GUI-Eyes: Tool-Augmented Perception for Visual Grounding in GUI Agents

Chen Chen[1,2,3], **Jiawei Shao**[2*], **Dakuan Lu**[2], **Haoyi Hu**[4], **Xiangcheng Liu**[1,3], **Hantao Yao**[1], **Wu Liu**[1]

[1]University of Science and Technology of China
[2]Institute of Artificial Intelligence (TeleAI), China Telecom
[3]Shanghai Innovation Institute
[4]Shanghai Jiao Tong University
chenchen0318@mail.ustc.edu.cn

Abstract

Recent advances in vision-language models (VLMs) and reinforcement learning (RL) have driven progress in GUI automation. However, most existing methods rely on static, one-shot visual inputs and passive perception, lacking the ability to adaptively determine when, whether, and how to observe the interface. We present GUI-Eyes, a reinforcement learning framework for active visual perception in GUI tasks. To acquire more informative observations, the agent learns to make strategic decisions on both whether and how to invoke visual tools, such as cropping or zooming, within a two-stage reasoning process. To support this behavior, we introduce a progressive perception strategy that decomposes the decision-making into coarse exploration and fine-grained grounding, coordinated by a two-level policy. In addition, we design a spatially continuous reward function tailored to tool usage, which integrates both location proximity and region overlap to provide dense supervision and alleviate the reward sparsity common in GUI environments. On the ScreenSpot-Pro benchmark, GUI-Eyes-3B achieves 44.8% grounding accuracy using only 3k labeled samples, significantly outperforming both supervised and RL-based baselines. These results highlight that tool-aware active perception, enabled by staged policy reasoning and fine-grained reward feedback, is critical for building robust and data-efficient GUI agents.

1 Introduction

The development of large language models (LLMs) (Touvron et al. 2023; Grattafiori et al. 2024; Team 2024; Yang et al. 2025) and vision language models (VLMs) (Wang et al. 2024c; Bai et al. 2025; Achiam et al. 2023; Hurst et al. 2024) has introduced new opportunities and challenges in applying these models to GUI tasks. Existing GUI agents are mainly based on supervised fine-tuning (SFT) (Wu et al. 2024b; Qin et al. 2025; Hong et al. 2024), where large annotated datasets are used to teach model interface understanding and action planning. However, this approach suffers from several limitations: it requires expensive, labor-intensive data collection, and the resulting models often lack robustness when deployed in unfamiliar or out-of-domain environments (Chai et al. 2024; Muennighoff et al. 2025).

DeepSeek-R1 (Guo et al. 2025) demonstrates that reinforcement learning (RL) can enhance large models' problem-solving ability without human-labeled data, by optimizing behavior through interaction and well-designed rewards. In GUI tasks, RL effectively reduces supervision demands while improving adaptability and generalization (Luo et al. 2025; Liu et al. 2025b; Lu et al. 2025; Zhou et al. 2025; Gao, Zhang, and Xu 2025), making it a promising alternative to SFT. Despite this potential, most existing methods still optimize only textual outputs (Lu et al. 2025; Luo et al. 2025; Liu et al. 2025b; Yuan et al. 2025), overlooking visual cues essential for GUI understanding. Real users rely on visual attention to locate elements and interpret layouts (Zheng et al. 2025; Xu et al. 2025; Hong et al. 2024; Su et al. 2025); models limited to language reasoning struggle with ambiguous instructions and complex interfaces. Thus, GUI agents must integrate perception and decision-making—learning not only what to see but how often to observe—to support robust visual-grounded interactions.

To address the limitations of supervised learning in terms of data annotation cost and generalization, we propose **GUI-Eyes**, a reinforcement learning framework centered on active perception. Unlike traditional GUI agents that rely on static, one-shot visual input, GUI-Eyes empowers the model to dynamically decide whether to invoke visual tools during reasoning, and to flexibly configure their parameters (e.g., crop region, zoom scale) for acquiring task-relevant observations step by step. By modeling visual perception as an optimizable policy, GUI-Eyes learns a **perception–reasoning–perception** loop that tightly coordinates visual observation with language-based decision-making.

As shown in Figure 1, GUI-Eyes-3B achieves superior performance on the ScreenSpot-Pro benchmark compared to existing models of similar or larger scales, validating the effectiveness of our tool-aware active perception framework.

Our main contributions are summarized as follows:

- We propose **GUI-Eyes**, a novel framework that integrates active perception into GUI agents. It enables the model to autonomously determine *when* and *how* to invoke visual tools during reasoning, achieving more precise and task-adaptive visual understanding.
- To support the learning of effective active perception behaviors, we design a multi-factor reward function that provides structured supervision over format correctness,

*Corresponding author.

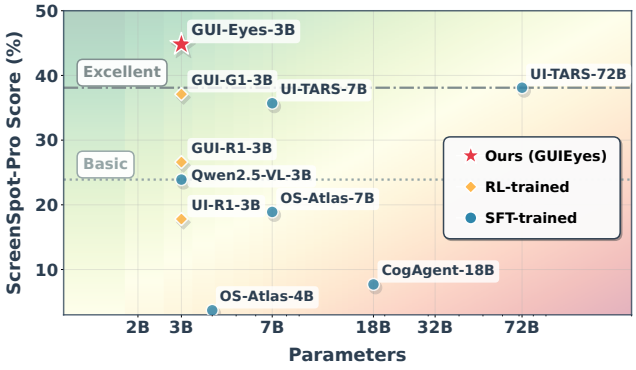

Figure 1: Performance Scaling of Multimodal UI Understanding Models on the ScreenSpot-Pro Benchmark. Our method achieves state-of-the-art performance.

initial localization, and spatial coverage. This facilitates more stable and generalizable tool-use policy learning.

- We conduct extensive experiments on the ScreenSpot-Pro benchmark. GUI-Eyes-3B achieves **44.8%** grounding accuracy using only **3,000 labeled samples**, significantly outperforming both supervised and RL-based baselines, thereby demonstrating strong sample efficiency and robust generalization.

2 Related Work

2.1 GUI Agents

With the growing capabilities of multimodal large language models (MLLMs) (Achiam et al. 2023; Wang et al. 2024c; Bai et al. 2025; Hurst et al. 2024), GUI-based agents have become a central focus in human–computer interaction research (Wang et al. 2024b,a; Zhang et al. 2024; Tang et al. 2025). Existing work in this area can be broadly categorized into two main paradigms: structure-driven and vision-driven approaches.

Structure-driven methods rely on structured representations such as HTML or DOM trees to parse and execute interface instructions (Gur et al. 2023; Kim, Baldi, and McAleer 2023; Deng et al. 2023; Zhou et al. 2023; Rawles et al. 2023). These methods benefit from explicit symbolic semantics and direct access to internal interface states.

Vision-driven methods, in contrast, operate directly on GUI screenshots, leveraging visual perception and language instructions for open-ended reasoning and grounding (Zhang et al. 2025; Hong et al. 2024; You et al. 2024; Liu et al. 2025a; Wu et al. 2025b). For instance, AppAgent (Zhang et al. 2025) explores autonomous interactions in mobile applications, while CogAgent (Hong et al. 2024) uses high-resolution visual encoders for better UI understanding. OS-Atlas (Wu et al. 2024b) proposes a unified action representation and pre-trains on 13M UI elements to enable cross-platform generalization. ScaleTrack (Huang et al. 2025a) designs a backtracking-based task to enhance multi-step reasoning, training on 7.5M screenshots for joint grounding and planning. Despite strong empirical results, these methods mainly depend on supervised learning with static input–output pairs, limiting their ability to actively acquire perceptual information or adjust reasoning strategies during inference.

2.2 Reinforcement Fine-Tuning

Traditional supervised fine-tuning (SFT) for GUI agents requires extensive annotated data (Wu et al. 2024b; Qin et al. 2025; Huang et al. 2025a; Wu et al. 2024a), making it costly and less generalizable. Recently, rule-based reinforcement learning (RL) has emerged as a more efficient alternative, especially in low-resource scenarios. DeepSeek-R1 (Guo et al. 2025) introduced this paradigm for large language models by employing predefined reward functions (e.g., symbolic correctness) to evaluate outputs without human feedback. This idea has been extended to GUI agents (Lu et al. 2025; Luo et al. 2025; Zhou et al. 2025; Liu et al. 2025b; Gao, Zhang, and Xu 2025; Wei et al. 2025; Gu et al. 2025), showing strong data efficiency. UI-R1 (Lu et al. 2025) first introduced rule-based RL for low-level GUI action prediction, achieving strong performance with only 130 mobile samples and substantially lowering data requirements. InfiGUI-R1 (Liu et al. 2025b) proposed the Actor2Reasoner framework, encouraging agents to "think before acting and reflect after execution," thereby improving their understanding and operation in complex UI layouts. Interestingly, GUI-G1 (Zhou et al. 2025) showed that, for some tasks, directly producing the final answer can even surpass step-wise reasoning, indicating that the utility of intermediate reasoning varies with task complexity and type.

Although these methods have achieved notable progress in data efficiency and policy learning, they still rely mainly on text-only reasoning (Wang et al. 2025a,b; Song et al. 2025), overlooking the role of visual information in complex GUI environments. Inspired by advances in visual reasoning and active perception (Zheng et al. 2025; Huang et al. 2025b; Xu et al. 2025), we argue that GUI agents should proactively observe and reason—leveraging visual tools to interpret screenshots and understand task-specific visual contexts. This integration enhances situational awareness and boosts performance in visually complex or ambiguous GUI scenarios, following recent studies emphasizing collaborative reasoning between language and vision modules (Shao and Li 2025; An et al. 2025).

3 GUI-Eyes

In this section, we introduce our method, including the Progressive Inference (3.1), Reward Design (3.2), and the Training Details (3.3). An overview of the overall architecture is shown in Figure 2.

3.1 Progressive Inference

Recent GUI agents typically rely on static, one-shot visual input and pre-defined tool usage, lacking the ability to actively perceive task-relevant information during reasoning. To overcome this limitation, we propose **GUI-Eyes**, a reinforcement learning framework that empowers agents with active perception capabilities. Instead of relying on fixed visual inputs, GUI-Eyes learns to strategically decide

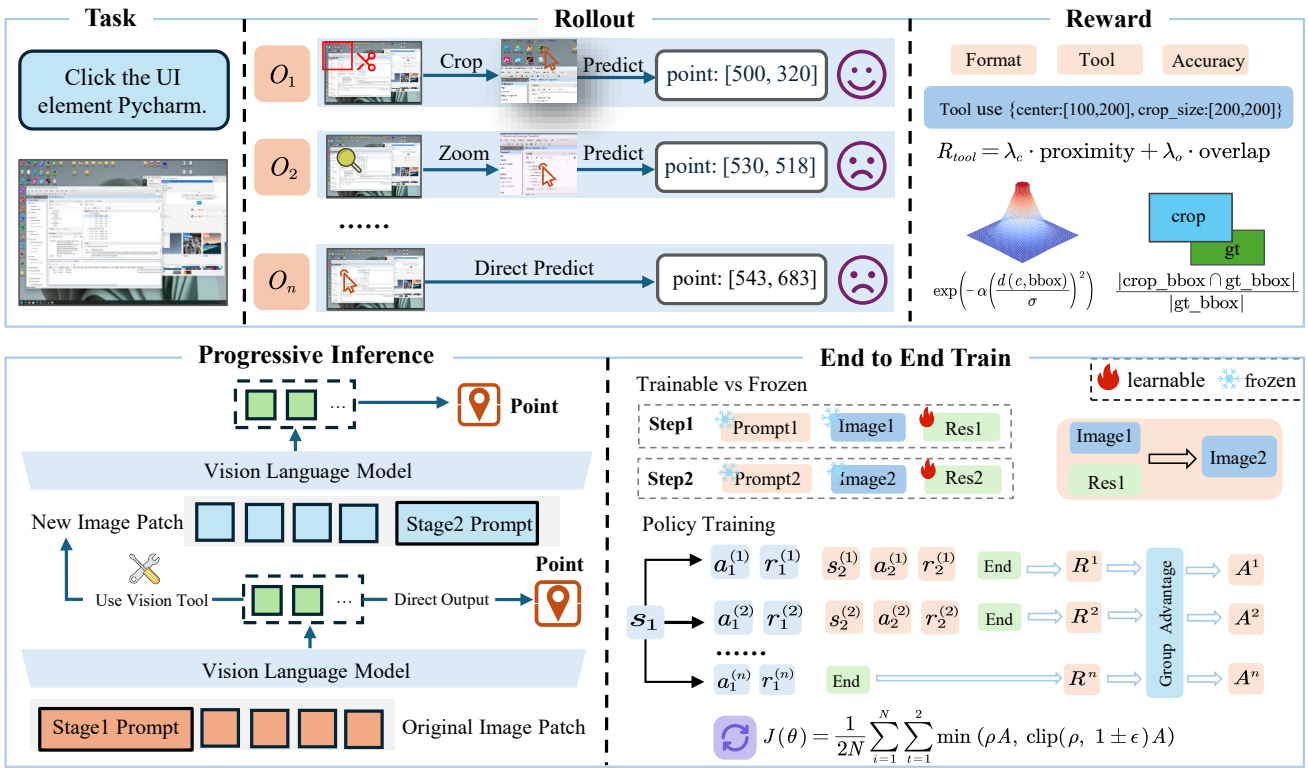

Figure 2: Overview of the GUI-Eyes Framework. The top illustrates a rollout example with optional visual tool invocation, together with a tool-specific reward function that combines spatial proximity and region overlap relative to the ground-truth. The bottom depicts the progressive inference architecture and end-to-end training pipeline, where the two-stage decision process is guided by stage-specific prompts, and visual inputs are dynamically generated through previously applied visual tools.

when and how to observe the GUI environment via a set of visual tools (e.g., cropping, zooming). The model forms a perception–reasoning–perception loop, enabling dynamic attention and adaptive visual understanding.

Stage 1: Active Perception Planning.
Given a natural language instruction and the original GUI screenshot, the model performs an initial grounding attempt. It then autonomously decides whether to invoke a visual tool and predicts its configuration parameters, such as the crop center, region size, and zoom scale. These parameters are used to generate an intermediate visual input (e.g., a cropped or zoomed image), which serves as refined perceptual input for the next stage.

As illustrated in the top part of Figure 2, the agent may choose different rollouts, such as applying Crop, Zoom, or directly predicting without a tool.

Stage 2: Reasoning with Focused Perception.
In the second stage, the model conducts more fine-grained reasoning based on the intermediate input. By focusing on visually clearer and task-relevant regions, the model enhances the accuracy and robustness of its prediction, particularly under high-resolution, cluttered, or ambiguous interface conditions.

This two-stage process enables the agent to interactively refine its perception based on task-specific feedback. A representative example of active visual grounding, in which the model invokes the Crop tool, is shown in Figure 3.

Rather than concatenating multi-turn inputs, we treat each inference round as a perception-informed decision point. The output of the previous step—such as the chosen visual tool and its manipulated image—reshapes the visual input for the next. As illustrated in the lower-left part of Figure 2, this process enables the agent to progressively refine its focus across reasoning steps, forming a closed-loop cycle of perception, reasoning, and re-perception.

3.2 Reward Design for Reinforcement Learning

As illustrated in the bottom-right part of Figure 2, GUI-Eyes performs end-to-end reinforcement learning that jointly optimizes perception and reasoning policies. Each training trajectory consists of two decision steps—Stage 1 for visual tool planning and Stage 2 for task execution—enabling a unified optimization of the progressive inference process.

To guide this optimization, we design a unified reward function that integrates perception-aware actions and outcome-level accuracy into a single learning signal:

$$R(\tau) = \lambda_{\text{acc}} R_{\text{acc}} + \lambda_{\text{format}} R_{\text{format}} + \lambda_{\text{tool}} R_{\text{tool}} \quad (1)$$

Format Reward R_{format}: Encourages syntactic correctness by penalizing malformed tags or invalid actions.

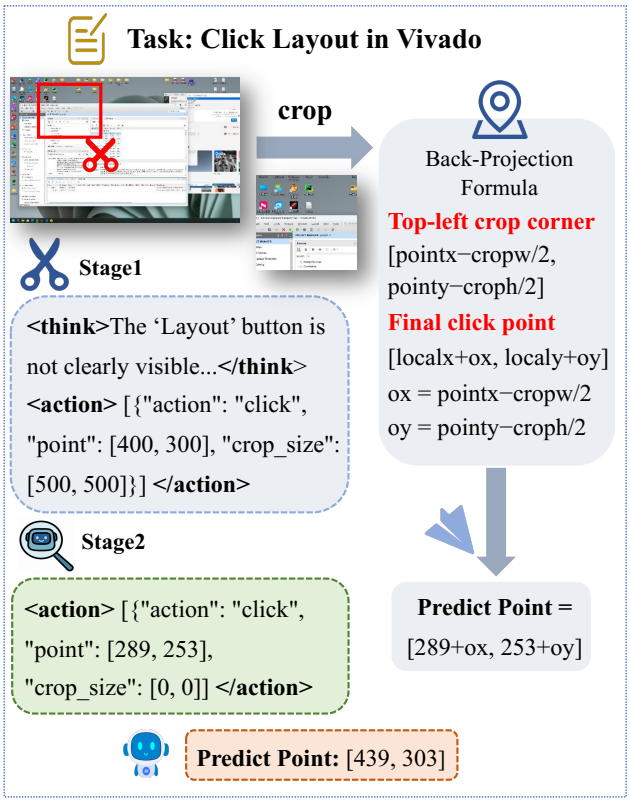

Figure 3: Inference Example of Tool-Augmented Reasoning with Cropping in a GUI Task.

Accuracy Reward R_{acc}: A binary reward that assigns 1 if the predicted point falls within the ground-truth bounding box, and 0 otherwise.

Tool Reward R_{tool}: Reflects the quality of tool usage, combining the proximity of the selected center c to the target and the region coverage:

$$R_{\text{tool}} = \lambda_{\text{center}} \cdot \exp\left(-\alpha \left(\frac{d(c, \text{gt\_bbox})}{\sigma}\right)^2\right) + \lambda_{\text{overlap}} \cdot \frac{|\text{crop\_bbox} \cap \text{gt\_bbox}|}{|\text{gt\_bbox}|} \tag{2}$$

Here, gt_bbox denotes the ground-truth bounding box of the target element, and crop_bbox represents the region produced by the model's tool action. The function $d(c, \text{gt\_bbox})$ computes the shortest distance from the selected center c to the boundary of gt_bbox. The weighting factors λ_{center} and λ_{overlap} balance the contributions of the center alignment and spatial coverage terms.

Remark: Our framework supports both **Crop** and **Zoom** tools, which share the same spatial parameters (center and size). Since zooming can be viewed as a visual transformation of cropping, we apply the same reward function to both, allowing for unified training and supervision.

When the model decides not to invoke any visual tool in the first stage, we set crop_size $= [0, 0, 0, 0]$. In this case, the IoU term of R_{tool} becomes zero, but the center term still gives a small positive reward if the predicted location is close to the ground-truth region. This encourages the model to make accurate direct predictions for simpler tasks, preventing unnecessary tool usage and promoting adaptive tool invocation based on task complexity.

3.3 Policy Optimization and Training Details

Building upon the unified reward described above, we optimize GUI-Eyes via end-to-end reinforcement learning. As illustrated in the bottom-right part of Figure 2, the two-stage reasoning process (perception and decision) is trained jointly as a unified trajectory, where the reward signal consistently guides both visual perception and task execution.

Advantage Computation. We use the total reward $R(\tau)$ to guide policy learning. Following prior work (e.g., GRPO (Shao et al. 2024)), the advantage $A(a_t^{(i)})$ of each sampled response is computed by normalizing its reward within a batch. Specifically, given N sampled responses $\{o_1, o_2, \ldots, o_N\}$ with corresponding rewards $\{R_1, R_2, \ldots, R_N\}$, the advantage for response i is computed as:

$$A_i = \frac{R_i - \text{mean}(R_1, R_2, ..., R_N)}{\text{std}(R_1, R_2, ..., R_N)} \tag{3}$$

We adopt an agent-centric variant of the GRPO algorithm (Feng et al. 2025) that supports multi-stage reasoning, treating each decision step as part of a unified trajectory. This enables joint optimization of visual perception and task execution policies via end-to-end reinforcement learning. Our optimization objective is formulated as follows:

$$J(\theta) = \mathbb{E}_{x\sim p(X),\, \{\tau_i\}\sim\pi_{\theta_{\text{old}}}} \left[\frac{1}{2N} \sum_{i=1}^{N} \sum_{t=1}^{2} \min\Big(\rho_\theta(a_t^{(i)}) A(a_t^{(i)}),\ \text{clip}(\rho_\theta(a_t^{(i)}), 1 \pm \epsilon) A(a_t^{(i)}) \Big) \right] \tag{4}$$

Here, $\rho_\theta(a_t^{(i)}) = \frac{\pi_\theta(a_t^{(i)})}{\pi_{\theta_{\text{old}}}(a_t^{(i)})}$ is the importance sampling ratio between the current and old policies. $A(a_t^{(i)})$ denotes the estimated advantage of action $a_t^{(i)}$, computed based on normalized total rewards. The clip operator stabilizes updates by constraining the impact of large policy shifts.

Each sampled trajectory τ_i consists of two decision steps corresponding to our two-stage reasoning process ($t = 1$ for perception, $t = 2$ for final decision). The objective averages over N samples and both reasoning steps per sample.

4 Experiment

In this section, we describe our experimental setup from three perspectives. Implementation Details outlines the training configuration, datasets, and evaluation benchmarks. The Experimental Results and Analysis section presents the

Model	CAD		Development		Creative		Scientific		Office		OS		Avg.		
	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Avg
Proprietary Models															
GPT-4o	2.0	0.0	1.3	0.0	1.0	0.0	2.1	0.0	1.1	0.0	0.0	0.0	1.3	0.0	0.8
Claude Computer Use	14.5	3.7	22.0	3.9	25.9	3.4	33.9	15.8	30.1	16.3	11.0	4.5	23.4	7.1	17.1
General Open-source Models															
Qwen2.5-VL-3B	9.1	7.3	22.1	1.4	26.8	2.1	38.2	7.3	33.9	15.1	10.3	1.1	23.6	3.8	16.1
Qwen2.5-VL-7B	16.8	1.6	46.8	4.1	35.9	7.7	49.3	7.3	52.5	20.8	37.4	6.7	38.9	7.1	26.8
GUI-Specific Models(SFT(+RL))															
CogAgent-18B	7.1	3.1	14.9	0.7	9.6	0.0	22.2	1.8	13.0	0.0	5.6	0.0	12.0	0.8	7.7
OS-Atlas-7B	12.2	4.7	33.1	1.4	28.8	2.8	37.5	7.3	33.9	5.7	27.1	4.5	28.1	4.0	18.9
ShowUI-2B	2.5	0.0	16.9	1.4	9.1	0.0	13.2	7.3	15.3	7.5	10.3	2.2	10.8	2.6	7.7
UGround-7B	14.2	1.6	26.6	2.1	27.3	2.8	31.9	2.7	31.6	11.3	17.8	0.0	25.0	2.8	16.5
UGround-V1-7B	15.8	1.2	51.9	2.8	47.5	9.7	57.6	14.5	60.5	13.2	38.3	7.9	45.2	8.1	31.1
UI-TARS-2B	17.8	4.7	47.4	4.1	42.9	6.3	56.9	17.3	50.3	17.0	21.5	5.6	39.6	8.4	27.7
UI-TARS-7B	20.8	9.4	58.4	12.4	50.0	9.1	63.9	**31.8**	63.3	20.8	30.8	16.9	47.8	16.2	35.7
InfiGUI-R1-3B	33.0	**14.1**	51.3	12.4	44.9	7.0	58.3	20.0	65.5	28.3	43.9	12.4	49.1	14.1	35.7
GUI-Specific Models(RL Only)															
UI-R1-3B	11.2	6.3	22.7	4.1	27.3	3.5	42.4	11.8	32.2	11.3	13.1	4.5	24.9	6.4	17.8
GUI-R1-3B	26.4	7.8	33.8	4.8	40.9	5.6	61.8	17.3	53.6	17.0	28.1	5.6	45.1	8.1	30.2
GUI-R1-7B	23.9	6.3	49.4	4.8	38.9	8.4	55.6	11.8	58.7	26.4	42.1	16.9	45.9	11.2	32.4
SE-GUI-3B	38.1	12.5	55.8	7.6	47.0	4.9	61.8	16.4	59.9	24.5	40.2	12.4	50.4	11.8	35.9
GUI-G1-3B	39.6	9.4	50.7	10.3	36.6	11.9	61.8	30.0	67.2	**32.1**	23.5	10.6	49.5	**16.8**	37.1
3B(ours)	**48.2**	9.4	**70.8**	**12.4**	**56.6**	**13.3**	**69.4**	19.1	**75.7**	24.5	**59.8**	**20.2**	**62.8**	15.7	**44.8**

Table 1: Performance comparison of different agent models across various task categories based on Text, Icon, and Average scores on ScreenSpot-Pro. Results marked in **bold** represent the best performance.

performance of GUI-Eyes across various benchmarks, comparing it to state-of-the-art methods and offering further insights into task-specific behaviors. Ablation Study analyzes the contribution of key components to overall performance.

4.1 Implementation Details

Training Details. We adopt Qwen2.5-VL-3B (Bai et al. 2025) as our base model and conduct training within the DeepEyes (Zheng et al. 2025) framework using the GRPO algorithm (Shao et al. 2024). Training is performed for **1** epoch with a batch size of **32** and a sampling temperature of 1.0 to encourage exploration. Policy optimization is carried out using the AdamW optimizer (Loshchilov and Hutter 2017) with a learning rate of 1×10^{-6}. All experiments are conducted on 8×NVIDIA H100-80G GPUs.

Training Dataset. Our training dataset is constructed by carefully sampling 3,000 instances from OS-Atlas (Wu et al. 2024b), OS-Genesis (Sun et al. 2024), GUI-R1 (Luo et al. 2025), and AndroidControl (Li et al. 2024). The dataset spans three major platform categories: Android, Desktop, and Web, thereby ensuring a diverse task distribution and comprehensive coverage of real-world GUI interactions.

Benchmarks and Evaluation Metrics. We evaluate GUI grounding performance on three established benchmarks: ScreenSpot, ScreenSpot-v2, and ScreenSpot-Pro. ScreenSpot (Cheng et al. 2024) contains relatively simple tasks focused on common mobile and desktop interfaces. ScreenSpot-v2 (Wu et al. 2024b) extends this by incorporating more diverse interface layouts and interaction patterns. ScreenSpot-Pro (Li et al. 2025) targets professional, high-resolution interfaces that exhibit greater structural and semantic complexity. It is designed to assess model generalization in more realistic GUI environments. Following the standard evaluation protocol (Cheng et al. 2024; Li et al. 2025), a prediction is considered correct if the predicted center point falls within the ground-truth bounding box.

Comparison Baselines. We evaluate our model against a wide range of existing methods across different categories: proprietary models (e.g., GPT-4o (Hurst et al. 2024), Claude Computer Use (Anthropic 2024)), general vision-language models (e.g., Qwen2.5-VL (Bai et al. 2025)), and GUI-specific models with supervised finetuning or reinforcement learning (e.g., OS-Atlas (Wu et al. 2024b), UI-TARS (Qin et al. 2025), CogAgent (Hong et al. 2024), ShowUI (Lin et al. 2024), SE-GUI (Yuan et al. 2025), GUI-R1 (Luo et al. 2025), InfiGUI-R1 (Liu et al. 2025b), GUI-G1 (Zhou et al. 2025)). All baseline results are collected from official papers or publicly released checkpoints.

4.2 Experimental Results and Analysis

Main Results

We evaluate our model, GUI-Eyes-3B, on three benchmarks—ScreenSpot (Cheng et al. 2024), ScreenSpot-v2 (Wu et al. 2024b), and ScreenSpot-Pro (Li et al. 2025) to assess its GUI grounding capabilities. As shown in Table 1 and Table 2, GUI-Eyes-3B achieves state-of-the-art performance across all three benchmarks, ranking first in overall accuracy and consistently outperforming prior methods on

Model	Training Samples	ScreenSpot Accuracy (%)				ScreenSpot-v2 Accuracy (%)			
		Mobile	Desktop	Web	Avg.	Mobile	Desktop	Web	Avg.
Proprietary Models									
GPT-4o	-	21.9	17.8	9.4	18.8	22.5	22.2	12.4	20.1
General Open-source Models									
Qwen2-VL-7B	-	50.3	40.4	27.4	42.9	39.4	50.1	27.7	39.8
Qwen2.5-VL-3B	-	-	-	-	55.5	55.5	44.0	39.1	46.9
Qwen2.5-VL-7B	-	-	-	-	84.7	92.8	78.4	85.4	86.5
GUI-Specific Models									
CogAgent-18B	222M	57.8	31.6	40.1	47.4	50.6	51.6	54.1	52.8
SeeClick-7B	1M	68.1	48.8	41.8	53.4	51.8	65.5	40.7	53.9
UGround-7B	10M	75.9	75.8	78.3	73.3	74.3	74.9	78.6	76.3
ShowUI-2B	256K	84.8	70.8	76.2	75.1	70.0	85.1	73.3	77.3
OSAtlas-4B	13M	56.2	74.9	69.9	68.5	74.9	56.9	70.7	68.5
OSAtlas-7B	13M	85.0	78.8	84.5	82.5	78.3	85.5	83.8	83.3
Aguvis-7B	1M	86.9	82.4	84.7	84.4	89.6	**86.8**	84.9	87.3
UI-TARS-2B	2M	85.0	81.4	79.8	82.3	87.9	81.4	82.9	84.7
Ours									
GUI-Eyes-3B	3K	**89.9**	**88.3**	**85.1**	**87.8**	**91.6**	86.2	**86.3**	**88.4**

Table 2: Comparison of model performance on ScreenSpot and ScreenSpot-v2. Results marked in **bold** represent the best performance.

Index	λ_{acc}	λ_{tool}	λ_{format}	Accuracy
1	0.4	0.5	0.1	43.2
2	0.55	0.35	0.1	44.5
3	**0.6**	**0.3**	**0.1**	**44.8**
4	0.65	0.25	0.1	42.6
5	0.7	0.2	0.1	41.2

Table 3: Grounding accuracy (%) on ScreenSpot-Pro under different reward-coefficient settings.

Reward Function	Text Acc	Icon Acc	Overall
Center Only	55.7	13.9	**39.7**
Overlap Only	57.9	15.4	**41.7**
Full (Ours)	62.8	15.7	**44.8**

Table 4: Grounding accuracy (%) on ScreenSpot-Pro using different tool reward functions.

both text and icon grounding tasks.

On ScreenSpot, it achieves an overall accuracy of 87.8%, leading across most platform settings. On ScreenSpot-v2, GUI-Eyes-3B reaches 88.4%, showing consistent improvements across all device categories. On ScreenSpot-Pro, it demonstrates strong generalization in complex domains, particularly in CAD (48.2%), development tools (70.8%), and scientific software (69.4%).

Compared to prior RL-based models such as GUI-R1-3B and GUI-G1-3B on the more challenging ScreenSpot-Pro benchmark, GUI-Eyes-3B delivers more balanced and robust performance, particularly in visually cluttered or low-saliency environments. Overall, these results validate the effectiveness and generalizability of our tool-augmented reasoning framework, underscoring its potential for real-world GUI interaction and automation systems.

Experimental Analysis

Text vs. Icon Performance. To further examine the behavior of the model in different query types, we conducted a comparative analysis of grounding performance in text-based versus icon-based queries. As illustrated in Figure 4, GUI-Eyes-3B achieves substantial improvements in text grounding accuracy across various domains in the ScreenSpot-Pro (Li et al. 2025) benchmark, with particularly notable gains in CAD and development tool scenarios. Despite being trained on only 3,000 labeled examples, our model surpasses several strong reinforcement learning baselines, including GUI-R1-3B and GUI-G1-3B, the latter trained with 17,000 RL samples, demonstrating strong generalization under limited supervision.

For icon-based queries, GUI-Eyes-3B demonstrates consistent performance gains, although the improvements are slightly smaller than those observed on text tasks. This suggests that the proposed method effectively handles both linguistic and symbolic grounding, leveraging the model's latent visual understanding to generalize across abstract, icon-driven interface elements. Future efforts could further enhance this capacity by incorporating targeted visual pretraining or lightweight symbol-aware augmentation strategies.

4.3 Ablation Study

Ablation Study on Reward Coefficient Sensitivity

We conduct an ablation study on **ScreenSpot-Pro** benchmark to examine the sensitivity of the model to different reward coefficients, specifically λ_{acc}, λ_{tool}, and λ_{format} in Equation 1. These parameters control the relative importance of accuracy, tool usage, and format rewards, respectively.

As shown in Table 3, the best configuration is obtained

with $\lambda_{\text{acc}} = 0.6$, $\lambda_{\text{tool}} = 0.3$, and $\lambda_{\text{format}} = 0.1$, achieving a grounding accuracy of **44.8%** on ScreenSpot-Pro. Different reward coefficients can influence the model's performance, particularly with respect to the trade-off between accuracy and tool utilization. Therefore, carefully tuning these weights is essential for achieving optimal results.

Ablation Study on Tool Reward Design

To better supervise the model's perceptual behavior, we formulate the tool reward R_{tool} with two key components (see Eq. 2): (1) *Center Proximity*, which measures the distance between the selected focus point and the target region; (2) *Region Overlap*, which quantifies the spatial intersection between the tool's operation area and the ground-truth bounding box.

We evaluate three reward variants on the ScreenSpot-Pro (Li et al. 2025) benchmark: (i) **Center Only**; (ii) **Overlap Only**; and (iii) **Full**, which combines both.

As summarized in Table 4, using either component in isolation results in limited gains, while the full reward yields significantly better grounding accuracy—especially for text queries. These results highlight the importance of jointly guiding both the initial attention point and the tool's coverage region to improve tool use and decision-making effectiveness.

The Impact of Tool Usage in Training

To evaluate the contribution of the tool-based perception mechanism in our framework, we conduct an ablation study by progressively disabling components of the tool learning pipeline. Specifically, we compare the following variants:

- **No Tool Usage:** The agent is restricted from invoking any visual tools during inference and must rely solely on the raw GUI screenshot. This setting corresponds to a cropping ratio of $\alpha = 0$ in Figure 5, serving as the baseline for evaluating the benefit of tool-based perception.
- **No Tool Training:** Visual tools remain accessible, but the tool invocation policy is no longer learned. Instead, we adopt a fixed heuristic inspired by the DiMo-GUI framework (Wu et al. 2025a), where the tool input is generated by cropping a region centered on the prediction from a strong pretrained model, GUI-R1-3B. The cropping ratio α is varied to examine the effect of input scale

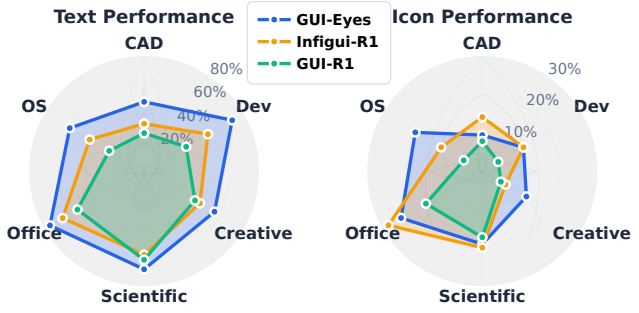

Figure 4: Radar plots comparing the grounding accuracy of GUI-Eyes-3B, Infigui-R1-3B, and GUI-R1-3B on text-based (left) and icon-based (right) queries across domains in the ScreenSpot-Pro benchmark.

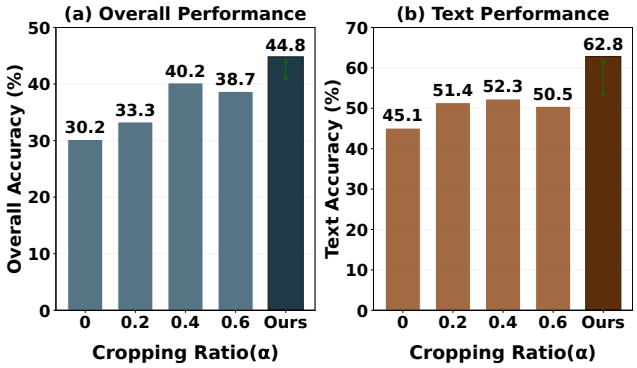

Figure 5: Ablation study comparing different tool-usage strategies on ScreenSpot-Pro. $\alpha = 0$ denotes no tool usage. $\alpha \in \{0.2, 0.4, 0.6\}$ are fixed cropping ratios generated from GUI-R1-3B predictions (static cropping). "Ours" refers to our GUIEyes-3B model, which dynamically learns when and how much to crop.

(e.g., $\alpha = 0.2, 0.4, 0.6$).

We select GUI-R1-3B as a comparison model because it is trained with reinforcement learning on the same data scale (3,000 samples), ensuring fair comparability. The results are summarized in Figure 5. As shown, the agent without any tool usage achieves the lowest performance (30.2% overall accuracy and 45.1% on text-based queries). When incorporating static cropping based on the pretrained model's prediction, performance improves across different cropping scales, with the best result at $\alpha = 0.4$ achieving 40.2% overall and 52.3% text accuracy. In contrast, our method outperforms these static strategies, achieving 44.8% overall accuracy and 62.8% on text queries. These findings highlight the importance of learning a dynamic tool policy to support perception refinement and robust decision-making.

5 Conclusion

In this work, we propose GUI-Eyes, a reinforcement learning framework that guides multimodal language models to perform structured perception-to-decision reasoning in graphical user interface (GUI) environments. The framework introduces an active perception mechanism, enabling the model to dynamically decide whether to invoke visual tools—such as cropping and zooming—and to configure them adaptively during inference, thereby acquiring more focused and task-relevant observations. To support effective tool usage, we design a spatially aware reward function that combines location proximity and region overlap, offering dense and stable optimization feedback. Extensive experiments demonstrate that GUI-Eyes-3B, trained on only 3,000 labeled samples, achieves 44.8% accuracy on the ScreenSpot-Pro benchmark, significantly outperforming both supervised and RL-based baselines. These results highlight the framework's strong generalization ability and data efficiency, underscoring its potential for building scalable and perceptually grounded GUI agents.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (NO. 2024YFE0203200), and the National Nature Science Foundation of China (NO. U24A20329).

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

An, H.; Hu, W.; Huang, S.; Huang, S.; Li, R.; Liang, Y.; Shao, J.; Song, Y.; Wang, Z.; Yuan, C.; et al. 2025. Ai flow: Perspectives, scenarios, and approaches. *arXiv preprint arXiv:2506.12479*.

Anthropic. 2024. Developing a computer use model. https://www.anthropic.com/news/developing-computer-use. Accessed: 2025-04-12.

Bai, S.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; Song, S.; Dang, K.; Wang, P.; Wang, S.; Tang, J.; et al. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Chai, Y.; Huang, S.; Niu, Y.; Xiao, H.; Liu, L.; Zhang, D.; Gao, P.; Ren, S.; and Li, H. 2024. Amex: Android multi-annotation expo dataset for mobile gui agents. *arXiv preprint arXiv:2407.17490*.

Cheng, K.; Sun, Q.; Chu, Y.; Xu, F.; Li, Y.; Zhang, J.; and Wu, Z. 2024. Seeclick: Harnessing gui grounding for advanced visual gui agents. *arXiv preprint arXiv:2401.10935*.

Deng, X.; Gu, Y.; Zheng, B.; Chen, S.; Stevens, S.; Wang, B.; Sun, H.; and Su, Y. 2023. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36: 28091–28114.

Feng, L.; Xue, Z.; Liu, T.; and An, B. 2025. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*.

Gao, L.; Zhang, L.; and Xu, M. 2025. UIShift: Enhancing VLM-based GUI Agents through Self-supervised Reinforcement Learning. *arXiv preprint arXiv:2505.12493*.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Gu, J.; Ai, Q.; Wang, Y.; Bu, P.; Xing, J.; Zhu, Z.; Jiang, W.; Wang, Z.; Zhao, Y.; Zhang, M.-L.; et al. 2025. Mobile-R1: Towards Interactive Reinforcement Learning for VLM-Based Mobile Agent via Task-Level Rewards. *arXiv preprint arXiv:2506.20332*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Gur, I.; Furuta, H.; Huang, A.; Safdari, M.; Matsuo, Y.; Eck, D.; and Faust, A. 2023. A real-world webagent with planning, long context understanding, and program synthesis. *arXiv preprint arXiv:2307.12856*.

Hong, W.; Wang, W.; Lv, Q.; Xu, J.; Yu, W.; Ji, J.; Wang, Y.; Wang, Z.; Dong, Y.; Ding, M.; et al. 2024. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14281–14290.

Huang, J.; Zeng, Z.; Han, W.; Zhong, Y.; Zheng, L.; Fu, S.; Chen, J.; and Ma, L. 2025a. Scaletrack: Scaling and back-tracking automated gui agents. *arXiv preprint arXiv:2505.00416*.

Huang, Z.; Ji, Y.; Rajan, A. S.; Cai, Z.; Xiao, W.; Hu, J.; and Lee, Y. J. 2025b. VisualToolAgent (VisTA): A Reinforcement Learning Framework for Visual Tool Selection. *arXiv preprint arXiv:2505.20289*.

Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; Radford, A.; et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Kim, G.; Baldi, P.; and McAleer, S. 2023. Language models can solve computer tasks. *Advances in Neural Information Processing Systems*, 36: 39648–39677.

Li, K.; Meng, Z.; Lin, H.; Luo, Z.; Tian, Y.; Ma, J.; Huang, Z.; and Chua, T.-S. 2025. Screenspot-pro: Gui grounding for professional high-resolution computer use. *arXiv preprint arXiv:2504.07981*.

Li, W.; Bishop, W.; Li, A.; Rawles, C.; Campbell-Ajala, F.; Tyamagundlu, D.; and Riva, O. 2024. On the effects of data scale on computer control agents. *arXiv e-prints*, arXiv–2406.

Lin, K. Q.; Li, L.; Gao, D.; Yang, Z.; Bai, Z.; Lei, W.; Wang, L.; and Shou, M. Z. 2024. Showui: One vision-language-action model for generalist gui agent. In *NeurIPS 2024 Workshop on Open-World Agents*, volume 1.

Liu, Y.; Li, P.; Wei, Z.; Xie, C.; Hu, X.; Xu, X.; Zhang, S.; Han, X.; Yang, H.; and Wu, F. 2025a. InfiGUIAgent: A Multimodal Generalist GUI Agent with Native Reasoning and Reflection. *arXiv preprint arXiv:2501.04575*.

Liu, Y.; Li, P.; Xie, C.; Hu, X.; Han, X.; Zhang, S.; Yang, H.; and Wu, F. 2025b. Infigui-r1: Advancing multimodal gui agents from reactive actors to deliberative reasoners. *arXiv preprint arXiv:2504.14239*.

Loshchilov, I.; and Hutter, F. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Lu, Z.; Chai, Y.; Guo, Y.; Yin, X.; Liu, L.; Wang, H.; Xiao, H.; Ren, S.; Xiong, G.; and Li, H. 2025. UI-R1: Enhancing Efficient Action Prediction of GUI Agents by Reinforcement Learning. *arXiv preprint arXiv:2503.21620*.

Luo, R.; Wang, L.; He, W.; and Xia, X. 2025. Gui-r1: A generalist r1-style vision-language action model for gui agents. *arXiv preprint arXiv:2504.10458*.

Muennighoff, N.; Yang, Z.; Shi, W.; Li, X. L.; Fei-Fei, L.; Hajishirzi, H.; Zettlemoyer, L.; Liang, P.; Candès, E.; and Hashimoto, T. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*.

Qin, Y.; Ye, Y.; Fang, J.; Wang, H.; Liang, S.; Tian, S.; Zhang, J.; Li, J.; Li, Y.; Huang, S.; et al. 2025. UI-TARS: Pioneering Automated GUI Interaction with Native Agents. *arXiv preprint arXiv:2501.12326*.

Rawles, C.; Li, A.; Rodriguez, D.; Riva, O.; and Lillicrap, T. 2023. Androidinthewild: A large-scale dataset for android device control. *Advances in Neural Information Processing Systems*, 36: 59708–59728.

Shao, J.; and Li, X. 2025. Ai flow at the network edge. *IEEE Network*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Song, H.; Jiang, J.; Min, Y.; Chen, J.; Chen, Z.; Zhao, W. X.; Fang, L.; and Wen, J.-R. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Su, A.; Wang, H.; Ren, W.; Lin, F.; and Chen, W. 2025. Pixel reasoner: Incentivizing pixel-space reasoning with curiosity-driven reinforcement learning. *arXiv preprint arXiv:2505.15966*.

Sun, Q.; Cheng, K.; Ding, Z.; Jin, C.; Wang, Y.; Xu, F.; Wu, Z.; Jia, C.; Chen, L.; Liu, Z.; et al. 2024. Os-genesis: Automating gui agent trajectory construction via reverse task synthesis. *arXiv preprint arXiv:2412.19723*.

Tang, F.; Xu, H.; Zhang, H.; Chen, S.; Wu, X.; Shen, Y.; Zhang, W.; Hou, G.; Tan, Z.; Yan, Y.; et al. 2025. A survey on (m) llm-based gui agents. *arXiv preprint arXiv:2504.13865*.

Team, Q. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Wang, J.; Xu, H.; Jia, H.; Zhang, X.; Yan, M.; Shen, W.; Zhang, J.; Huang, F.; and Sang, J. 2024a. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. *Advances in Neural Information Processing Systems*, 37: 2686–2710.

Wang, J.; Xu, H.; Ye, J.; Yan, M.; Shen, W.; Zhang, J.; Huang, F.; and Sang, J. 2024b. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *arXiv preprint arXiv:2401.16158*.

Wang, K.; Zhang, P.; Wang, Z.; Wang, Q.; Gao, Y.; Li, L.; Yang, Z.; Wan, C.; Chen, H.; Lu, Y.; et al. 2025a. VAGEN: Training VLM Agents with Multi-Turn Reinforcement Learning. 2025. *URl: https://github. com/RAGEN-AI/VAGEN*.

Wang, P.; Bai, S.; Tan, S.; Wang, S.; Fan, Z.; Bai, J.; Chen, K.; Liu, X.; Wang, J.; Ge, W.; et al. 2024c. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Wang, Z.; Wang, K.; Wang, Q.; Zhang, P.; Li, L.; Yang, Z.; Jin, X.; Yu, K.; Nguyen, M. N.; Liu, L.; et al. 2025b. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*.

Wei, Z.; Yao, W.; Liu, Y.; Zhang, W.; Lu, Q.; Qiu, L.; Yu, C.; Xu, P.; Zhang, C.; Yin, B.; et al. 2025. Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning. *arXiv preprint arXiv:2505.16421*.

Wu, H.; Chen, H.; Cai, Y.; Liu, C.; Ye, Q.; Yang, M.-H.; and Wang, Y. 2025a. DiMo-GUI: Advancing Test-time Scaling in GUI Grounding via Modality-Aware Visual Reasoning. *arXiv preprint arXiv:2507.00008*.

Wu, Q.; Cheng, K.; Yang, R.; Zhang, C.; Yang, J.; Jiang, H.; Mu, J.; Peng, B.; Qiao, B.; Tan, R.; et al. 2025b. GUI-Actor: Coordinate-Free Visual Grounding for GUI Agents. *arXiv preprint arXiv:2506.03143*.

Wu, Q.; Xu, W.; Liu, W.; Tan, T.; Liu, J.; Li, A.; Luan, J.; Wang, B.; and Shang, S. 2024a. Mobilevlm: A vision-language model for better intra-and inter-ui understanding. *arXiv preprint arXiv:2409.14818*.

Wu, Z.; Wu, Z.; Xu, F.; Wang, Y.; Sun, Q.; Jia, C.; Cheng, K.; Ding, Z.; Chen, L.; Liang, P. P.; et al. 2024b. Os-atlas: A foundation action model for generalist gui agents. *arXiv preprint arXiv:2410.23218*.

Xu, Y.; Li, C.; Zhou, H.; Wan, X.; Zhang, C.; Korhonen, A.; and Vulić, I. 2025. Visual Planning: Let's Think Only with Images. *arXiv preprint arXiv:2505.11409*.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

You, K.; Zhang, H.; Schoop, E.; Weers, F.; Swearngin, A.; Nichols, J.; Yang, Y.; and Gan, Z. 2024. Ferret-ui: Grounded mobile ui understanding with multimodal llms. In *European Conference on Computer Vision*, 240–255. Springer.

Yuan, X.; Zhang, J.; Li, K.; Cai, Z.; Yao, L.; Chen, J.; Wang, E.; Hou, Q.; Chen, J.; Jiang, P.-T.; et al. 2025. Enhancing Visual Grounding for GUI Agents via Self-Evolutionary Reinforcement Learning. *arXiv preprint arXiv:2505.12370*.

Zhang, C.; He, S.; Qian, J.; Li, B.; Li, L.; Qin, S.; Kang, Y.; Ma, M.; Liu, G.; Lin, Q.; et al. 2024. Large language model-brained gui agents: A survey, 2025. *URL https://arxiv. org/abs/2411.18279*.

Zhang, C.; Yang, Z.; Liu, J.; Li, Y.; Han, Y.; Chen, X.; Huang, Z.; Fu, B.; and Yu, G. 2025. Appagent: Multimodal agents as smartphone users. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, 1–20.

Zheng, Z.; Yang, M.; Hong, J.; Zhao, C.; Xu, G.; Yang, L.; Shen, C.; and Yu, X. 2025. DeepEyes: Incentivizing" Thinking with Images" via Reinforcement Learning. *arXiv preprint arXiv:2505.14362*.

Zhou, S.; Xu, F. F.; Zhu, H.; Zhou, X.; Lo, R.; Sridhar, A.; Cheng, X.; Ou, T.; Bisk, Y.; Fried, D.; et al. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

Zhou, Y.; Dai, S.; Wang, S.; Zhou, K.; Jia, Q.; et al. 2025. GUI-G1: Understanding r1-zero-like training for visual grounding in gui agents. *arXiv preprint arXiv:2505.15810*.