Does Chain-of-Thought Reasoning Help Mobile GUI Agents? An Empirical Study

Li Zhang[*,1,2], **Longxi Gao**[*,1], **Mengwei Xu**[1]
[1]Beijing University of Posts and Telecommunications
[2]Tsinghua University
{li.zhang,glx,mwx}@bupt.edu.cn

Abstract

Reasoning capabilities have significantly improved the performance of vision-language models (VLMs) in domains such as mathematical problem-solving, coding, and visual question-answering. However, their impact on real-world applications remains unclear. This paper presents a large-scale empirical study on the effectiveness of reasoning-enabled VLMs in mobile GUI agents. We evaluate six pairs of VLMs, including both commercial and open-source lightweight models, by comparing their base and reasoning-enhanced versions across static and interactive benchmarks. Our findings show that, under the evaluated benchmarks and configurations, reasoning-enabled VLMs generally provide only marginal improvements over their non-reasoning counterparts and can even degrade performance in certain agent configurations. Notably, reasoning and non-reasoning VLMs fail on different sets of tasks, suggesting that reasoning does have an impact, but its benefits and drawbacks counterbalance each other. We attribute these inconsistencies to the limitations of benchmarks and VLMs. Based on the findings, we provide insights for further enhancing mobile GUI agents in terms of benchmarks, VLMs, and their adaptability in dynamically invoking reasoning VLMs.

1 Introduction

Reasoning capabilities significantly enhance large language models (LLMs) and vision-language models (VLMs) by utilizing long chain-of-thought (CoT) thinking and extended test-time computation (Snell et al., 2024; Wei et al., 2022). Empirical evidence from recent studies demonstrates that such enhanced reasoning abilities yield superior performance in domains like mathematical problem-solving, coding, and visual question answering (Snell et al., 2024; Guo et al., 2025; Wang et al., 2024). These models with reasoning capabilities have established new benchmark records in their respective fields, surpassing previous LLMs/VLMs that lack reasoning.

Despite these advancements, the complexities inherent in real-world applications pose significant challenges. *Does reasoning help with real-world complex multimodal tasks, beyond coding and math?* In this study, we focus on a practical, unsolved task, i.e., mobile GUI agents, particularly for mobile device control tasks (Wen et al., 2024a; Rawles et al., 2023, 2024; Wen et al., 2024b; Xu, 2025; Gao et al., 2024), which present a unique testbed due to their intricate visual layouts, diverse functionalities, and the requirement for multi-step reasoning and interaction to achieve user goals. Existing state-of-the-art (SOTA) mobile GUI agents without reasoning still struggle to deliver satisfactory and practical success rates in real-world environments (Wen et al., 2024b; Rawles et al., 2024; Simular, 2025). We hypothesize that incorporating reasoning ability, similar to its application in other domains, could potentially enhance the performance of mobile GUI agents by improving task comprehension, environmental adaptation, and action decision-making. Therefore, evaluating the effectiveness of reasoning VLMs in this demanding downstream task is of critical importance.

Methodology and Experiments. This study fills the existing gap by conducting a comprehensive empirical evaluation of reasoning VLMs in mobile GUI agents. Specifically, we select six pairs of VLMs with and without reasoning capabilities, including (1) two commercial VLMs: Gemini 2.0 Flash (Google, 2025) and Claude 3.7 Sonnet (Anthropic, 2025a), and (2) four open-source VLMs: AgentCPM-GUI-8B (Zhang et al., 2025b), GLM-4.5V (GLM-V Team, 2026), Qwen3-VL-4B, and Qwen3-VL-8B (Bai et al., 2025). We select two static benchmarks, AndroidControl (Li et al., 2024a) and ScreenSpot (Cheng et al., 2024),

[*]Authors contributed equally to this work. Corresponding author: Mengwei Xu.

Findings of the Association for Computational Linguistics: ACL 2026, pages 7981–7996

and one interactive testbed AndroidWorld (Rawles et al., 2024). For each benchmark, we implement and test different agent setups upon the VLMs.

Results and Findings. Through experiments and analysis, we make the following key observations.

(1) On static benchmarks, reasoning VLMs generally have marginal improvements over non-reasoning VLMs, and even suffer severe performance degradation under certain agent setups. For instance, in AndroidControl, Gemini Thinking achieves an average action prediction accuracy of 54.4%, only 0.8% higher than the non-reasoning version. In grounding tasks within ScreenSpot, performance improvements are observed only with Claude Thinking with normalized center-point output; in other setups, accuracy drops by up to 29.7%.

(2) On the interactive mobile testbed AndroidWorld, Claude Thinking achieves a 64.7% task completion rate with set-of-mark prompting, and is 6.3% higher than the non-reasoning version. This highlights the effectiveness and potential of reasoning VLMs in real-world mobile GUI automation tasks. Nonetheless, Gemini 2.0 Flash and Qwen3-VL series reasoning VLMs exhibit slight performance drops compared to their base variant, indicating that improvements are model-specific.

(3) Surprisingly, the reasoning and non-reasoning VLMs fail on a substantially different set of test cases. For example, Gemini Thinking fails on 36%, 9%, and 12% of tasks in ScreenSpot, AndroidControl, and AndroidWorld, respectively, that Gemini can successfully accomplish. Vice versa, Gemini Thinking also succeeds up to 10% of tasks that Gemini fails. This suggests that the lack of accuracy improvement at the benchmark level is not because reasoning has no effect, but rather its positive and negative impacts counterbalance each other. These inconsistencies emphasize the need for a deeper investigation into the role of reasoning in mobile GUI agents.

(4) Our manual investigation of the reasoning process reveals that errors in reasoning VLMs stem from limitations in both mobile GUI agent benchmarks and the underlying VLMs. We find that reasoning VLMs exhibit similarities to human thought processes when operating smartphones. However, this advanced understanding does not translate into performance gains due to inherent benchmark limitations, such as vague task instructions and the inability to evaluate multiple possible actions within static datasets. Furthermore, during the reasoning phase, VLMs sometimes fail to comprehend screen details accurately and may generate responses that are inconsistent with the reasoning processes.

(5) Reasoning VLMs significantly increase model output tokens by at least 3.11× and up to 14.78×, leading to higher response latency and monetary costs without clear performance benefits. As observed in ScreenSpot, the average number of output tokens increases from 37.6 to 238.5. Without strict output constraints, reasoning VLMs may generate additional tokens in their final responses, e.g., to summarize their thought process. This raises costs and practicality concerns regarding the indiscriminate use of reasoning VLMs for all mobile GUI agent tasks.

Implications. We derive several implications for enhancing mobile GUI agents. (1) Mobile GUI agents with reasoning VLMs are better to be evaluated on interactive testbeds, instead of static benchmarks. This could avoid the intrinsic limitations of static benchmarks. (2) The underlying VLMs should be specifically trained for mobile GUI agents to improve grounding and screen comprehension at the reasoning phase. It is also crucial to maintain consistency from reasoning to final outputs. (3) Resource efficiency will become a major obstacle toward reasoning-enhanced mobile GUI agent, due to the excessive task completion latency and token expense. Efficient reasoning is critical to a practical reasoning-enhanced mobile GUI agent.

Contributions. The contributions of this study are summarized as follows. (1) We conduct the first empirical study of VLMs' reasoning capabilities in mobile GUI agents, a critical downstream task focused on automatic smartphone control. (2) We demonstrate the limited performance gains from reasoning VLMs and highlight their limitations, particularly in failing tasks that non-reasoning VLMs can successfully complete. (3) We perform an in-depth error analysis of the reasoning process, categorizing errors based on VLM limitations and benchmark constraints. Our findings provide valuable insights for advancing future research in this area.

2 Background

2.1 Mobile GUI Agents

From API-based agents to GUI agents. Traditional mobile agents, like Apple Siri (Apple, 2024) and Google Assistant (Google, 2024), relied on static, API-driven interactions. These agents op-

erated based on predefined rules and could only automate tasks with exposed APIs, therefore limiting their adaptability. Recently, leveraging the advancements in LLMs and VLMs, modern mobile GUI agents have shifted from API-dependent automation to direct interpretation and operation on mobile screens (Wen et al., 2024a; Wang et al., 2023; Li et al., 2024b; OpenAI, 2025; Anthropic, 2024; Zhang et al., 2024a; Gao et al., 2025; Zhang and Xu, 2025). Instead of being restricted by predefined API calls, these agents analyze screen contents, user instructions, and execute actions based on visual and textual information, making them more adaptable to various unseen tasks and applications (Zhang et al., 2025a).

Prior studies have shown that automating mobile GUI tasks is challenging, especially in real-world settings (Rawles et al., 2024; Zhang et al., 2024c). Unlike API-based agents, GUI agents must handle diverse screen layouts, ambiguous elements, and dynamic content, often resulting in inconsistent performance. Despite progress in LLM/VLM reasoning for tasks like math and programming, its potential in mobile GUI automation remains underexplored. This study aims to examine whether incorporating reasoning can improve task completion in complex and dynamic mobile environments.

2.2 Reasoning LLMs/VLMs

To enhance reasoning capabilities for solving more complex tasks, OpenAI's o1 series models (OpenAI, 2024) became the first commercial models to adopt the test-time scaling technique in 2024. These models follow a "think first, then answer" approach. By allocating more computation to this stage, these models produce more accurate final answers, as demonstrated by their performance in solving complex mathematical, coding, and multimodal reasoning problems (Wang et al., 2024; Guo et al., 2025; Snell et al., 2024).

We deem **multimodal reasoning to be essential for mobile GUI agents**, as they encounter unseen and complex tasks within mobile apps. These complexities increase in dynamic and unpredictable mobile contexts, such as frequently changing app content and intermittent network conditions. Allocating time for reasoning allows mobile GUI agents to adapt to environmental changes, correct their own mistakes, and ultimately determine the optimal path for completing a GUI automation task (Anthropic, 2025b). With the reasoning process, we observe that mobile GUI agents develop a comprehensive understanding of both the current GUI state and the task instruction, leading to confident and accurate actions. This deeper understanding is crucial for handling complex and dynamic mobile environments.

Despite these promising outcomes, very few studies have explored how such reasoning processes benefit mobile GUI agents (Zhang and Zhang, 2024; Zhang et al., 2024b; Dong et al., 2025). Existing research primarily uses CoT prompting in highly controlled environments. This study aims to bridge this gap by conducting a large-scale, comprehensive investigation into whether reasoning improves mobile GUI agent performance in real-world scenarios using VLMs with intrinsic reasoning capabilities.

3 Methodology

In this section, we describe the methodology employed in this empirical study.

Benchmarks. We incorporate both representative static and interactive benchmarks as follows.

(1) ScreenSpot (Cheng et al., 2024) is a GUI grounding dataset with more than 600 GUIs and over 1.2K task instructions. A grounding task is defined as: given a task instruction, the VLM identifies the corresponding GUI component to be acted upon, and outputs its coordinates. We use the "mobile" subset within ScreenSpot.

(2) AndroidControl (Li et al., 2024a) is a static dataset proposed by Google and contains more than 14K tasks across 800+ Android apps. In this study, we follow the experimental setup and evaluation approach used in AndroidControl.

(3) AndroidWorld (Rawles et al., 2024) is an interactive mobile GUI agent benchmark proposed by Google. It uses predefined function calls to access internal app states for task completion verification, enabling a more accurate evaluation. We use its 116 tasks to demonstrate capabilities of GUI agents in real-world scenarios.

Models and Agents. Models combined with curated prompts form mobile GUI agents. We use two pairs of commercial VLMs–Gemini 2.0 Flash (Google, 2025) and Claude 3.7 Sonnet (Anthropic, 2025a)–each including a base model without reasoning and its reasoning-enabled variant.[1] We also include four smaller open-source

[1]Gemini base model: *gemini-2.0-flash-001*, reasoning model: *gemini-2.0-flash-thinking-exp-01-21*. Claude base model: *claude-3-7-sonnet-20250219*, reasoning model: *claude-3-7-sonnet-20250219* with thinking mode enabled and

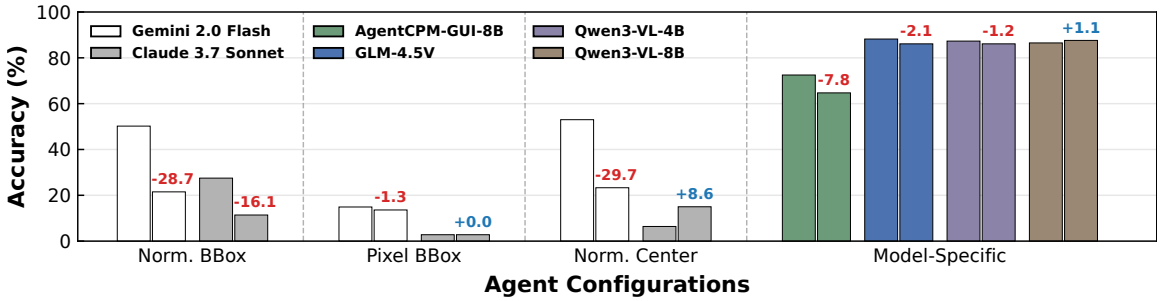


Figure 1: Mobile GUI grounding accuracy of different VLM/prompt designs on ScreenSpot. Each same-color bar pair: first is non-reasoning, second is reasoning.

VLMs trained specifically for mobile GUI tasks: AgentCPM-GUI-8B (Zhang et al., 2025b), GLM-4.5V (GLM-V Team, 2026), Qwen3-VL-4B (Bai et al., 2025), and Qwen3-VL-8B (Bai et al., 2025). In addition to these six primary pairs, we incorporate a recently released, locally deployable model, GLM-4.6V (Z.ai, 2025), specifically to evaluate the impact of strictly constrained reasoning token budgets. The mobile GUI agents in this study are built on top of these VLMs but differ in their prompting strategies. For ScreenSpot, we instruct the agent to output three different formats for a grounded GUI element: (1) normalized bounding box *(e.g., [0.08, 0.688, 0.92, 0.735])*; (2) pixel-based bounding box *(e.g., [127, 34, 235, 978])*; and (3) normalized center point *(e.g., [255, 370])*. For AndroidControl, we use the ER prompt (Li et al., 2024a), which takes the task instruction and previous action list as input. We further modify its input to get three variants: (1) task instruction only; (2) step instruction only; and (3) task and step instructions. For AndroidWorld, we employ three agent designs: (1) M3A with set-of-mark prompting (Yang et al., 2023); (2) M3A with accessibility tree (a11y tree) prompting; and (3) T3A with a11y tree prompting. For the open-source VLMs, we prefer using their model-specific prompts, when available, to better align with their training data.

Metrics. On static benchmarks, we report grounding accuracy for ScreenSpot and action prediction accuracy for AndroidControl. The evaluation method for AndroidControl follows the approach detailed in its original work (Li et al., 2024a). For AndroidWorld, we assess end-to-end task completion rates. During experiments, we log all traces, including model responses, reasoning processes, and screenshots, for token count and error analysis.

a budget token number of 1024.

4 Experimental Results

In this section, we first report task completion accuracy (§4.1), followed by a task-level analysis to assess the impact of reasoning (§4.2). We then categorize reasoning-related errors (§4.3) and quantify the token cost of reasoning-enabled VLMs (§4.4). We discuss key insights for improving reasoning-capable GUI agents in §4.5.

4.1 Benchmark-level Analysis

Static benchmarks. The results presented in Figure 1 and Figure 2 reveal a trend: *reasoning VLMs generally do not improve, or trivially improve the performance of mobile GUI agents on static benchmarks.* In some agent setups, it even leads to a significant performance drop.

Specifically, in ScreenSpot (Cheng et al., 2024), we evaluate GUI grounding accuracy across different VLMs and grounding output formats. As shown in Figure 1, reasoning generally degrades grounding accuracy when using normalized and pixel-based bounding boxes across VLMs. Gemini Thinking and Claude Thinking exhibit substantial accuracy reductions (28.7% and 16.1%, respectively, for normalized bounding boxes), indicating a negative impact on this grounding task. However, for normalized center points and model-specific prompting designs, the effect of reasoning is mixed. For example, while Gemini Thinking and AgentCPM Thinking show notable declines in grounding accuracy (29.7% and 7.8%, respectively), Claude Thinking improves by 8.6%. These results suggest that the effectiveness of reasoning is highly dependent on both the model and the task. In AndroidControl, as shown in Figure 2, reasoning VLMs yield only a marginal geometric mean improvement of 0.75% in accuracy across all configurations. Overall, our evaluation across two distinct static GUI benchmarks indicates that

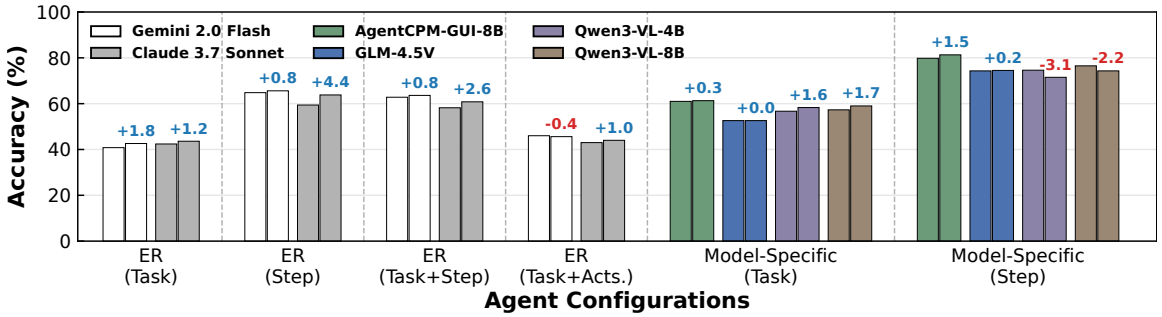


Figure 2: Action prediction accuracy of different VLM/agent on AndroidControl. Each same-color bar pair: first is non-reasoning, second is reasoning.

incorporating reasoning VLMs into mobile GUI agents does not consistently enhance performance and, in some cases, may even hinder it.

Interactive testbed. We then use AndroidWorld as an interactive testbed to evaluate mobile GUI agents, along with three different agent setups proposed in their study (Rawles et al., 2024). The results in Figure 3 indicate that different model pairs exhibit distinct behaviors. With reasoning enabled in Gemini, performance drops by an average of 2.7%, demonstrating its non-positive impact on task completion rates. In contrast, Claude Thinking enhances the performance with an average improvement of 6.3%. Notably, Qwen3-VL-4B exhibits the largest performance regressions when reasoning is enabled, consistent with the repetitive reasoning behavior observed by us and in recent studies (Pipis et al., 2025).

We further analyze task completion rates categorized by difficulty levels in AndroidWorld. Table 1 shows that enabling reasoning in VLMs produces inconsistent effects that vary with both the VLM and the agent variant. Performance on hard tasks often remains flat with reasoning enabled and can even decline in some cases—for example, Gemini 2.0 Flash and GLM-4.5V show drops on T3A (a11y tree)—indicating that reasoning does not consistently improve outcomes on complex interactive tasks. The Qwen3-VL series generally performs better without reasoning enabled under the prompting design used in this study, despite prior reports claiming superior performance for the reasoning variants (Bai et al., 2025). This discrepancy suggests that the effectiveness of reasoning depends heavily on model-specific prompting strategies and output formats, and that limitations on hard tasks remain a key bottleneck for generalized mobile GUI agents.

4.2 Task-level Analysis

From the prior results, we conclude that reasoning VLMs do not benefit mobile GUI agents in static benchmarks, as they typically achieve comparable performance regardless of whether reasoning is enabled. Their improvement in AndroidWorld is model-specific but not substantial. Moreover, the reported performance is based on the entire dataset, without assessing the impact of reasoning capabilities on individual tasks. In this section, we conduct a deeper analysis of individual tasks within each benchmark to determine whether reasoning VLMs enhance or hinder mobile GUI agent performance.

We focus on two categories of tasks. (1) Tasks that cannot be completed by non-reasoning models but are successfully completed with reasoning VLMs (**F**→**T** in Table 2). These tasks demonstrate the advantages of reasoning VLMs in improving mobile GUI agents. (2) Tasks that can be completed with non-reasoning VLMs but fail in reasoning VLMs (**T**→**F** in Table 2). These tasks highlight potential limitations of current reasoning VLMs, which may halt their integration into existing mobile GUI agents. We focus on the task execution results of the two commercial VLM series that have overlapped agent designs.

Result inconsistency after the adoption of reasoning VLMs. Our results are presented in Table 2 with the following observations. First, applying reasoning to previously successful tasks introduces a substantial number of inconsistencies, which undermines the accuracy achieved by mobile GUI agents in the non-reasoning mode across most benchmarks and experimental setups. For example, on ScreenSpot with normalized bounding-box output, Gemini and Claude fail 36% and 18% of tasks after reasoning, respectively, even though having successfully completed these tasks without reasoning. Similarly, in AndroidControl, Gemini

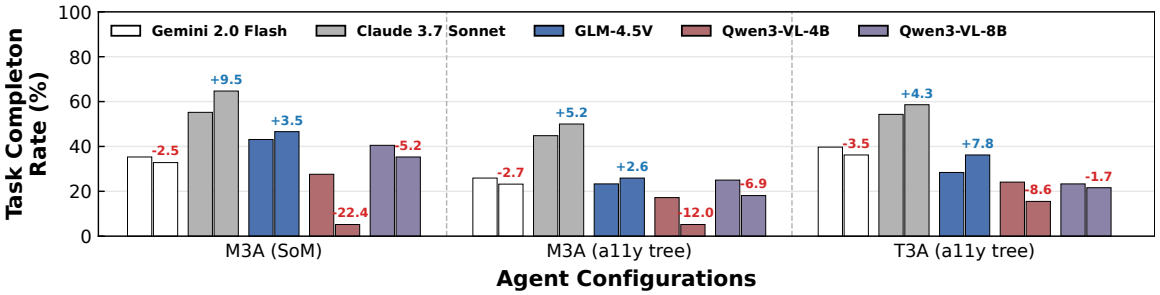


Figure 3: Task completion rates with different agent designs on AndroidWorld. Each same-color bar pair: first is non-reasoning, second is reasoning. AgentCPM-GUI-8B is omitted due to its inability to follow instructions.

VLM	M3A (SoM)			M3A (a11y tree)			T3A (a11y tree)		
	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard
Gemini 2.0 Flash	0.29	0.15	0.10	0.28	0.07	0.03	0.33	0.18	0.17
+ Thinking	0.31	0.13	0.07	0.18	0.09	0.03	0.35	0.15	0.07
Claude 3.7 Sonnet	0.40	0.45	0.16	0.42	0.29	0.10	0.53	0.26	0.16
+ Thinking	0.56	0.46	0.16	0.40	0.31	0.10	0.50	0.34	0.17
GLM-4.5V	0.54	0.42	0.11	0.33	0.17	0.05	0.39	0.17	0.16
+ Thinking	0.59	0.39	0.21	0.36	0.22	0.05	0.54	0.19	0.11
Qwen3-VL-4B	0.36	0.25	0.05	0.25	0.11	0.05	0.36	0.17	0.11
+ Thinking	0.10	0.03	0.00	0.10	0.03	0.00	0.23	0.06	0.11
Qwen3-VL-8B	0.52	0.36	0.11	0.34	0.17	0.11	0.38	0.11	0.11
+ Thinking	0.46	0.31	0.16	0.21	0.19	0.05	0.31	0.08	0.21

Table 1: Task completion rates on AndroidWorld categorized by task difficulties.

Thinking fails an average of 37 tasks, while Claude Thinking fails 16 tasks. These results indicate that the reasoning process in current VLMs significantly reduces accuracy on tasks that they could otherwise complete without reasoning.

Second, for tasks that are impossible to complete by non-reasoning models, reasoning provides a moderate improvement. For instance, Gemini Thinking achieves average improvements of 7.7%, 8.1%, and 8.8% across the three benchmarks, respectively. However, in most cases–except for Claude Thinking, which shows significant improvement in AndroidWorld with M3A-SoM–these accuracy gains do not compensate for the overall reduction in task completion rates caused by the reasoning process. Thus, existing reasoning VLMs may have a slightly negative impact on mobile GUI agents.

4.3 Error Analysis

We then take a deeper look at the tasks where reasoning VLMs lead to shifts from *completed* to *failed*, aiming to contrast reasoning and non-reasoning VLMs. In ScreenSpot, we find that approximately all errors are attributed to incorrect grounding coordinate outputs. This suggests a significant limitation in the grounding capability of the current VLM reasoning process, which is a key functionality required by mobile GUI agents.

However, in another static benchmark, AndroidControl, grounding errors nearly disappear due to a better mobile GUI agent design. By incorporating view hierarchies (which include bounding boxes for each GUI element) alongside screenshots as input, mobile GUI agents can more precisely extract the coordinates of GUI elements during reasoning. Nevertheless, we also observe a large number of errors causing mobile GUI agents to fail in previously successful tasks under non-reasoning modes.

We combine all tasks with T→F under all setups using Claude and Claude Thinking, manually identify the errors, and then categorize them based on their sources: *benchmark* and *VLM*. We present different errors within each category across a total of 67 tasks, along with their explanations and percentage distributions, in Table 3. All tasks were executed on Claude models, as the API provides comprehensive reasoning processes for our diagnosis, whereas Gemini Thinking's API does not yet support this functionality.

• **Benchmark** contributes to more than 70% errors. The most significant portion of errors (47.8%) stems from the "Weak Evaluation Method", where

Benchmark	Setup	Gemini 2.0 Flash				Claude 3.7 Sonnet			
		T→F	*F→T*	*T→T*	*F→F*	*T→F*	*F→T*	*T→T*	*F→F*
ScreenSpot	Norm. BBox	36.06%	7.37%	14.14%	42.43%	17.93%	1.79%	9.56%	70.72%
	Pixel BBox	11.55%	10.16%	3.39%	74.90%	0.40%	0.40%	2.39%	96.81%
	Center Point	35.26%	5.58%	17.73%	41.43%	0.99%	9.56%	5.38%	84.06%
AndroidControl	Task Inst.	8.42%	10.22%	32.46%	48.9%	3.4%	4.6%	39.0%	53.0%
	Step Inst.	5.0%	5.8%	59.8%	29.4%	1.0%	5.4%	58.4%	35.2%
	Task Inst. + Step	6.8%	7.6%	56.0%	29.6%	3.2%	5.8%	55.0%	36.0%
	Task Inst. + Act.	9.2%	8.8%	36.8%	45.2%	5.8%	6.8%	37.2%	50.2%
AndroidWorld	M3A-SoM	12.17%	10.43%	22.61%	54.78%	0.86%	10.43%	54.31%	34.48%
	M3A-A11y Tree	10.43%	6.09%	15.56%	67.83%	6.03%	11.21%	38.79%	43.97%
	T3A-A11y Tree	12.28%	9.65%	27.19%	50.88%	5.17%	9.48%	49.14%	36.21%

Table 2: Task completion statistics (% of all tasks) across benchmarks and task setups with reasoning and non-reasoning VLMs. **T→F**: Tasks completed in non-reasoning mode but failed in reasoning mode; **F→T**: Tasks failed in non-reasoning mode but completed in reasoning mode; **T→T**: Tasks completed in both modes; **F→F**: Tasks failed in both modes. A high T→F value indicates a negative impact of the reasoning process; a high F→T value indicates a positive impact.

Error Source	Error Type	Explanation	Percentage	Example
Benchmark	Weak Evaluation Method	Various false negative actions may complete a task	47.8%	Fig. 5
	Static GUI Input Limitation	GUI agents receive only static, individual mobile GUIs	14.9%	Fig. 6
	Unclear Task Instruction	Vague or ambiguous task instructions	11.9%	Fig. 7
VLM	Limited GUI Comprehension	Unable to fully understand the GUI context	10.5%	Fig. 8
	Reasoning-Response Inconsistency	Correct reasoning process but inconsistent response	8.9%	Fig. 9
	Others	Incorrect grounding, incorrect reasoning, and hallucination	6.0%	Fig. 10/ 11/ 12

Table 3: AndroidControl error analysis for tasks completed by Claude without reasoning but failed with reasoning enabled (i.e., **T→F** in Table 2). Examples of each error are illustrated in Appendix A.1.

various correct actions that could continue or complete a task are erroneously evaluated as incorrect. This is a common limitation of static benchmarks and has been noted in previous studies (Zhang et al., 2024c; Rawles et al., 2024; Im et al., 2025). Another major issue (14.9%) is the "Static GUI Input Limitation". Since the benchmark feeds only one GUI at a time, the reasoning VLM struggles to determine whether prior states satisfy the requirements of a given task instruction. After reasoning, it may attempt to revert and check whether the prior condition was met, leading to incorrect outputs compared to the benchmark. Additionally, some task instructions within the benchmark are unclear, making them difficult for even humans to understand, and thus unsuitable for mobile GUI agents.

- **VLM.** The remaining 25.4% of errors stem from limitations in current reasoning VLMs. The most significant one is "Limited GUI Comprehension", where during the reasoning phase, the VLM misinterprets the GUI context and generates incorrect responses. More critically, even if the VLM deduces the correct output during reasoning, it may produce an inconsistent final response. As illustrated by the phenomenon of "reasoning-response inconsistency" (see Figure 9 for a detailed trace), the model's internal logic often disconnects from its final action generation. These inconsistencies further downgrade the performance. Additionally, we observe a few grounding errors, reasoning errors, and hallucinations after applying reasoning. Demonstrations of these errors can be found in Appendix A.1.

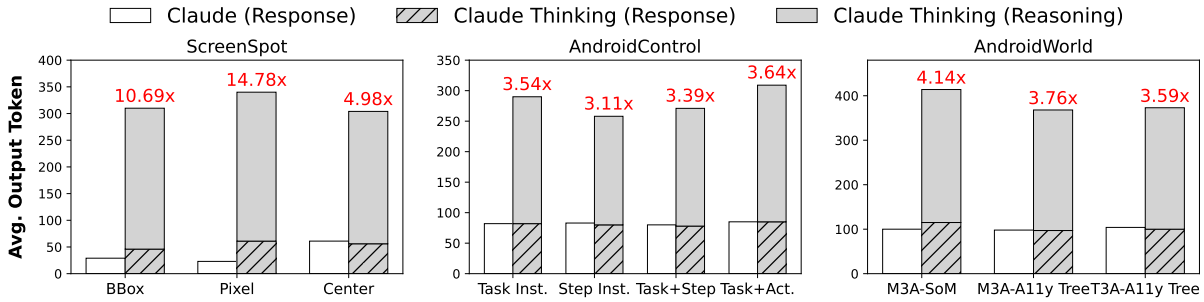


Figure 4: Comparison of average output token count between the Claude reasoning model and its base model without reasoning. Reasoning increases token consumption by at least 3× compared to the non-reasoning model, resulting in higher monetary costs and increased response latency.

4.4 Token Costs and Reasoning Budgets

Another concern regarding the integration of reasoning VLMs in mobile GUI agents is their high latency and substantial token costs during the reasoning process. To quantitatively assess this issue, we calculate and compare the number of model output tokens across all benchmarks and agent setups, both with and without reasoning enabled. We focus particularly on Claude 3.7 Sonnet, which returns its full thinking trace, whereas other proprietary models may expose only summarized or limited reasoning tokens. For its thinking mode, we accumulate the number of tokens generated during both the reasoning process and the final responses. The results in Figure 4 show that across all three benchmarks and setups, the reasoning process incurs at least 3.11× the token cost, with a maximum increase of 14.78×. More specifically, on ScreenSpot, the average number of output tokens without reasoning is 37.6, whereas enabling reasoning increases this value to 238.5. This significantly raises both token costs and response time, although prior results indicate no considerable performance improvements. These findings highlight an important question: when should mobile GUI agents leverage advanced reasoning VLMs to enhance performance while maintaining acceptable latency and monetary costs? Another observation is that on AndroidControl and AndroidWorld, the number of final response tokens remains identical. However, on ScreenSpot, the reasoning model generates additional information to summarize its reasoning process, resulting in a higher number of response tokens. This phenomenon stems from weak output constraints in the agent setups.

To further investigate the relationship between reasoning token expenditure and agent performance, we conduct an additional experiment evaluating mobile GUI agents under explicitly constrained reasoning budgets. We deploy a recently released open-source VLM, GLM-4.6V (Z.ai, 2025), and enforce maximum reasoning token limits. We evaluate this model on the AndroidControl benchmark (using the step instruction prompt configuration) across six reasoning budgets: 0 (no reasoning), 128, 256, 512, 1,024, and unconstrained. As shown in Table 4, our results yield two main observations regarding reasoning budgets.

Reasoning Budget (tokens)	Action Type Accuracy (%)	Action Accuracy (%)
0 (No Reasoning)	76.83	70.32
128	**77.81**	**71.55**
256	77.65	71.00
512	77.12	70.38
1,024	76.98	70.33
Unconstrained	76.90	70.41

Table 4: Performance of GLM-4.6V on AndroidControl under different reasoning token budgets. A small budget (128 tokens) achieves the best performance, while larger budgets degrade accuracy.

First, a small reasoning budget helps. Compared to the strict non-reasoning setting (0 tokens), allocating a small reasoning budget improves both action type accuracy and overall action accuracy. The optimal setting in this configuration is 128 tokens, which yields a 0.98% improvement in action type accuracy and 1.23% improvement in action accuracy over the non-reasoning baseline.

Second, more tokens do not reliably help. Performance does not scale monotonically with a larger reasoning budget. Beyond 128 tokens, accuracy fluctuates and generally degrades. This indicates a distinct budget-performance tradeoff: allocating excessive test-time tokens allows the model to generate longer, potentially noisier reasoning chains,

which do not consistently translate into better action predictions and may distract the agent. These findings reinforce the conclusion that simply enabling unconstrained reasoning is suboptimal for mobile GUI agents. Instead, allocating small, targeted reasoning budgets may offer the best balance of latency, cost, and accuracy.

4.5 Implications

Generally, the reasoning process can provide an in-depth understanding of task instructions and GUIs. However, this capability does not consistently lead to correct responses when evaluated on static mobile GUI benchmarks due to their intrinsic limitations. Furthermore, based on the results above, we derive the following implications for improving mobile GUI agent development and evaluation.

• *For VLMs*: It is crucial to train VLMs on more comprehensive datasets to enhance their grounding and screen understanding capabilities during reasoning. This requires large datasets with appropriate annotations. Additionally, addressing inconsistencies between reasoning processes and final outputs through robust, domain-specific reward functions in reinforcement learning is essential (Guo et al., 2025).

• *For benchmarks*: Mobile GUI agents should ideally be evaluated on interactive benchmarks due to the inherent limitations of the current evaluation design of static benchmarks (i.e., requiring two identical actions). Real-world mobile environments could provide richer contextual information, therefore enabling mobile GUI agents to conduct more nuanced reasoning. Regardless of whether benchmarks are static or interactive, it is crucial to define clear and unambiguous tasks.

• *For mobile GUI agents*: To fully leverage the reasoning capability of VLMs, integrating more contextual information–whether through dynamic innovations in external tools or by incorporating holistic information into system prompts–may enhance mobile GUI agents. Otherwise, without relevant contextual information, the reasoning process is prone to generating suboptimal outcomes. What's more, adopting adaptive reasoning is crucial for mitigating long latency and high token costs, thereby maintaining the practicality of mobile GUI agents in real-world scenarios.

• *Scope of implications and generalizability*: Our findings apply most directly to the mobile GUI setting studied in this paper, using current VLMs on the evaluated benchmarks. In this setting, reasoning does not bring consistent gains. Its effect varies across benchmarks, agent designs, output formats, and model families, while also increasing token and latency costs. Some observations may extend to other GUI agents, including the limits of static benchmarks, the gap between reasoning and final actions, and the cost of always enabling reasoning. Other findings are more specific to mobile environments, such as errors related to view hierarchies, accessibility trees, and smartphone-specific UI structures. Direct studies on web and desktop environments are needed to test how far these findings generalize in future work.

5 Conclusion

In this work, we conduct the first empirical study to investigate whether the reasoning capabilities of commercial VLMs enhance the performance of mobile GUI agents. Using six series of VLMs with and without reasoning enabled, we comprehensively evaluate various mobile GUI agents under different configurations across static and interactive benchmarks. Current reasoning-enabled VLMs generally provide only marginal or even performance degradations, with a significant concern that they often fail tasks that could be completed without reasoning. We categorize the errors arising from the reasoning process and offer practical guidance for future research on improving mobile GUI agents, VLMs, and benchmarks.

Limitations

Although extensive experiments were conducted in this empirical study, there are still a few limitations in the evaluated benchmarks, models, and targeted platforms: (1) There are various grounding and GUI automation benchmarks (Wu et al., 2024; Li et al., 2025; Xu et al., 2024; Lu et al., 2024); however, due to monetary constraints, we evaluated only a few of them. We plan to include more benchmarks in future work. (2) We tested only six pairs of VLMs; however, numerous other commercial and open-source VLMs with reasoning capabilities have emerged (Kimi, 2025; Deepmind, 2025; Liu et al., 2025). Evaluating these models—including some smaller ones—would enhance the comprehensiveness of this study. (3) This study focuses on the mobile platform, while much of existing work targets desktop and web GUI tasks (Xie et al., 2024; Zhou et al., 2023; Bonatti et al., 2024). Incorporating experiments on these platforms would

make our findings and insights more robust in the domain of GUI agents.

References

Anthropic. 2024. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. https://www.anthropic.com/news/3-5-models-and-computer-use.

Anthropic. 2025a. Claude 3.7 sonnet and claude code. https://www.anthropic.com/news/claude-3-7-sonnet.

Anthropic. 2025b. Claude's extended thinking. https://www.anthropic.com/research/visible-extended-thinking.

Apple. 2024. Siri - apple. https://www.apple.com/siri/.

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, and 45 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Rogerio Bonatti, Dan Zhao, Francesco Bonacci, Dillon Dupont, Sara Abdali, Yinheng Li, Yadong Lu, Justin Wagle, Kazuhito Koishida, Arthur Bucker, and 1 others. 2024. Windows agent arena: Evaluating multi-modal os agents at scale. *arXiv preprint arXiv:2409.08264*.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. 2024. 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), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 9313–9332. Association for Computational Linguistics.

Google Deepmind. 2025. Gemini 2.5: Our most intelligent ai model — blog.google. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/#gemini-2-5-thinking.

Lingzhong Dong, Ziqi Zhou, Shuaibo Yang, Haiyue Sheng, Pengzhou Cheng, Zongru Wu, Zheng Wu, Gongshen Liu, and Zhuosheng Zhang. 2025. Say one thing, do another? diagnosing reasoning-execution gaps in vlm-powered mobile-use agents. *arXiv preprint arXiv:2510.02204*.

Longxi Gao, Li Zhang, Pengzhi Gao, Wei Liu, Jian Luan, and Mengwei Xu. 2025. Gui-shift: Enhancing vlm-based gui agents through self-supervised reinforcement learning. *arXiv preprint arXiv:2505.12493*.

Longxi Gao, Li Zhang, Shihe Wang, Shangguang Wang, Yuanchun Li, and Mengwei Xu. 2024. Mobileviews: A large-scale mobile gui dataset. *arXiv preprint arXiv:2409.14337*.

GLM-V Team. 2026. Glm-4.5v and glm-4.1v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*.

Google. 2024. Google assistant, your own personal google. https://assistant.google.com/.

Google. 2025. Gemini flash thinking - google deepmind. https://deepmind.google/technologies/gemini/flash-thinking/.

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*.

Youngmin Im, Byeongung Jo, Jaeyoung Wi, Seungwoo Baek, Tae Hoon Min, Joo Hyung Lee, Sangeun Oh, Insik Shin, and Sunjae Lee. 2025. Modular and multi-path-aware offline benchmarking for mobile gui agents. *arXiv preprint arXiv:2512.12634*.

Kimi. 2025. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*.

Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. 2025. Screenspot-pro: Gui grounding for professional high-resolution computer use. *arXiv preprint arXiv:2504.07981*.

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

Yuanchun Li, Hao Wen, Weijun Wang, Xiangyu Li, Yizhen Yuan, Guohong Liu, Jiacheng Liu, Wenxing Xu, Xiang Wang, Yi Sun, and 1 others. 2024b. Personal llm agents: Insights and survey about the capability, efficiency and security. *arXiv preprint arXiv:2401.05459*.

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

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*.

OpenAI. 2024. Openai o1 system card. https://openai.com/index/openai-o1-system-card/.

OpenAI. 2025. Introducing operator. https://openai.com/index/introducing-operator/.

Charilaos Pipis, Shivam Garg, Vasilis Kontonis, Vaishnavi Shrivastava, Akshay Krishnamurthy, and Dimitris Papailiopoulos. 2025. Wait, wait, wait... why do reasoning models loop? *arXiv preprint arXiv:2512.12895*.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, and 1 others. 2024. Androidworld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. 2023. Androidinthewild: A large-scale dataset for android device control. In *Advances in Neural Information Processing Systems*, volume 36, pages 59708–59728. Curran Associates, Inc.

Simular. 2025. Agent s2. https://www.simular.ai/agent-s2.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*.

Bryan Wang, Gang Li, and Yang Li. 2023. Enabling conversational interaction with mobile ui using large language models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, pages 1–17.

Yiqi Wang, Wentao Chen, Xiaotian Han, Xudong Lin, Haiteng Zhao, Yongfei Liu, Bohan Zhai, Jianbo Yuan, Quanzeng You, and Hongxia Yang. 2024. Exploring the reasoning abilities of multimodal large language models (mllms): A comprehensive survey on emerging trends in multimodal reasoning. *arXiv preprint arXiv:2401.06805*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Hao Wen, Yuanchun Li, Guohong Liu, Shanhui Zhao, Tao Yu, Toby Jia-Jun Li, Shiqi Jiang, Yunhao Liu, Yaqin Zhang, and Yunxin Liu. 2024a. Autodroid: Llm-powered task automation in android. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking, ACM MobiCom 2024, Washington D.C., DC, USA, November 18-22, 2024*, pages 543–557. ACM.

Hao Wen, Shizuo Tian, Borislav Pavlov, Wenjie Du, Yixuan Li, Ge Chang, Shanhui Zhao, Jiacheng Liu, Yunxin Liu, Ya-Qin Zhang, and 1 others. 2024b. Autodroid-v2: Boosting slm-based gui agents via code generation. *arXiv preprint arXiv:2412.18116*.

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*.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. 2024. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Mengwei Xu. 2025. Every software as an agent: Blueprint and case study. *arXiv preprint arXiv:2502.04747*.

Yifan Xu, Xiao Liu, Xueqiao Sun, Siyi Cheng, Hao Yu, Hanyu Lai, Shudan Zhang, Dan Zhang, Jie Tang, and Yuxiao Dong. 2024. Androidlab: Training and systematic benchmarking of android autonomous agents. *arXiv preprint arXiv:2410.24024*.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. 2023. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*.

Z.ai. 2025. zai-org/glm-4.6 — huggingface.co. https://huggingface.co/zai-org/GLM-4.6.

Chaoyun Zhang, Shilin He, Liqun Li, Si Qin, Yu Kang, Qingwei Lin, and Dongmei Zhang. 2025a. Api agents vs. gui agents: Divergence and convergence. *Preprint*, arXiv:2503.11069.

Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue Liu, Qingwei Lin, and 1 others. 2024a. Large language model-brained gui agents: A survey. *arXiv preprint arXiv:2411.18279*.

Jiwen Zhang, Jihao Wu, Yihua Teng, Minghui Liao, Nuo Xu, Xiao Xiao, Zhongyu Wei, and Duyu Tang. 2024b. Android in the zoo: Chain-of-action-thought for GUI agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 12016–12031. Association for Computational Linguistics.

Li Zhang, Shihe Wang, Xianqing Jia, Zhihan Zheng, Yunhe Yan, Longxi Gao, Yuanchun Li, and Mengwei Xu. 2024c. Llamatouch: A faithful and scalable

testbed for mobile ui task automation. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, pages 1–13.

Li Zhang and Mengwei Xu. 2025. Scaling test-time compute in mobile gui agents with parallel speculative execution. In *Proceedings of the 2nd International Workshop on Edge and Mobile Foundation Models*, pages 25–30.

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*.

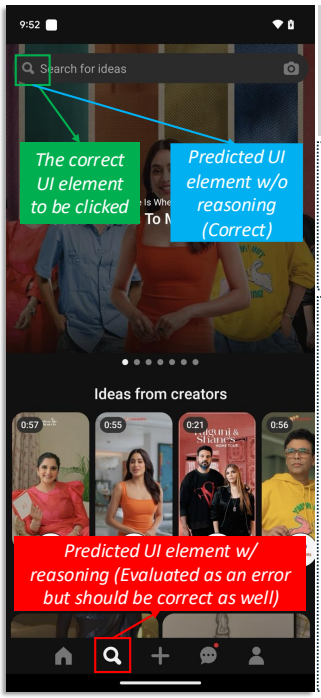
Zhuosheng Zhang and Aston Zhang. 2024. You only look at screens: Multimodal chain-of-action agents. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 3132–3149. Association for Computational Linguistics.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and 1 others. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

A Appendix

A.1 Demonstrations of Errors by Reasoning VLMs in AndroidControl

Table 3 categorizes the errors introduced by reasoning VLMs in AndroidControl. In this section, we illustrate these errors using traces collected during our experiments. We present the inputs and responses of both non-reasoning and reasoning VLMs.

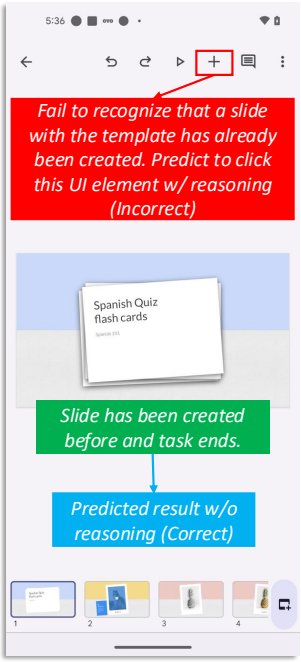


Input:
Task instruction: I would like to browse through the street art on the Pinterest app to get some ideas for street art since I want to paint some street art on the wall of my living room.
Step instruction: Click on the search icon.
Ground Truth: *{"action_type": "click", "x": 76, "y": 220}*

Claude 3.7 Sonnet (w/o thinking): I can see that there is a small search icon at coordinates [77, 220] and a larger search bar with text "Search for ideas" at coordinates [534, 220] Additionally, there's a Search tab at the bottom navigation bar at coordinates [372, 2274]. Since the step instruction is to "Click on the search icon", I'll click on the small search icon that's explicitly labeled as "Search" in the UI elements. *{"action_type":"click", "x":77,"y":220}*

Claude 3.7 Sonnet (w/ thinking):
 # Reasoning process
 # Analyze the current UI state.
1. There is a search icon at the top left of the screen (coordinates [77, 220]).
2. There's also a search tab at the bottom of the screen (coordinates [372, 2274]).
3. There's a search bar with text "Search for ideas" (coordinates [534, 220])
 # Analyze the next step.
I should click on the element with text "Search" at coordinates [77, 220]. However, I also notice there's a full search tab at the bottom of the screen. Looking at the bottom of the screen in the screenshot, I can see the tab bar with Home, Search, Create, Notifications, and Saved tabs. *The Search tab is at position [372, 2274]. This seems like the most appropriate option to click since it's labeled as "Search, Tab" which would take the user to the search functionality of Pinterest.*
Answer: *{"action_type":"click","x":372,"y":2274}*

Figure 5: Benchmark Error: Weak Evaluation Method.

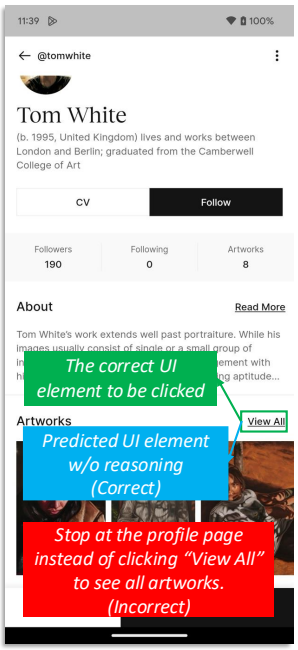


Input:
Task instruction: In Google Slides, I want to create a new Google Slide with a template flash card for presentation at the anniversary ceremony.
Previous Action List: (just take the second action as an example)
{"action_type": "click", "x": 953, "y": 2231}
Ground Truth: *{"action_type":"status","goal_status":"successful"}*

Claude 3.7 Sonnet (w/o thinking): We have already opened Google Slides and are viewing a presentation with a flash card template. The first slide shows a Spanish Quiz flash card template which is suitable for an anniversary presentation.
{"action_type":"status","goal_status":"successful"}

Claude 3.7 Sonnet (w/ thinking):
 # Reasoning process
 # Analyze the current UI state.
1. We're already in Google Slides and there's a presentation open with flash card templates.
2. The current slide (Slide 1) shows "Spanish Quiz flash cards" with "Spanish 101" text.
3. The existing slides appear to be flash card templates - exactly what we need, but themed for Spanish learning.
 # Analyze the next step.
Since the task specifically asks to create a new slide with a flash card template for an anniversary ceremony, and the current template is Spanish-related, *it makes the most sense to add a new slide. Let's click the "Add slide" button first.*
Answer: *{"action_type":"click","x":1022,"y":2211}*

Figure 6: Benchmark Error: Static GUI Input Limitation.

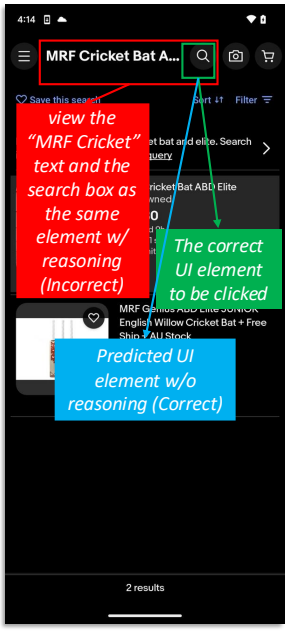


Input:
Task instruction: On the Peggy app, search for Arts by Tom White.
Previous actions *(just take the first action as an example)*:
{"action_type": "open_app", "app_name": "Peggy"};
Ground Truth: *{"action_type": "click", "x": 977, "y": 1563}*

Claude 3.7 Sonnet (w/o thinking): We're already on Tom White's profile page. The "Artworks" section is visible on the screen, showing some of his art pieces. To fully explore his artworks, I should click on "View All" to see all of his art pieces. *{"action_type":"click","x": 977, "y": 1563}*

Claude 3.7 Sonnet (w/ thinking):
 # Reasoning process
 # Analyze the previous action list.
1. The app was opened and was waited for to load.
2. A click action at position (986, 190) was performed and text "Tom White" was input.
3. A click at position (540, 344) was performed. A scroll down action was performed.
 # Analyze the current UI state.
1. We're on Tom White's profile page. We can see his bio, follower count, and artwork information.
2. The "Artworks" section is visible at the bottom of the screen, showing his art pieces.
 # Analyze the next step.
The task is to search for "Arts by Tom White", and we've already searched for "Tom White" and are now viewing his profile with his artworks visible. So the task has been successfully completed.
Answer: *{"action_type":"status","goal_status":"successful"}*

Figure 7: Benchmark Error: Unclear Task Instruction.

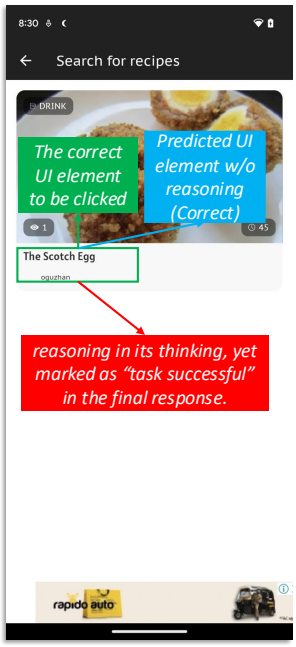


Input:
Task instruction: Compare prices of English willow cricket bat grade 1 from the different retailers in *Ebay* app.
Ground Truth: *{"action_type": "click", "x": 764, "y": 201}*

Claude 3.7 Sonnet (w/o thinking): I need to search for "English willow cricket bat grade 1" since the current search is showing MRF cricket bats but not specifically English willow grade 1 bats as requested in the task. I'll modify the search. *{"action_type":"click","x":765,"y":201}*

Claude 3.7 Sonnet (w/ thinking):
 # Reasoning process
 # Analyze the current UI state.
1. We're in the eBay app with search results for "MRF Cricket Bat ABD Elite".
2. There are only 2 results showing: MRF Cricket Bat ABD Elite (Pre-Owned); MRF Genius ABD Elite JUNIOR English Willow Cricket Bat (Brand New).
 # Analyze the instruction.
The task asks to compare prices of "English willow cricket bat grade 1" from different retailers.
 # Analyze the next step.
I should either:
1. Click on the search box to modify the search terms.
2. Click on "your original query" link to see what the original search was and possibly modify it.
I think the best action is to click on the search box to enter a new search term.
Answer: *{"action_type":"click","x":424,"y":202}*

Figure 8: VLM Error: Limited GUI Comprehension.

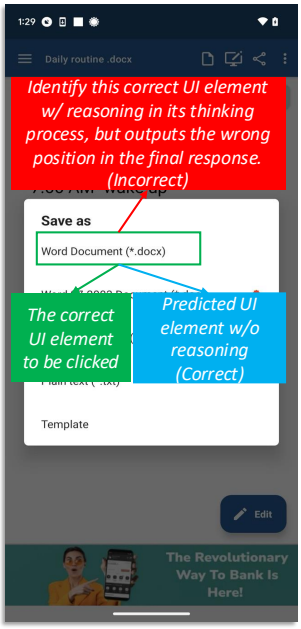


Input:
Task instruction: Open the All Recipes App. Type in a search bar and search for "The Scotch Egg", then select to view it.
Ground Truth: *{"action_type": "click", "x": 539, "y": 950}*

Claude 3.7 Sonnet (w/o thinking): I can see that we are already in the *All Recipes* app, and "The Scotch Egg" recipe is displayed on the screen. Looking at the task instruction, we need to "Open the All Recipes App, Type in a search bar and search for The Scotch Egg, then select to view it." Since the recipe is already visible on the screen, we need to click on it to view the details. *{"action_type":"click","x":540,"y":951}*

Claude 3.7 Sonnet (w/ thinking):
 # Reasoning process
 # Analyze the current UI state.
1. We appear to already be in the *All Recipes* app.
2. There's a search bar with the text "Search for recipes" at the top.
3. "The Scotch Egg" is already displayed in the search results (with text at coordinates [540, 951]).
 # Analyze the next step.
I can see "The Scotch Egg" text at coordinates [540, 951], which appears to be a clickable item in the search results. Clicking on this should open the detailed view of the recipe.
The logical next action would be to click on "The Scotch Egg" entry to view its details.
Answer: Since we can see we've already completed all parts of the task instruction (opened the app, searched for The Scotch Egg, and we're currently viewing it), I should indicate that the goal has been successfully completed. *{"action_type":"status","goal_status":"successful"}*

Figure 9: VLM Error: Reasoning-Response Inconsistency.

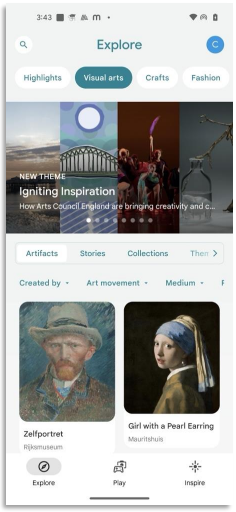


Input:
Task instruction:
On the *OfficeSuite* app, save the document named Daily routine as Weekday schedule.
Previous actions *(just take the first action as an example)*:
{"action_type": "click", "x": 1025, "y": 211}
Ground Truth: *{"action_type": "click", "x": 540, "y": 959}*

Claude 3.7 Sonnet (w/o thinking): I need to select one of these file formats and then I'll need to rename the document. First, I'll select the Word Document format since it's the most common and compatible format. *{"action_type":"click","x":540,"y":959}*

Claude 3.7 Sonnet (w/ thinking):
 # Reasoning process
 # Analyze the current UI state.
1. We're in the *OfficeSuite* app, and there's a "Save as" dialog open.
2. The document is currently named "Daily routine.docx" as shown in the title bar, and we need to rename it to "Weekday schedule" when saving.
3. The UI shows different format options to save the document.
 # Analyze the next step.
I can see that the "Save as" dialog is open but I don't see a text field to enter the new filename yet. It seems I first need to select the file format, and then I'll likely be prompted to enter the filename.
I should click on the first option "Word Document (.docx)"* to proceed with the save process.
Answer: *I should click on the first option "Word Document (*.docx)"* to continue with the save process. *{"action_type":"click","x":540,"y":627}*

Figure 10: VLM Error: Incorrect Grounding.

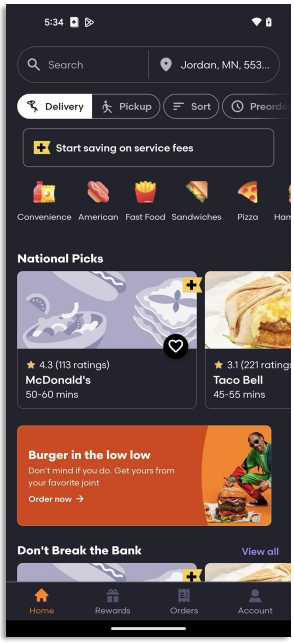


Input:
Task instruction:
In the *Art & Culture* app, I would like to look for visual arts because I have been very passionate about visual art.
Step instruction: Swipe up to browse more arts.
Ground Truth: *{"action_type": "scroll", "direction": "down"}*

Claude 3.7 Sonnet (w/o thinking): *{"action_type":"scroll","direction":"down"}* ✓

Claude 3.7 Sonnet (w/ thinking):
 # Reasoning process
 # Analyze the current UI state.
The current screen shows some art pieces including "Zelfportret" and "Girl with a Pearl Earring" at the bottom part of the screen.
 # Analyze the next step.
To view more art content that might be below these, *I should perform a scroll down action.*
Looking at the available actions, *I should use: {"action_type":"scroll","direction":"up"}*
Answer: *{"action_type":"scroll","direction":"up"}* ✗

Figure 11: VLM Error: Incorrect Reasoning. The model correctly infers that it should scroll down to view more content but incorrectly outputs a scroll-up action in both its reasoning process and final answer.



Input:
Task instruction: I'd want to see Skechers women's ultra flex sneakers in the *Amazon* app.
Previous actions: N/A
Ground Truth: *{"action_type": "open_app", "app_name": "Amazon"}*

Claude 3.7 Sonnet (w/o thinking): I can see that we need to search for Skechers women's ultra flex sneakers on *Amazon*, but we're currently in what appears to be a food delivery app (showing restaurants like McDonald's and Taco Bell). We need to open the *Amazon* app first. *{"action_type": "open_app", "app_name": "Amazon"}* ✓

Claude 3.7 Sonnet (w/ thinking):
 # Reasoning process
 # Analyze the current UI state.
Based on the screenshot and UI elements, I can see that the user is currently in what appears to be a food delivery app (showing restaurants like McDonald's and Taco Bell), not the *Amazon* app.
 # Analyze the next step.
Since *we need to open the Amazon app*, and we're in a different app, the appropriate action would be to:
1. Navigate to the home screen first.
2. Then open the Amazon app.
Answer: *{"action_type":"navigate_home"}* ✗

Figure 12: VLM Error: Hallucination. The "navigate_home" action is not in the given action space.