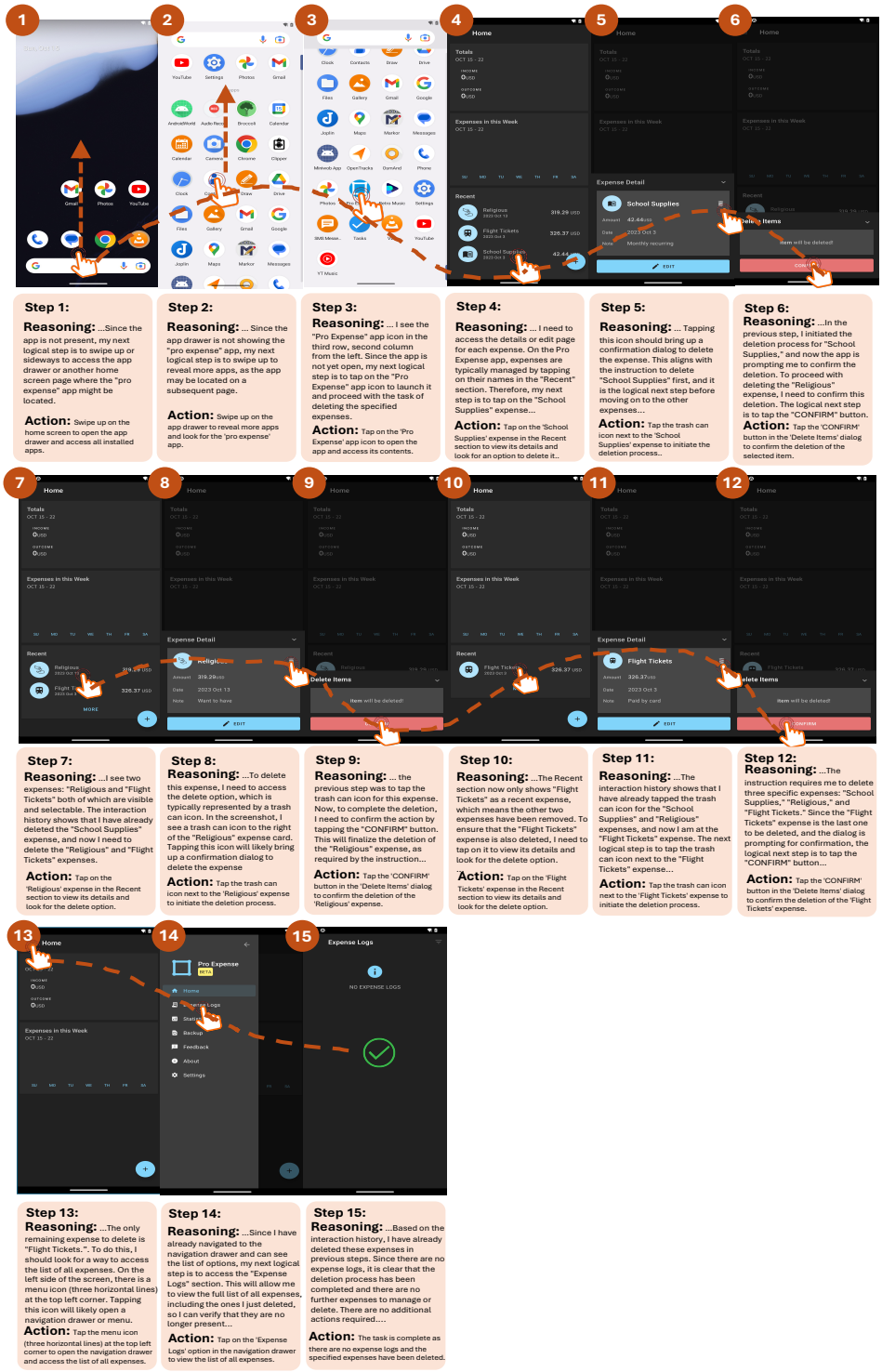


Figure 12: Trajectory Example of GUI-Libra-7B for Task 18 ExpenseDeleteMultiple: Delete the following expenses from pro expense: School Supplies, Religious, Flight Tickets.

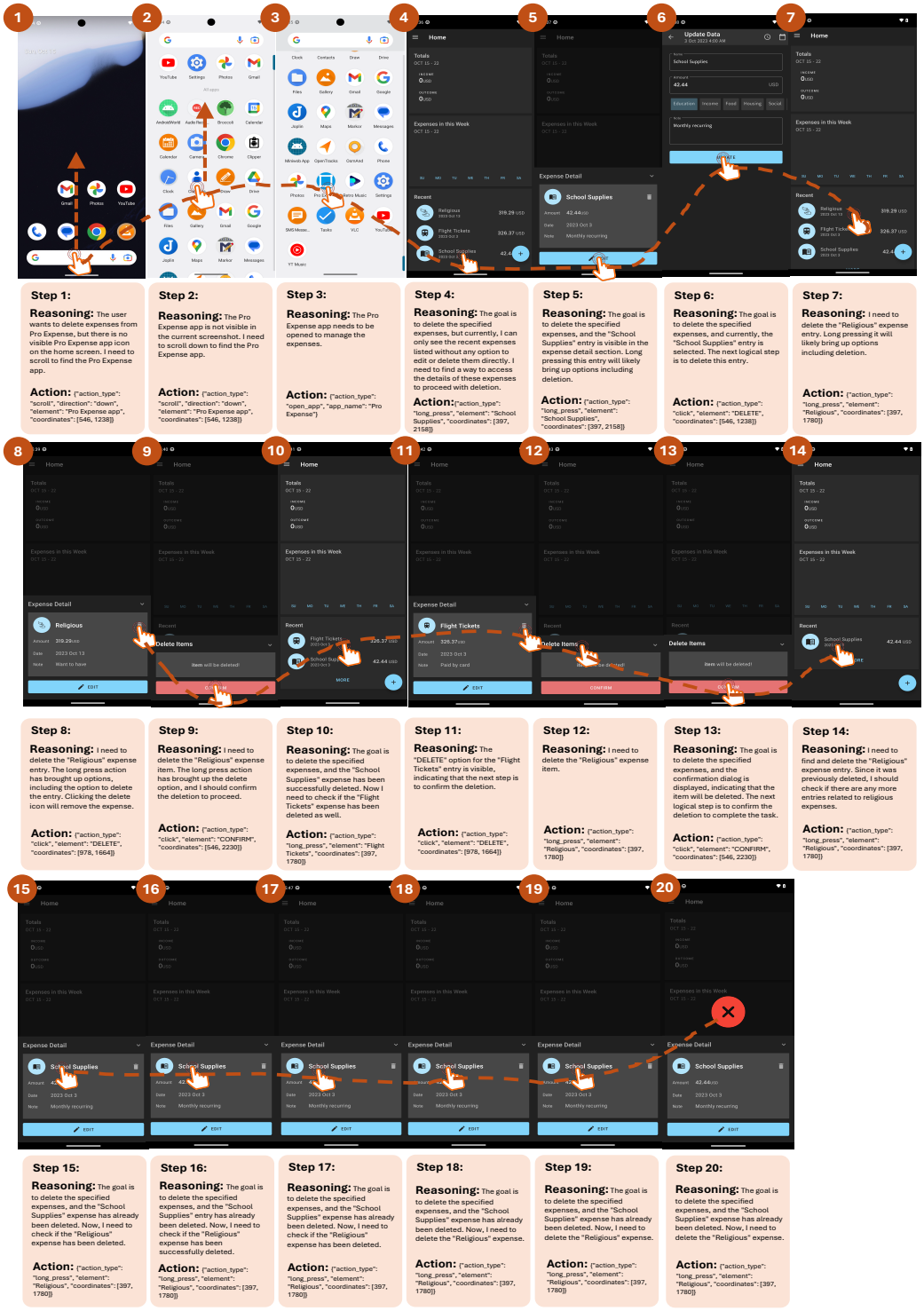


Figure 13: Trajectory Example of the base model Qwen2.5-VL-7B-Instruct for Task 18 ExpenseDeleteMultiple: Delete the following expenses from pro expense: School Supplies, Religious, Flight Tickets.



Figure 14: Trajectory Example of GUI-Libra-4B for WebArena-Lite-v2 Task: *Follow ['lahwaacz', 'Koushik', 'Vinta Chen'] on Gitlab.*

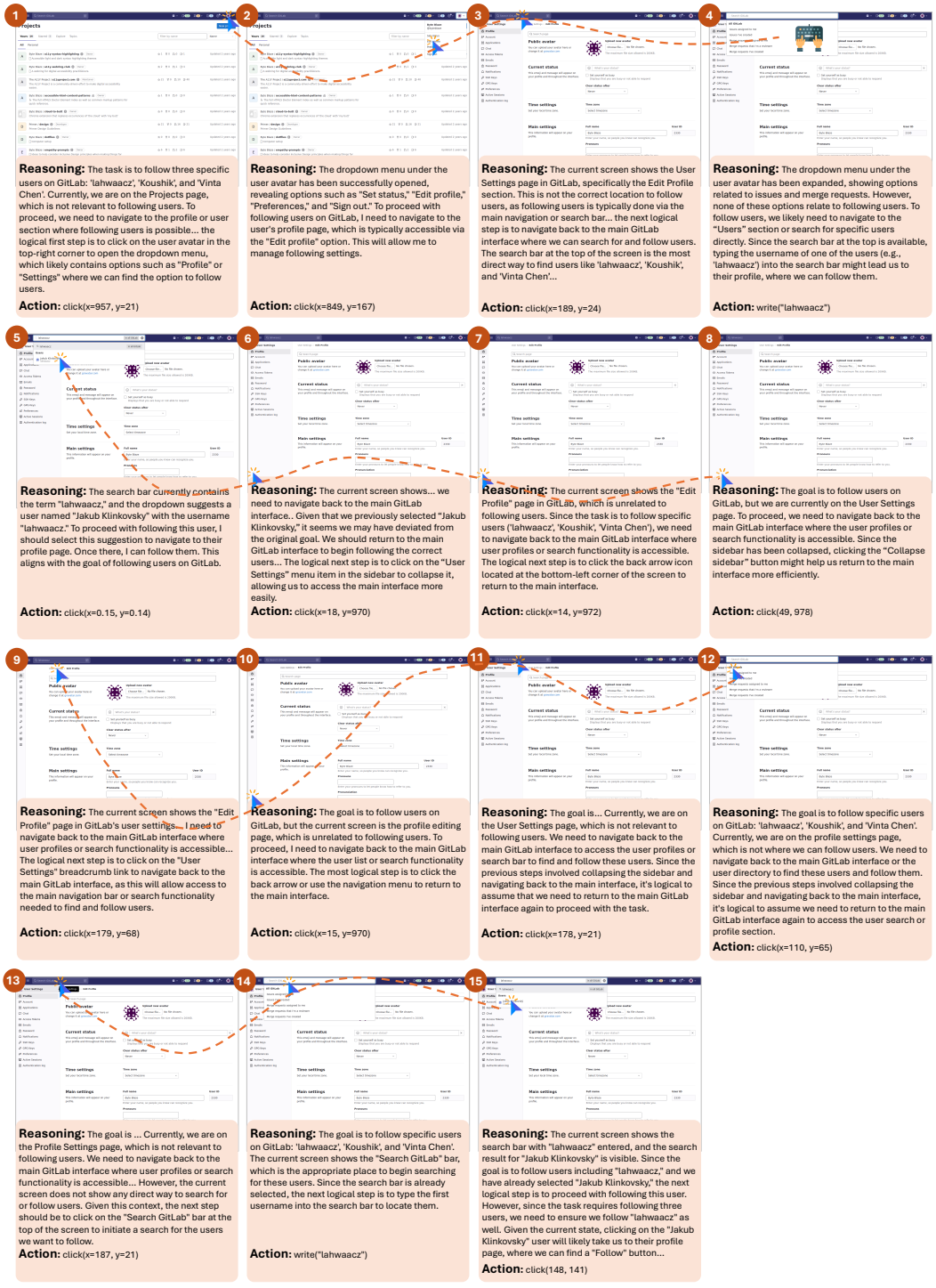


Figure 15: Trajectory Example of Qwen3-VL-4B-Instruct for WebArena-Lite-v2 Task: *Follow ['lahwaacz', 'Koushik', 'Vinta Chen'] on Gitlab.*