Mobile-R1: Towards Interactive Capability for VLM-Based Mobile Agent via Systematic Training

Jihao Gu[*†1], **Qihang Ai**[*†1], **Yingyao Wang**[*1], **Pi Bu**[*1], **Jingxuan Xing**[*3], **Yue Cao**[1], **Zekun Zhu**[3], **Wei Jiang**[3], **Ziming Wang**[1], **Yingxiu Zhao**[1], **Ming-Liang Zhang**[1], **Jun Song**[‡1,2], **Yuning Jiang**[‡3], **Bo Zheng**[1,2,3]
[1]Future Living Lab, Alibaba Group [2]Alibaba Group Holding Limited
[3]Taobao & Tmall Group of Alibaba
{gujihao.gjh, aiqihang.aqh, wangyingyao.wyy, bupi.wj, jsong.sj}@taobao.com

Abstract

Vision-language model-based mobile agents have gained the ability to understand complex instructions and mobile screenshots , benefiting from reinforcement learning paradigms like Group Relative Policy Optimization (GRPO). However, existing approaches centers on offline training or local action-level rewards often trap agents in local optima, hindering effective exploration and error correction with the environment. Crucially, we find that directly applying task-level rewards often leads to convergence difficulties due to the sparse nature of GUI interactions. To address these challenges, we present **Mobile-R1**, a systematic training recipe that bridges atomic action execution and strategic task completion. We propose a hierarchical curriculum consisting of three stages: (1) format alignment for reasoning structure, (2) on-policy exploration with verifiable action feedback to ground basic execution, and (3) multi-turn task-level training with realistic environment to unlock exploration and self-correction. This hierarchical strategy effectively bootstraps the agent, significantly enhancing its capability for exploration and self-correction (the "Eureka" moments). Furthermore, addressing the critical scarcity of diverse GUI data in non-English ecosystems, we contribute a comprehensive Chinese mobile dataset covering 28 applications with 24,521 high-quality manual annotations, and establish a rigorous benchmark with 500 trajectories. We will open source all resources, including the dataset, benchmark, model weight, and codes: https://mobile-r1.github.io/Mobile-R1/.

1 Introduction

Vision Language Model (VLM)-based agents have the capability to effectively integrate textual instructions with visual inputs, allowing them to devise comprehensive operational strategies and execute actions for complex tasks (Li and Huang, 2025; Gu et al., 2025). These agents not only comprehend intricate instructions but also engage in multi-turn planning (Nguyen et al., 2024; Huang et al., 2024) and interact with external tools or environments (Yuan et al., 2024; Shao et al., 2023), making them particularly well-suited for autonomous operation on mobile devices. Specifically, VLM-based mobile agents are driven by textual instructions, understand screenshots of mobile screens, and generate multi-turn actions to accomplish the task goals required by the instructions.

Several pioneers have explored relevant technologies. For instance, the AppAgent (Li et al., 2024c) and Mobile-Agent series (Wang et al., 2024b) introduced multi-modal agents, while UI-TARS (Qin et al., 2025) excels in comprehending screenshot content and navigating graphical user interfaces (GUIs). With the advancement of foundational models, such as Qwen2.5-VL (Bai et al., 2025), mobile agents are demonstrating even greater potential. Recently, studies like UI-R1 (Lu et al., 2025b) and GUI-R1 (Luo et al., 2025) draw inspiration from DeepSeek-R1 (Guo et al., 2025), attempting to use Group Relative Policy Optimization (GRPO) to guide the model's thinking and reasoning about the environment and actions. These above methods can adapt to immediate observations but struggle with the changing mobile environments, due to their reliance on action-level rewards that only guide the agent to predict the best single action at each step.

Notably, we emphasize that a mobile agent should engage in dynamic, multi-turn interactions with the mobile environment. Therefore, employing action-guided SFT or RL training alone is suboptimal and merely a temporary solution. As illustrated in Figure 1(a), the action-induced agent is encouraged to "think longer" merely to select a precise single-step action. This excessive supervision

[*]Equal contribution, Random order.
[†]Work done during an internship at Alibaba Group.
[‡]Corresponding Author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 30802–30820

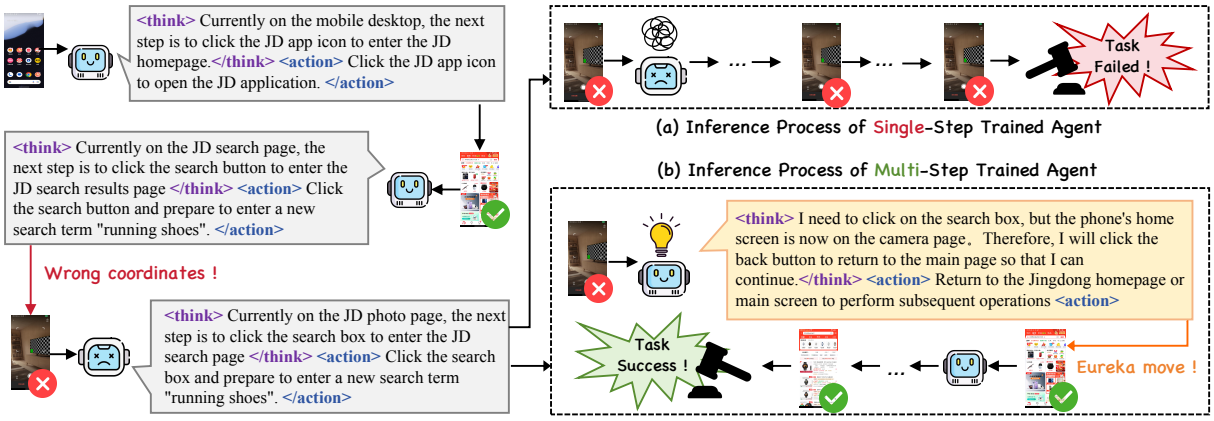

Figure 1: Comparison of Action-Level and Task-Level Rewards. Figure (a) illustrates an action-induced agent that is encouraged to "think longer" before selecting a single-step action. In contrast, Figure (b) depicts an agent trained at the task-level rewards, which explores and adjusts its trajectory over multi-turn interactions with the environment.

of action-level rewards can easily lead to local optima, significantly diminishing the model's capacity for exploration and self-correction. Conversely, multi-turn task-oriented learning is the ideal approach, as shown in Figure 1(b). However, directly applying task-level rewards often leads to training instability due to the sparse nature of success signals in complex GUIs. Successful training thus depends on satisfying three core requirements: **1) Multi-turn Trajectories:** the agent must incorporate historical trajectories to optimize for task completion rather than single turns. **2) Verifiable On-policy Exploration:** before attempting long-horizon tasks, the agent must first stabilize its interaction capability through online exploration with verifiable feedback, avoiding the cold-start failure of sparse rewards. **3) Trajectory Correction:** the agent should possess long-term planning and error reflection abilities to prevent getting trapped in local dilemmas.

In this paper, we strive to develop a systematic training recipe including realitic environment reinforcement learning, namely **Mobile-R1**, for the VLM-based mobile agent to acquire the interactive capability. To ensure stable training and bridge the gap between action execution and task planning, we propose a robust three-stage hierarchical curriculum: format finetuning, on-policy exploration with verifiable feedback, and task-level training. In Stage 1, we gather high-quality action trajectory samples, including self-correction trajectories, to enable the model to learn error correction formats (Cold Start). Thereafter, Stage 2 employs on-policy GRPO with **verifiable action feedback** (i.e., combining ground-truth matching with format constraints) to ground the agent's execution capability. In Stage 3, multi-turn GRPO training is conducted, focusing on task-level rewards that are applied to the entire trajectory. This stage is designed to encourage dynamic exploration within multi-turn interactions. Moreover, we introduce a new benchmark encompassing 500 trajectories with 1842 steps in total as well as a high-Quality trajectory dataset, addressing the under-representation of the Chinese ecosystem in existing mobile agent research. Our method demonstrates superior performance on this benchmark. Surprisingly, we discover that our agent is capable of correcting itself from an incorrect state back to the correct action (called the *eureka move*), demonstrating the emergence of reasoning capabilities through our multi-turn online reinforcement learning.

The main contributions are as follows:

- **Comprehensive Chinese Benchmark.** We introduce a challenging mobile agent benchmark that includes 500 trajectories with human annotation, filling the gap in non-English GUI evaluation.
- **High-Quality Trajectory Dataset.** We contribute a high-quality dataset featuring 4,635 manually annotated trajectories with 24,521 steps in total, which facilitates robust VLM-based agent training.
- **Robust Training Recipe for Mobile-R1.** We present a systematic three-stage training strategy that effectively bridges atomic action execution and task-level planning. Enabling multi-turns of interaction between the mobile agent and the environment. Experiments confirm this strategy enables critical self-correction behaviors.
- **Open Source Resources.** We will open source

all our resources, including the dataset, benchmark, model weight, and codes[1].

2 Related Work

2.1 Mobile Agent Framework

Graphical User Interface (GUI) agents are designed to operate in digital environments with graphical elements such as buttons and images. Their applications span web navigation, mobile app interactions, and desktop automation (Chen et al., 2025c). In the mobile agent, work has evolved from API-based frameworks using commercial models to open-source, end-to-end frameworks. Earlier API-based frameworks such as the AppAgent series (Zhang et al., 2025a; Li et al., 2024c) and the Mobile-Agent series (Wang et al., 2024b,a, 2025) used commercial models like GPT for planning and prediction, relying on complex workflows; MobileSteward (Liu et al., 2025b) extends this direction to cross-app instructions via self-evolving app-oriented agents. Recent advancements in open-source VLM have led to training these models on GUI-specific data. For instance, UI-TARS (Qin et al., 2025) continuously trains Qwen-2-VL (Wang et al., 2024c) models, specifically the 7B and 72B variants, on approximately 50 billion tokens. ShowUI (Lin et al., 2025) enhances Qwen2-VL-2B using UI-guided token selection and high-quality GUI datasets. UI-R1 (Lu et al., 2025b) explores rule-based reinforcement learning to boost VLMs' reasoning, while InquireMobile (Ai et al., 2025) further teaches the agent to actively request human assistance under the same RL paradigm.

Progress along these lines is increasingly assessed by dedicated benchmarks such as Mobile-bench (Deng et al., 2024) and AndroidLens (Cao et al., 2025), which probe LLM-driven workflows and long-latency tasks with nested sub-targets, respectively.

2.2 Visual Reasoning Model

DeepSeek R1 (Guo et al., 2025) showed that reinforcement learning with rule-based incentives helps large language models (LLMs) develop unique reasoning skills. Researchers are expanding this to multi-modal reasoning. VisualThinker-R1-Zero (Zhou et al., 2025) is the first to achieve enhanced visual reasoning with a non-SFT 2B

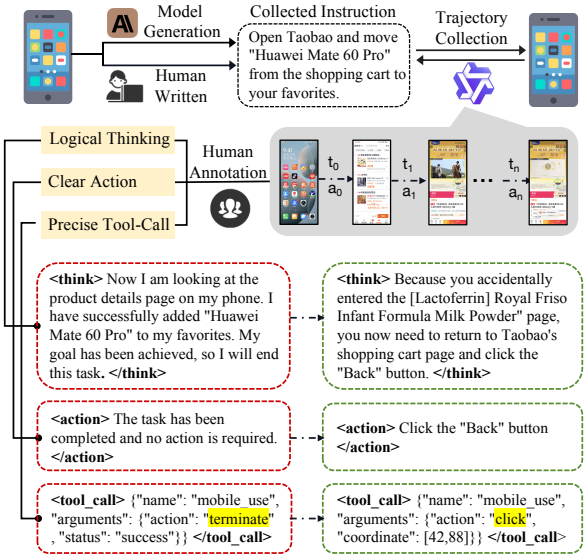

Figure 2: Pipeline of data collection.

model. Visual Reinforcement Fine-Tuning (Visual-RFT) (Liu et al., 2025c) targets visual tasks, including image classification, object detection, and reasoning grounding. Skywork R1 (Peng et al., 2025) uses multi-modal transfer to improve R1-series LLMs for visual tasks, combining SFT with reinforcement learning (i.e., GRPO) for better cross-modal reasoning. Notable works include R1-OneVision (Yang et al., 2025) and R1-V (Chen et al., 2025a), with R1-V exploring Reinforcement Learning with Verifiable Reward (RLVR) to boost VLMs' visual reasoning. It demonstrated that RLVR methods exhibit strong out-of-distribution generalization, whereas SFT excels in training domain tasks (Chen et al., 2025b).

3 Trajectory Dataset

To address the scarcity of data with explicit reasoning supervision and the under-representation of non-English ecosystems, we first constructed a high-quality dataset comprised of 4,635 trajectories with 24,521 manually annotated steps, supporting the community's efforts in training and evaluating powerful agents. Unlike prior datasets that only focus on action outcomes, ours explicitly captures the intermediate reasoning process within complex Chinese mobile applications. The pipeline of data collection is shown in Figure 2, which is divided into trajectory collection and trajectory annotation.

3.1 Trajectory Collection

We first selected 28 Chinese mobile apps, including both commercial and system types[2]. We cre-

[1] https://mobile-r1.github.io/Mobile-R1/

[2] All apps and prompts are provided in the Appendix.

Agents	Backbone	Training Strategy	Reward Source	Environment
Agent Q (Putta et al., 2024)	LLM-based	DPO	Action-level	Off-line
ARPO (Lu et al., 2025a)	VLM-based	GRPO	Task-level	On-line
UI-TARS (Qin et al., 2025)	VLM-based	DPO	Action-level	Off-line
GUI-R1 (Luo et al., 2025)	VLM-based	GRPO	Action-level	On-line
AgentCPM (Zhang et al., 2025b)	VLM-based	GRPO	Action-level	On-line
GUI-Critic-R1 (Wanyan et al., 2025)	VLM-based	GRPO	Action-level	On-line
InfiGUI-R1 (Liu et al., 2025a)	VLM-based	GRPO	Action-level	On-line
UI-R1 (Lu et al., 2025b)	VLM-based	GRPO	Action-level	On-line
Mobile-R1 (Ours)	VLM-based	GRPO	Action- & Task-level	On-line

Table 1: Comparison of several existing RL-based agent. Among them, "Reward Source" refers the agent's reward for executing an single action (i.e., action-level), or completing a task (i.e., task-level).

ated a diverse set of instructions for each app through manual crafting of common tasks and automatic generation by Claude 3.5 Sonnet (Anthropic, 2024). These instructions were manually reviewed, and any unreasonable ones were removed, leaving 1,510 instructions. Thereafter, we used the Qwen2.5-VL-3B (Bai et al., 2025) model as the mobile agent to execute these tasks, with a maximum of 25 steps, allowing multiple executions per instruction to simulate different mobile states for the same task. In this process, we gather a total of 4,635 raw action execution trajectories.

3.2 Trajectory Annotation

As shown in Figure 2, the action trajectory annotation process consists of a three step: logical thinking, clear action, and precise tool-calling.

Logical Thinking In this step, annotations ensure logical coherence and decision-making throughout the action trajectory. Annotators evaluate the model's "thinking" for errors or redundancies using a format:

```
<think>Currently on the phone home screen, the next step is to click the Taobao app to enter Taobao.<think>
```

The red part indicates the current state or interface, the blue part indicates the next action and the purple part indicates the goal of the action. Correct but redundant or incorrect data is rewritten.

Clear Action This stage involves clarifying instructions to ensure they are clear and explicit. These instructions guide actions according to ANDROIDCONTROL (Li et al., 2024b).

Precise Tool-Calling Through careful annotation in the previous two stages, we obtained thinking and basic instructional tasks. Here, "tool-calling" refers to mapping the action's natural language description into our standardized action space. Thereafter, annotators evaluate if actions are effective using the current screenshot and action descriptions. Correct tool-calls are kept, while incorrect ones are corrected.

3.3 Dataset Statistics

The overview of the dataset is shown in Table 2. The dataset consists of 4,635 high-quality, fine-grained mobile interaction trajectories, characterised by rigorous human verification and diverse task complexity. The distribution of the trajectory length is detailed in Figure 4.

Type	App	Instruction	Trajectory	Data
Number	28	1,510	4,635	24,521

Table 2: Overview of our trajectory dataset.

4 Our Mobile-R1

As shown in Figure 3, Mobile-R1 employs a curriculum learning framework designed to bootstrap reasoning in sparse-reward environments. The training recipe consists of three progressive stages: 1) Format Alignment (Warm-up) using the CoT data described in Section 3, 2) On-Policy Exploration with Verifiable Feedback to ground atomic execution, and 3) Multi-Turn Task-Level Optimization to unlock long-horizon planning and self-correction.

4.1 Preliminary

4.1.1 Task Definition

Mobile agents are driven by textual instructions, understand screenshots of mobile screens, and multi-turn generate action to accomplish the task goals. Instructions are divided into task level ("create an event for tomorrow at 2 pm") and action level ("click the icon in the top left corner").

4.1.2 Response Format

Following the pioneers, we format response as `<think>`, `<action>`, `<tool_call>`.

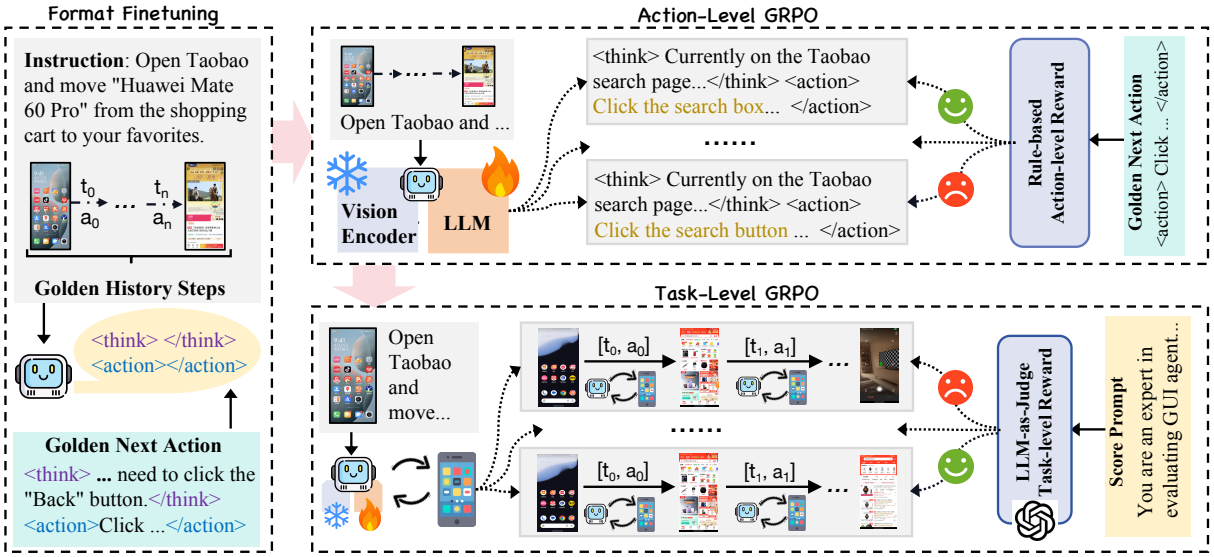

Figure 3: Our training framework consists of three stages: initial format finetuning, online training via action-level reward, followed by online training via task-level reward based on multi-turn trajectories.

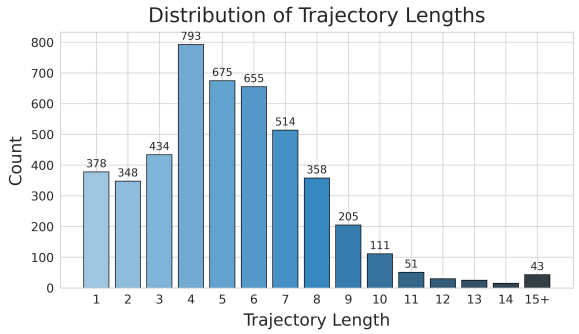

Figure 4: Distribution of trajectory length.

4.1.3 Action Space

We adopt a unified action space to ensure that all task-level instructions can be decomposed into a sequence of atomic actions via `<tool_call>`. There are eight actions: key, click, swipe, long press, type, system button, terminate, and wait[3].

4.1.4 Group Relative Policy Optimization

GRPO (Shao et al., 2024) builds on the Proximal Policy Optimization (PPO) by using group-normalized token-level advantages. This method tackles the issue of sparse or high-variance rewards in LLMs without needing a value function or critic.

Given a batch of G generated responses $\{o_i\}_{i=1}^{G}$ from a query, where each response $o_i = (o_i(1), \ldots, o_i(|o_i|))$, the GRPO objective function is defined as:

$$J_{\text{GRPO}}(\theta) = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\min\left[\frac{\pi_\theta(o_i(t)|o_i,<t)}{\pi_{\text{old}}(o_i(t)|o_i,<t)}\hat{A}_{i,t}, \text{clip}\left(\frac{\pi_\theta(o_i(t)|o_i,<t)}{\pi_{\text{old}}(o_i(t)|o_i,<t)}, 1-\epsilon, 1+\epsilon\right)\hat{A}_{i,t}\right] \quad (1)$$

where: $\pi_\theta(o_i(t)|o_i,<t)$ and $\pi_{\text{old}}(o_i(t)|o_i,<t)$ are the probabilities of generating token $o_i(t)$ under the current and old policies, respectively. ϵ is the clipping hyperparameter for the probability ratio. $\hat{A}_{i,t}$ is the group-normalized advantage for token $o_i(t)$ in response o_i:

$$\hat{A}_{i,t} = \frac{r_i - \mu}{\sigma}, \quad (2)$$

with r_i being the total reward for response o_i, and μ, σ being the mean and standard deviation of all rewards $\{r_j\}_{j=1}^{G}$ within the current batch.

4.2 Stage1: Format Finetuning

This stage aims to train the model to predict the next action from instructions and operation history. To equip the model with this fundamental capability, we start with initial Supervised Fine-Tuning (SFT) as cold start using the action-level instructions from Section 3. This essential stage builds a strong link between user intent, GUI state, and actions, supporting later reinforcement learning.

4.3 Stage2: On-Policy Exploration with Verifiable Feedback

Subsequently, to prevent the "cold-start" failure common in sparse-reward settings, the model was

[3]The definitions can be found in the Appendix.

trained using GRPO through action-level rewards with verifiable feedback (derived from ground-truth annotations) as a dense reward signal. The reward function at this stage is a combination of two components: a verifiable action reward and a format reward, as follows:

$$R_{action} = R_{Act} + R_{Fmt}, \tag{3}$$

where R_{Act} quantifies the correctness of the executed action, and R_{Fmt} ensures the format integrity of the generated output.

Action-Level Reward (R_{Act})

R_{Act} assesses the correctness of the action prediction, with its calculation varying based on the action type.

- For coordinate-based actions (e.g., click, swipe), R_{Act} is 1 if the predicted coordinate $C = [x, y]$ falls within the ground truth bounding box $B = [x_1, y_1, x_2, y_2]$ of the target GUI element; otherwise, it is 0. This ensures precise spatial interaction.
- For non-coordinate actions (e.g., 'type(text=)'), R_{Act} is 1 if the predicted action or its argument (e.g., the text string to type, the specific 'back' command) exactly matches the ground truth; otherwise, it is 0. This guarantees faithful command execution.

Format Reward (R_{Fmt})

R_{Fmt} incentivizes the model to produce structured, interpretable outputs.

- `<think>`: The internal reasoning processes.
- `<action>`: The immediate next action to be executed.
- `<tool_call>`: The final answer or tool/API invocation for the current step.

R_{Fmt} is a binary reward (1 for full compliance, 0 otherwise) that enforces adherence to these tagging and structural requirements, ensuring well-formed and semantically organized responses.

4.4 Stage3: Task-level Online Training

While Stage 2 ensures atomic precision, it encourages greedy local optima. In Stage 3, we transition to multi-step task-level GRPO to enable strategic exploration. Here, the agent receives a reward upon task completion, forcing it to credit assignment over long horizons, enabling the robuteness of agent and emergence of "Eureka" moments.

Firstly, we define this multi-turn interaction problem as a Markov decision process (Lu et al., 2025a).

Each trajectory τ is a sequence of observations s_t (representing mobile screenshot), agent actions a_t (including thinking, action and tool_use), and a scalar reward R obtained upon trajectory completion. The core objective is to train a policy π_θ that maximizes the accumulated rewards over these interaction sequences, where a_t is sampled from the policy conditioned on the preceding state-action history:

$$\begin{aligned} &\tau = \{s_t, a_t\}_{t=0}^{T-1}, \quad \text{where} \\ &a_t \sim \pi_\theta(\{s_i\}_{i=\max(0,t-W)}^{t-1}, \{a_i\}_{i=0}^{t-1}). \end{aligned} \tag{4}$$

We define W as the sliding window size, which controls the maximum number of observations (s_i) considered. It only caps the number of *historical screenshots* $\{s_i\}$ fed to the model, which is a VRAM constraint. The full textual history ($\langle$think$\rangle$, $\langle$action$\rangle$, $\langle$tool_call$\rangle$) is always preserved, so the agent can still reason over steps outside the visual window (e.g., recognizing an earlier mis-click). The specific value of W is given in Appendix E.

The reward R_{task} of this stage is composed of two components: a Format Reward (R_{Fmt}) and a Trajectory-Level Reward (R_{Traj}), formulated as:

$$R_{task} = R_{Fmt} + R_{Traj}. \tag{5}$$

Format Reward (R_{Fmt})

Differing from Stage 2, in this stage, R_{Fmt} for the entire trajectory is obtained by averaging the format reward of all actions. Moreover, R_{Fmt} is set to $[-1, 1]$ to impose stricter penalties for errors.

Trajectory-Level Reward (R_{Traj})

To obtain a comprehensive evaluation signal for multi-turn interactions, an external, high-fidelity MLLM, GPT-4o (OpenAI, 2023), is employed to assign a scalar reward score to the entire historical interaction trajectory $\tau = (s_0, a_0, \ldots, a_n)$.

Drawing inspiration from prior work (Sun et al., 2024), we establish two primary scoring criteria for GPT-4o[4]:

[4]The version we used is GPT-4o-0806. Prompts provided to GPT-4o can be found in the Appendix.

- Trajectory Coherence: This checks if steps and actions consistently follow the target instruction, actions are clear and specific, and if there are no unnecessary steps.

- Task Completion: This evaluates if the task is fully completed, all necessary interactions are made, and errors are handled properly.

The 5-level scoring rubric is applied by GPT-4o, yielding a final score within the range $[0, 1]$.

5 Experiment

5.1 Implementation Details

5.1.1 Virtual Environment

The Android Studio emulator serves as our primary mobile GUI interactive environment. A local monitoring script runs alongside the emulator, actively managing the interaction loop.

5.1.2 Datasets and Benchmark

In the first two stages, we utilized 1,000 and 3,459 trajectory samples. In the third stage, we trained using only five frequently used Android apps—Jingdong, Pinduoduo, Taobao, Fliggy, and Bilibili—creating 20 unique task trajectories per app, totaling 100. For our evaluation benchmark, we separate 500 human-annotated trajectories with 1,842 steps from the dataset, of which 225 trajectories are specifically from long-tail unseen applications to better evaluate generalization. We also evaluated English benchmark AndroidControl(Li et al., 2024a) and GUI-Odyssey(Lu et al., 2024).

5.1.3 Training Settings

Our experiments utilize Qwen2.5-VL-3B as the base model, with GRPO implementation (including for trajectory-level interaction training) adapted from the open-r1 framework (Face, 2025). For hyperparameter settings, Stage 1: train for 2 epochs at a learning rate of 1×10^{-4}. Stage 2: train for 2 epochs at 1×10^{-7} with 8 rollouts and a temperature of 1 for exploration. Stage 3: train for 2 epochs at 1×10^{-6} with a temperature of 1.

More detailed hyperparameter configurations are listed in the Appendix E.

5.1.4 Baselines

Qwen2.5-VL-3B (Bai et al., 2025), UI-R1-3B, UI-R1-3B-E (Lu et al., 2025b), GUI-R1-3B, GUI-R1-7B (Luo et al., 2025), AgentCPM-8B (Zhang et al., 2025b) are baselines.

Model	Acc.	Task Succ.	Tail Succ.	Avg Err.↓
Qwen2.5-VL-3B	54.49	7.20	16.33	651
Qwen2.5-VL-7B	63.46	12.80	21.60	523
Qwen2.5-VL-32B	75.90	30.40	-	280
UI-R1-3B	56.13	17.20	-	451
UI-R1-3B-E	59.12	9.40	-	473
GUI-R1-3B	61.29	12.00	-	461
GUI-R1-7B	71.72	**32.60**	-	298
AgentCPM-8B	71.65	30.00	-	338
Mobile-R1 (Ours)	**78.55**	<u>30.60</u>	**37.40**	**241**
Stage1 & Stage2	<u>77.69</u>	29.40	**36.00**	<u>255</u>
Stage1	75.68	24.40	29.80	280

Table 3: Overall Performance Comparison. **Bold** and <u>underline</u> indicate the best and second-best results.

5.1.5 Evaluation Metrics

We evaluate the model's performance using the following metrics:

- **Accuracy (Acc.):** The probability of correctly performing each step in a trajectory, correct if both format and action match definitions R_F and R_{Act}.

- **Task Success Ratio (Task Succ.):** The probability of a complete trajectory being executed entirely correctly.

- **Tail Success Ratio (Tail Succ):** The probability that the task within a trajectory is ultimately completed successfully, regardless of intermediate errors or deviations.

- **Action Argument Error Number (Avg Err.):** The count of errors of incorrect action.

5.2 Experimental Result

We evaluated all models on our benchmark, with experimental results shown in Table 3. We have the following observations: 1) Our Mobile-R1 outperformed all baselines, achieving an accuracy of 78.55, 2.7 points higher than the best baseline. With Stage 3 training, the Mobile-R1's Task Success Ratio increased by 1.2 points over the Stage 1 & 2 model, benefiting from the task-level GRPO. 2) Our Stage 1 & 2 allows the Qwen2.5-VL-3B model to surpass its standard version and outperform baselines, highlighting the importance of action- and task-level rewards.

Notably, to further verify the necessity of our hierarchical training recipe, we conducted an experiment that directly proceeds from Stage 1 to Stage 3. Skipping Stage 2 causes the reward to plateau throughout Stage 3. Without the dense supervision provided by verifiable action rewards in

Model	Android Low		Android High		Odyssey		AITZ	
	TM	EM	TM	EM	TM	EM	TM	EM
Qwen2.5-VL-7B	**94.1**	85.0	75.1	62.9	59.54	46.28	–	–
OS-Genesis-7B	90.7	74.2	65.9	44.4	11.67	3.63	19.98	8.45
OS-Atlas-7B	73.0	67.3	70.4	56.5	**91.83**	**76.76**	74.13	58.45
Aguvis-7B	93.9	89.4	65.6	54.2	26.71	13.54	35.71	18.99
Odyssey-7B	65.1	39.2	58.8	32.7	90.83	73.67	59.17	31.60
Mobile-R1(3B)	93.5	**87.1**	**76.5**	**65.2**	75.24	53.11	**77.05**	**60.50**

Table 4: Performance Comparison on Android Control (Android), GUI-Odyssey (Odyssey), and AITZ (Zhang et al., 2024). **Bold** indicate the best results. "–" denotes that AITZ numbers are not reported for the base model.

Stage 2, the agent fails to accumulate meaningful task-level signals under the sparse-reward setting and thus cannot improve upon the SFT baseline. We have tested reward-versus-steps curve of our Stage 3 shown in Appendix G.

Moreover, we extended our evaluation to the English benchmark, Android Control and GUI-Odyssey with related works (Sun et al., 2024; Wu et al., 2024; Xu et al., 2024; Lu et al., 2024). These benchmarks utilize two standard metrics: Type Match (TM), which verifies if the predicted action type matches the ground truth, and Exact Match (EM), which additionally requires all parameters to be correct. Table 4 showed that, for Android Control, our 3B model demonstrated significantly superior performance compared to a range of 7B models. The sole exception was a slightly lower score on the TM-Low metric, which we attribute to the inherent limitations of a 3B model in advanced instruction-following. For GUI-Odyssey, our model achieved robust performance with only 3B parameters. All results are second only to the Odyssey-7B and Aguvis-7B models that underwent targeted training. Overall, these results underscore our model's robust and competitive performance on English-language tasks.

Moreover, we extended our evaluation to the English benchmarks Android Control, GUI-Odyssey, and AITZ (Sun et al., 2024; Wu et al., 2024; Xu et al., 2024; Lu et al., 2024; Zhang et al., 2024). These benchmarks utilize two standard metrics: Type Match (TM), which verifies if the predicted action type matches the ground truth, and Exact Match (EM), which additionally requires all parameters to be correct. Table 4 showed that, for Android Control, our 3B model demonstrated significantly superior performance compared to a range of 7B models. The sole exception was a slightly lower score on the TM-Low metric, which we attribute to the inherent limitations of a 3B model in advanced instruction-following. For GUI-Odyssey, our model achieved robust performance with only 3B parameters; all results are second only to the Odyssey-7B and Aguvis-7B models that underwent targeted training. On AITZ, Mobile-R1 (3B) further outperforms every 7B baseline, reaching 77.05 TM and 60.50 EM, indicating that our three-stage recipe generalizes well beyond the Chinese apps seen during training.

Overall, these results underscore our model's robust and competitive performance on English-language tasks.

5.3 Robustness Analysis

To assess the generalization capabilities of Mobile-R1, we validated our method's impact on end-to-end task performance by evaluating the success rate across 50 random tasks on physical devices. As detailed in Fiure 6, the model exhibited a consistent improvement trajectory. A minor initial dip in performance was observed during Stage 1, attributable to the action-level pre-training. Nevertheless, our method culminated in a significant 24 percentage point improvement in the final task success rate.

5.4 Qualitative Visualization

To effectively illustrate the performance of our Mobile-R1, we randomly selected several cases from the test set for qualitative analysis. As shown in Figure 5, we have the following observations: 1) Qwen2.5-VL-3B-Instruct failed at the second step, while Mobile-R1 completed the entire task accurately. 2) At the second step, Qwen2.5-VL-3B-Instruct clicked the wrong icon labeled "Hotel" when the correct choice was "Hotel Package." Mobile-R1 was able to click the correct icon, displaying precise icon identification. 3) At the final

Figure 5: Comparison of reasoning trajectories between Mobile-R1 and Qwen2.5-VL-3B-Instruct on the task "*Open Fliggy, enter the hotel package, enter the popular live broadcast, find Fliggy Super VIP, and follow the anchor*". In this case, Qwen2.5-VL-3B-Instruct failed at the second step, while Mobile-R1 completed the whole task accurately.

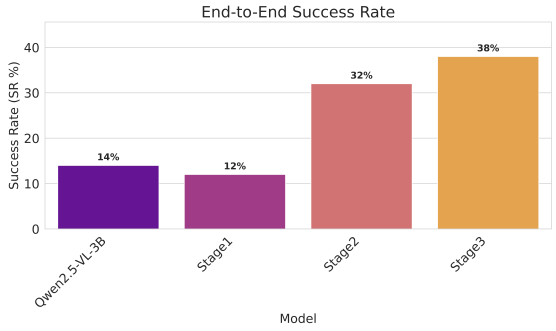

Figure 6: Task success on physical devices.

step, Mobile-R1 accurately recognized that the tab had already been marked as "Followed," which eliminated the need for further actions, thus concluding the task with superior contextual awareness.

6 Conclusion

In this paper, we introduced a systematic training recipe and bridge the community's need for high-quality Chinese GUI training and evaluation data. Through our three-stage training process—including format finetuning, action-level GRPO training, and task-level GRPO training in realitic dynamic environments—Mobile-R1 significantly advances the capabilities of VLM-based mobile agents, demonstrating superior exploration and error correction abilities, and overcoming the limitations of prior methods that rely solely on single action prediction. Experimental results demonstrate that our Mobile-R1 surpasses all baselines in all metrics.

7 Limitations

From the perspective of training strategy, while Mobile-R1 demonstrates promising capabilities in robust GUI interactionthis, our approach is essentially a systematic training recipe that employs both action-level and task-level rewards to guide the RL process, enabling the agent to improve progressively. It provides the interaction logic between GUI and environment for end-to-end training; however, we introduce no algorithmic innovation in RL itself. This will be a future direction: devising novel RL methods and refining reward design to achieve more efficient end-to-end interaction with mobile environments. Regarding dataset, our final dataset, after rigorous quality filtering and difficulty balancing, represents only a small fraction of the overall data source. Therefore, expanding the dataset size and incorporating more languages will be the focus of our future work.

Stage 3 of our Mobile-R1 uses a sliding window of size W over historical screenshots (Eq. 4; value in Appendix E) due to VRAM limits. The full *textual* history is retained, which suffices for the self-correction behaviors we observe (Fig. 1b, Fig. 9), but long-range *visual* recall beyond W steps is intrinsically restricted. Scaling W via visual-token compression or external memory is left for future work.

8 Ethical Considerations

Data Privacy: Our dataset was collected from publicly available applications. We have strictly

anonymized all sensitive Personally Identifiable Information (PII), such as phone numbers, addresses, and account details, during both the collection and annotation phases to protect user privacy.

References

Qihang Ai, Pi Bu, Yue Cao, Yingyao Wang, Jihao Gu, Jingxuan Xing, Zekun Zhu, Wei Jiang, Zhicheng Zheng, Jun Song, and 1 others. 2025. Inquiremobile: Teaching vlm-based mobile agent to request human assistance via reinforcement fine-tuning. *arXiv preprint arXiv:2508.19679*.

Anthropic. 2024. Claude 3.5 sonnet. https://www.anthropic.com/news/claude-3-5-sonnet.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and 1 others. 2025. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Yue Cao, Yingyao Wang, Pi Bu, Jingxuan Xing, Wei Jiang, Zekun Zhu, Junpeng Ma, Sashuai Zhou, Tong Lu, Jun Song, and 1 others. 2025. Androidlens: Long-latency evaluation with nested sub-targets for android gui agents. *arXiv preprint arXiv:2512.21302*.

Liang Chen, Lei Li, Haozhe Zhao, Yifan Song, and Vinci. 2025a. R1-v: Reinforcing super generalization ability in vision-language models with less than $3. Accessed: 2025-02-02.

Liang Chen, Lei Li, Haozhe Zhao, Yifan Song, Vinci, Lingpeng Kong, Qi Liu, and Baobao Chang. 2025b. Rlvr in vision language models: Findings, questions and directions. *Notion Post*.

Peng Chen, Pi Bu, Yingyao Wang, Xinyi Wang, Ziming Wang, Jie Guo, Yingxiu Zhao, Qi Zhu, Jun Song, Siran Yang, Jiamang Wang, and Bo Zheng. 2025c. Combatvla: An efficient vision-language-action model for combat tasks in 3d action role-playing games. *Preprint*, arXiv:2503.09527.

Shihan Deng, Weikai Xu, Hongda Sun, Wei Liu, Tao Tan, Liujianfeng Liujianfeng, Ang Li, Jian Luan, Bin Wang, Rui Yan, and 1 others. 2024. Mobilebench: An evaluation benchmark for llm-based mobile agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8813–8831.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Jihao Gu, Yingyao Wang, Pi Bu, Chen Wang, Ziming Wang, Tengtao Song, Donglai Wei, Jiale Yuan, Yingxiu Zhao, Yancheng He, Shilong Li, Jiaheng Liu, Meng Cao, Jun Song, Yingshui Tan, Xiang Li, Wenbo Su, Zhicheng Zheng, Xiaoyong Zhu, and Bo Zheng. 2025. "see the world, discover knowledge": A chinese factuality evaluation for large vision language models. *Preprint*, arXiv:2502.11718.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. 2024. Understanding the planning of llm agents: A survey. *arXiv preprint arXiv:2402.02716*.

Jiahao Li and Kaer Huang. 2025. A summary on gui agents with foundation models enhanced by reinforcement learning. *arXiv preprint arXiv:2504.20464*.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. 2024a. On the effects of data scale on computer control agents. *arXiv e-prints*, pages arXiv–2406.

Wei Li, William E Bishop, Alice Li, Christopher Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. 2024b. On the effects of data scale on ui control agents. *Advances in Neural Information Processing Systems*, 37:92130–92154.

Yanda Li, Chi Zhang, Wanqi Yang, Bin Fu, Pei Cheng, Xin Chen, Ling Chen, and Yunchao Wei. 2024c. Appagent v2: Advanced agent for flexible mobile interactions. *arXiv preprint arXiv:2408.11824*.

Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Stan Weixian Lei, Lijuan Wang, and Mike Zheng Shou. 2025. Showui: One vision-language-action model for gui visual agent. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19498–19508.

Yuhang Liu, Pengxiang Li, Congkai Xie, Xavier Hu, Xiaotian Han, Shengyu Zhang, Hongxia Yang, and Fei Wu. 2025a. Infigui-r1: Advancing multimodal gui agents from reactive actors to deliberative reasoners. *Preprint*, arXiv:2504.14239.

Yuxuan Liu, Hongda Sun, Wei Liu, Jian Luan, Bo Du, and Rui Yan. 2025b. Mobilesteward: Integrating multiple app-oriented agents with self-evolution to automate cross-app instructions. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 1*, pages 883–893.

Ziyu Liu, Zeyi Sun, Yuhang Zang, Xiaoyi Dong, Yuhang Cao, Haodong Duan, Dahua Lin, and Jiaqi Wang. 2025c. Visual-rft: Visual reinforcement fine-tuning. *arXiv preprint arXiv:2503.01785*.

Fanbin Lu, Zhisheng Zhong, Shu Liu, Chi-Wing Fu, and Jiaya Jia. 2025a. Arpo: End-to-end policy optimization for gui agents with experience replay. *arXiv preprint arXiv:2505.16282*.

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

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

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

Dang Nguyen, Jian Chen, Yu Wang, Gang Wu, Namyong Park, Zhengmian Hu, Hanjia Lyu, Junda Wu, Ryan Aponte, Yu Xia, and 1 others. 2024. Gui agents: A survey. *arXiv preprint arXiv:2412.13501*.

R OpenAI. 2023. Gpt-4 technical report. arxiv 2303.08774. *View in Article*, 2(5).

Yi Peng, Xiaokun Wang, Yichen Wei, Jiangbo Pei, Weijie Qiu, Ai Jian, Yunzhuo Hao, Jiachun Pan, Tianyidan Xie, Li Ge, and 1 others. 2025. Skywork r1v: Pioneering multimodal reasoning with chain-of-thought. *arXiv preprint arXiv:2504.05599*.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents. *Preprint*, arXiv:2408.07199.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, and 1 others. 2025. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*.

Yunfan Shao, Linyang Li, Junqi Dai, and Xipeng Qiu. 2023. Character-llm: A trainable agent for role-playing. *arXiv preprint arXiv:2310.10158*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, and 1 others. 2024. Os-genesis: Automating gui agent trajectory construction via reverse task synthesis. *arXiv preprint arXiv:2412.19723*.

Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024a. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. *arXiv preprint arXiv:2406.01014*.

Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024b. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *arXiv preprint arXiv:2401.16158*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024c. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *Preprint*, arXiv:2409.12191.

Zhenhailong Wang, Haiyang Xu, Junyang Wang, Xi Zhang, Ming Yan, Ji Zhang, Fei Huang, and Heng Ji. 2025. Mobile-agent-e: Self-evolving mobile assistant for complex tasks. *arXiv preprint arXiv:2501.11733*.

Y. Wanyan, X. Zhang, H. Xu, and et al. 2025. Look before you leap: A gui-critic-r1 model for pre-operative error diagnosis in gui automation. *arXiv preprint arXiv:2506.04614*.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and 1 others. 2024. Os-atlas: A foundation action model for generalist gui agents. *arXiv preprint arXiv:2410.23218*.

Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. 2024. Aguvis: Unified pure vision agents for autonomous gui interaction. *arXiv preprint arXiv:2412.04454*.

Yi Yang, Xiaoxuan He, Hongkun Pan, Xiyan Jiang, Yan Deng, Xingtao Yang, Haoyu Lu, Dacheng Yin, Fengyun Rao, Minfeng Zhu, and 1 others. 2025. R1-onevision: Advancing generalized multimodal reasoning through cross-modal formalization. *arXiv preprint arXiv:2503.10615*.

Siyu Yuan, Kaitao Song, Jiangjie Chen, Xu Tan, Yongliang Shen, Ren Kan, Dongsheng Li, and Deqing Yang. 2024. Easytool: Enhancing llm-based agents with concise tool instruction. *arXiv preprint arXiv:2401.06201*.

Chi Zhang, Zhao Yang, Jiaxuan Liu, Yanda Li, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. 2025a. Appagent: Multimodal agents as smartphone users. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–20.

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

Zhong Zhang, Yaxi Lu, Yikun Fu, Yupeng Huo, Shenzhi Yang, Yesai Wu, Han Si, Xin Cong, Haotian Chen, Yankai Lin, and 1 others. 2025b. Agentcpm-gui: Building mobile-use agents with reinforcement fine-tuning. *arXiv preprint arXiv:2506.01391*.

Hengguang Zhou, Xirui Li, Ruochen Wang, Minhao Cheng, Tianyi Zhou, and Cho-Jui Hsieh. 2025. R1-zero's" aha moment" in visual reasoning on a 2b non-sft model. *arXiv preprint arXiv:2503.05132*.

A Overview of Appendix

We have over 5 pages of this appendix, comprising the following subsections for the convenience of readers:

- **Appendix B**: More details of all prompts.
- **Appendix C**: More details of the dataset.
- **Appendix D**: More details of action space.
- **Appendix E**: More details of training settings.

More experimental analysis

- **Appendix F**: Robustness analysis of Mobile-R1.
- **Appendix G**: Analysis of Task-level Training.

More visualization and cases

- **Appendix H**: Visualization of performance comparison among models.

We hope that our efforts will serve as a source of inspiration for more readers!

B Prompts

B.1 A.1 Prompts of Dataset Generation

The prompt used to generate instructions for execution by mobile agents, derived from Claude 3.5, is shown in Figure 7. The **bond** indicates the name of an app , which can be replaced by any app listed in Table 6.

Prompt for Instruction Generation

我在开发一个能熟练使用**淘宝**完成常见任务的代理。请帮我生成 10 个任务，对代理进行测试。要求任务要有具体的任务参数，比如商品、店铺、主播的名称，商品的数量和价格，衣服的颜色和尺码，评论的内容等等。
(I am developing an agent capable of proficiently completing common tasks on **Taobao**. Please help me generate 10 test tasks for evaluating the agent' s performance. Each task should include specific parameters, such as the name of the product, store, or live stream host, the quantity and price of the item, the color and size of clothing, the content of a review, and so on.)

Figure 7: Prompt for Instruction Generation.

B.2 Prompts of Mobile Agent

The prompt used to guide the Qwen2.5-VL to execute the instructions for trajectory collection. The **bond** indicates the instruction, which can be replaced by any instruction of our dataset.

Prompt for Agent Exploration

You are a mobile GUI agent. You are given a task and your action history, with screenshots. You need to perform the next action to complete the task.

You are provided with function signatures within <tools></tools> XML tags:
<tools>

```
{
  "name": "mobile_use",
  "arguments": {
    "type": "function",
    "function": {
      "name_for_human": "mobile_use",
      "name": "mobile_use",
      "description": "Use a touchscreen
      to interact with a mobile device."
    }
  }
}
```

<tools>

For each function call, return a json object with function name and arguments within <tool_call></tool_call> XML tags:
<tool_call>

```
{
    "name": <function-name>,
    "arguments": <args-json-object>
}
```

</tool_call>

Analyze the task and historical actions, and predict the next step.
Output your reasoning process within the <think></think> tag.
Output the action to be performed in this step within the <action></action> tag.
Output the final answer within the <tool_call></tool_call> tag.

User Task: **Open Taobao and find the store I am following.**

B.3 Prompts of Judge Model

As our experiments were conducted on Chinese applications, we accordingly prompted the scoring model in Chinese. The obtained scores were subsequently divided by 4 to normalize them into the $[0, 1]$ range. The prompt is shown in Figure 8.

Prompt of Judge Model

你是一位评估 GUI 代理任务轨迹的专家。你的任务是评估 GUI 操作任务轨迹的质量和有效性。一个轨迹包含以下组件：用户指令：描述用户的预期任务（例如，“打开淘宝，将‘iphone16 pro max’加入购物车”）。动作历史：包括两个关键部分：每一步的推理和动作：由代理执行的一系列动作，包括推理过程和最终执行的动作。GUI 截图：初始界面和在执行完每一步动作后的界面截图（从上到下依次排列）。
在评估轨迹时，请考虑以下关键方面：
评估标准：
轨迹连贯性：低级步骤和相应动作是否遵循朝向目标指令靠近？动作是否清晰描述且具体？是否存在冗余或不必要的动作？
任务完成情况：轨迹是否成功完成了指令任务？是否完成了所有必要的交互？错误情况是否得到适当处理？评分指南：
根据评估标准，按 0 到 4 的等级对轨迹进行评分：
4: 任务完美完成，成功执行多项动作实现目标。序列逻辑清晰且没有明显冗余。
3: 任务大部分完成，成功执行多项动作。然而，由于指令中的挑战或不确定性，完成情况不完美，或者过程存在效率低下。
2: 任务部分完成，执行了一些成功动作。然而，由于任务或环境限制，目标未完全实现，或者序列以循环或错误结束。
1: 仅执行了少量动作。虽然有完成任务的尝试，但轨迹早期偏离目标或在执行和逻辑上表现出显著低效。
0: 任务完全失败，开始时没有执行有意义的动作。序列要么立即陷入死锁、重复循环，或在完成任务上没有价值。或者任务完全不可访问。
注意：如果任务相对复杂，但轨迹表现出有价值的尝试，即使任务没有完全完成，也请考虑向上调整分数。然而，如果任务复杂但轨迹未能执行对任务完成有意义贡献的动作，则不应奖励额外分数。
您需要根据代理的动作和截图综合评估得分。
输出格式：

```
{
"reason": <your thoughts and reasoning
process for the score>,
"reward": <your score from 0-4>
}
```

一定不要输出多余内容，直接输出 json 格式的答案。

Figure 8: Prompt of Judge Model.

B.4 Prompts of Training System

This prompt was designed to ensure consistency with the Qwen2.5-VL's GUI prompt. The **bond** indicates the instruction, which can be replaced by any instruction of our dataset.

Prompts of Training System

You are a mobile GUI agent. You are given a task and your action history, with screenshots. You need to perform the next action to complete the task.

You are provided with function signatures within <tools></tools> XML tags:
<tools>

```
{
  "name": "mobile_use",
  "arguments": {
    "type": "function",
    "function": {
      "name_for_human": "mobile_use",
      "name": "mobile_use",
      "description": "Use a touchscreen
      to interact with a mobile device."
    }
  }
}
```

<tools>

For each function call, return a json object with function name and arguments within <tool_call></tool_call> XML tags:
<tool_call>

```
{
    "name": <function-name>,
    "arguments": <args-json-object>
}
```

</tool_call>

Analyze the task and historical actions, and predict the next step.
Output your reasoning process within the <think></think> tag.
Output the action to be performed in this step within the <action></action> tag.
Output the final answer within the <tool_call></tool_call> tag.

User Task: **Open Taobao, search for “iPhone 16 Pro Max", and add it to the shopping cart.**

C More Details of the Dataset

C.1 App Statistics from the Dataset

Our dataset includes 28 Chinese mobile applications, covering a wide range of daily-use scenarios. The detailed results are shown in Table 6.

Table 5: Definition of Atomic Action Space

Action	Definition
$key()$	Perform a key event on the mobile device, supporting adb's 'keyevent' syntax.
$click(x, y)$	Click the point on the screen with coordinate (x, y).
$swipe(x_1, y_1, x_2, y_2)$	Swipe from the starting point with coordinate (x_1, y_1) to the end point with coordinates2 (x_2, y_2).
$long\_press(x, y, time)$	Press the point on the screen with coordinate (x, y) for specified seconds.
$type(text)$	Input the specified text into the activated input box.
$system\_button(button)$	Press the system button, e.g., $Back, Home, Menu, Enter$.
$terminate(status)$	Terminate the current task and report its completion status, e.g., $success, failure$.
$wait(time)$	Wait specified seconds for the change to happen.

C.2 App Statistics from Open-Source Data

To facilitate the training and utilization of our dataset, focusing on Chinese mobile applications, we have open-sourced a sample of 1007 trajectories covering 28 different applications. Notably, for reasons related to data review, we have selected a portion of the data for open access. In the coming months, we will gradually organize and release additional data. The detailed results are shown in Table 7.

D Atomic Action Space

Table 5 presents all atomic operations considered in our framework. There are eight actions: key, click, swipe, long press, type, system button, terminate, and wait.

E Training Settings

Our experiments utilize Qwen2.5-VL-3B as the base model, with GRPO implementation (including for trajectory-level interaction training) adapted from the open-r1 framework.

For action-level training:

- Supervised Fine-Tuning (SFT): LoRA was applied for SFT, training for 2 epochs with a learning rate of 1×10^{-4}. The mini-batch size per device was set to 1, and a gradient accumulation number of 8 was used.
- GRPO Training: This phase involved 2 epochs of training with a learning rate of 1×10^{-7}. A mini-batch size of 1 per device and a gradient accumulation number of 2 were configured. To encourage exploration, the number of rollouts was set to 8, and the temperature was set to 1.

For trajectory-level training:

- We utilized two parallel virtual machine instances running locally to conduct experiments, enabling real-time interaction with the simulated environment.
- The number of rollouts was set to 4, and the maximum exploration steps per interaction was limited to 14. This allowed the model to explore and potentially backtrack historical operations upon error detection.
- Training was conducted for 2 epochs with a learning rate of 1×10^{-6}. A temperature of 1 was used to encourage exploration.
- A gradient accumulation number of 4 was applied, and due to the continuous real-time interaction with the virtual environment, a mini-batch size of 1 per device was maintained.
- Our Android emulator was configured to a Medium Phone device profile. To balance computational efficiency, each screenshot (window size W of Eq 4) fed to the model was downsampled by a factor of 2 in both dimensions.

F Robustness analysis of Mobile-R1

We performed a robustness analysis on 225 trajectories from unseen apps. The experimental results are shown in Figure 10. The following observations can be made: 1) All of the Models exhibits a noticeable decline in accuracy, indicating challenges in its robustness and generalization. 2) Mobile-R1 demonstrates the best performance in scores, emphasizing its enhanced generalization capabilities. This improvement is largely attributable to the crucial role of Stage 3 training, which significantly bolsters both robustness and adaptability.

Figure 9: Comparison of the thinking processes of the Stage2 model and Stage3 model of Mobile-R1 under the same task "*Open Fliggy, enter the hotel package, enter the popular live broadcast, find Fliggy Super VIP, and follow the anchor.*"

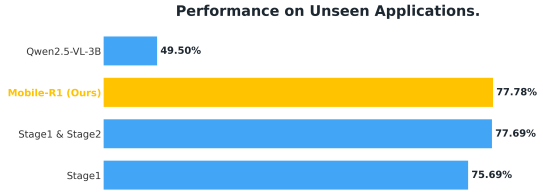

Figure 10: Robustness Analysis on Unseen Applications.

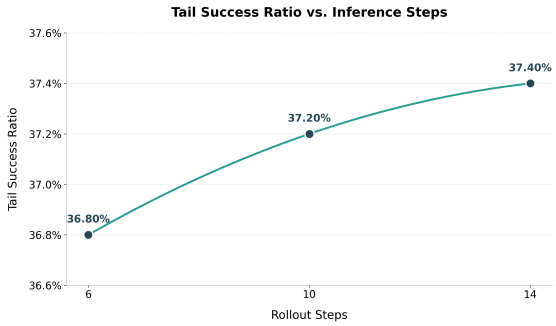

Figure 11: Tail success rate on different inference steps.

We further investigated the impact of varying the number of training steps in Stage 3. Specifically, we tracked the model's tail success ratio as a function of the training steps. As illustrated in Figuree 11, the tail success ratio exhibits a consistent upward trend with the increase in steps. This finding suggests that a more prolonged phase of end-to-end exploration is beneficial for achieving final task success.

To comprehensively unveil the upper bounds and maximum potential of our model's performance, we studied how model accuracy changes with pass@k — a metric that defines the probability that at least one of the k attempts successfully solves the given problem. We tested our models on 50 randomly sampled complete trajectories (185 actions) from our test set. During inference, we used a temperature of 0.7. Figure 12 shows both models' accuracy increases significantly as k grows, highlighting their ability to solve problems with multiple attempts. Notably, our Mobil-R1 model exhibited superior performance compared to the baseline model, achieving an absolute accuracy improvement of up to 7 percentage points.

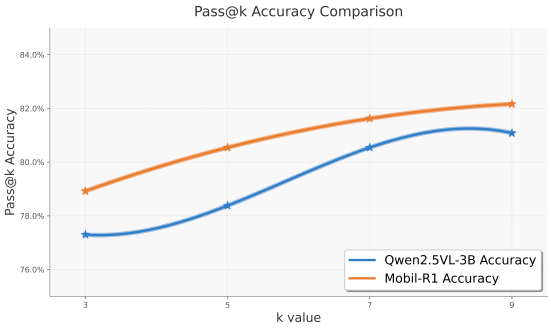

Figure 12: Pass K performance of Mobile-R1.

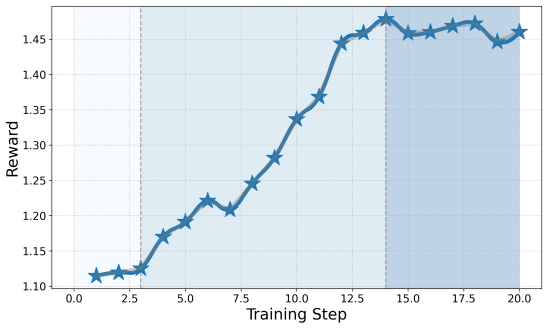

Figure 13: Reward score during Stage3 training.

G Analysis of Task-level Training

The reward score of Stage3 shown in Figure 13, reveals slow reward growth in the first three training

steps, likely due to instability caused by aggressive exploration or policy changes. However, between steps 4 and 14, the reward growth accelerates, suggesting effective learning. Subsequently, from step 15 onward, the growth stabilizes, indicating that the policy is gradually converging.

Overall, Figure 13 shows that despite the initial instability induced by aggressive exploration, the policy successfully stabilizes and converges within two epochs. This two-epoch schedule is therefore a deliberate trade-off that balances exploration depth against computational cost, and the observed plateau after step 15 justifies stopping there rather than training for additional epochs.

H More Qualitative Results

Figure 9 demonstrates the comparison of the thinking processes between the Stage2 model and Stage3 model of Mobile-R1 under the same task. At the final step of the task, Mobile-R1 accurately recognized that the tab was already marked as "Followed", eliminating the need for further actions and thus concluding the task with superior contextual awareness. In contrast, the Stage 2 model still attempted to click on the "Feizhu Super VIP" tab, demonstrating a lack of awareness of the task's completion status.

Notably, this Eureka moment does not require revisiting earlier screenshots: the "Followed" cue that triggers termination is fully contained in the Step 4 screenshot, while the memory of prior actions is carried via the textual trajectory. This is a concrete instance of the two dependency sources identified in §4.4, and explains why self-correction remains effective under a bounded visual window W.

Figure 14 demonstrates the successful trajectory of the Mobile-R1 agent in completing the task "*Open Taobao and find the followed shops.*"

Figure 15 demonstrates the successful trajectory of the Mobile-R1 agent in completing the task "*Download and install Honor of Kings.*"

I Human Validation of the GPT-4o Judge

To verify that GPT-4o is a trustworthy trajectory-level judge for Stage 3, we performed a blind human evaluation.

Setup. We randomly sampled 30 trajectories previously scored by GPT-4o, stratified across the five rubric levels (0–4; see Appendix B.3). Three human annotators — blind to the GPT-4o scores — independently re-scored each trajectory using the *same* rubric, viewing only the screenshots, thinking traces, and actions. The annotator majority label was taken as the human score.

Results. Exact-level agreement was $21/30$ (70.0%), and the remaining 9 cases each differed by exactly one level, so $|\Delta| \leq 1$ agreement reached $30/30$ (100%) with a mean absolute deviation of only 0.30 levels. The overall ranking was well preserved: Spearman $\rho = 0.92$ and Kendall $\tau_b = 0.86$, with a quadratic-weighted Cohen's $\kappa = 0.92$ on the ordinal five-level rubric.

Discussion. These results support the reasonableness of using GPT-4o as the trajectory-level judge. Importantly, Stage 3 uses the *group-normalized* advantage in Eq. (1): absolute scoring errors are partially cancelled by the group mean, so what primarily drives the policy gradient is the *relative* ordering within a rollout group, which the high rank correlation directly validates. Small boundary-level discrepancies therefore have limited impact on the learning signal. Three design choices further stabilize the judge: (i) the five-level rubric reduces scoring variance; (ii) the prompt (Fig. 8) fixes two evaluation dimensions — *Trajectory Coherence* and *Task Completion* — to mitigate drift; and (iii) scores are re-normalized to $[0, 1]$ before being used as rewards.

J All Resources

We are committed to releasing all our resources, including the dataset, model weights, and framework implementation, which are slated for public release, to foster reproducibility. The project homepage is available at: https://mobile-r1.github.io/Mobile-R1/. To adhere to the double-blind review process, the resource links on the page are currently placeholders. They will be made fully public upon acceptance of the paper.

Figure 14: Qualitative result of Mobile-R1 under the task “*Open Taobao and find the followed shops.*"

Figure 15: Qualitative result of Mobile-R1 under the task “*Download and install Honor of Kings.*"

Table 6: Statistics from Dataset

#	App	Data	Trajectory
1	Alipay	5	1
2	Amap	446	93
3	App Store	16	4
4	Baidu	439	106
5	Baidu Maps	1844	308
6	Bilibili	2457	376
7	Browser	213	39
8	Calculator	443	56
9	Calendar	390	69
10	Dianping	4	1
11	Douyin	447	106
12	Eleme	538	129
13	Fliggy	2934	593
14	Idle Fish	29	5
15	JD	1092	211
16	Kuaishou	2938	619
17	Luckin Coffee	144	18
18	Meituan	520	93
19	Mobile System	38	7
20	Notes	110	19
21	Pinduoduo	707	115
22	Quark	390	67
23	Taobao	4326	821
24	Tencent Maps	912	177
25	WeChat	33	6
26	Weather	396	91
27	Xiaohongshu	2706	504
28	Zhuanzhuan	4	1
	Total	24,521	4,635

Table 7: Statistics from Open-Source Data

#	App	Data	Trajectory
1	Alipay	3	1
2	Amap	227	50
3	App Store	11	4
4	Baidu	160	52
5	Baidu Maps	191	50
6	Bilibili	148	55
7	Browser	174	39
8	Calculator	195	50
9	Calendar	162	50
10	Dianping	4	1
11	Douyin	152	51
12	Eleme	193	50
13	Fliggy	191	50
14	Idle Fish	27	5
15	JD	200	50
16	Kuaishou	136	51
17	Luckin Coffee	97	18
18	Meituan	252	50
19	Mobile System	7	3
20	Notes	64	19
21	Pinduoduo	180	51
22	Quark	181	50
23	Taobao	221	50
24	Tencent Maps	231	50
25	WeChat	33	6
26	Weather	159	50
27	Xiaohongshu	296	50
28	Zhuanzhuan	4	1
	Total	3,924	1,007