



GUI-G²: GAUSSIAN REWARD MODELING FOR GUI GROUNDING

Fei Tang^{1,2*}, Zhangxuan Gu², Zhengxi Lu¹, Xuyang Liu²
 Shuheng Shen², Changhua Meng², Wen Wang¹, Wenqi Zhang¹
 Yongliang Shen^{1†}, Weiming Lu¹, Jun Xiao¹, Yueting Zhuang¹

¹Zhejiang University, ²Ant Group
 {flysugar, syl}@zju.edu.cn shuheng.ssh@antgroup.com

GitHub: <https://github.com/zju-real/GUI-G2>
 Project: <https://zju-real.github.io/GUI-G2>

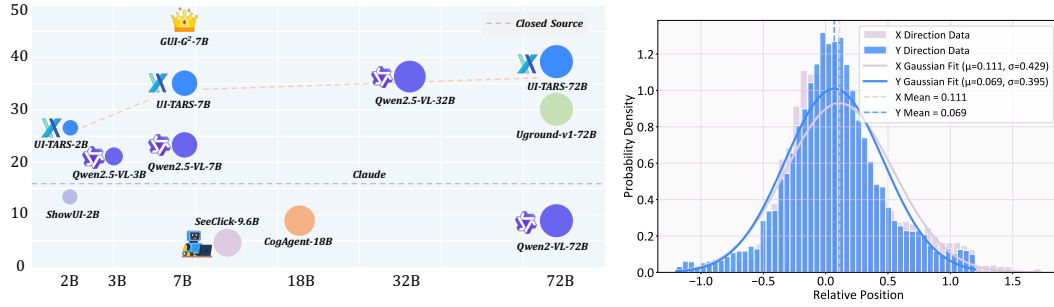


Figure 1: GUI grounding performance and human click behavior. Left: Performance comparison of various models on ScreenSpot-Pro. Right: Human click distribution from AITW (Rawles et al., 2023) reveals natural Gaussian patterns around target centers ($\mu = 0.111, \sigma = 0.429$), validating our design choice of continuous Gaussian rewards over discrete binary feedback.

ABSTRACT

Graphical User Interface (GUI) grounding maps natural language instructions to precise interface locations for autonomous interaction. Current reinforcement learning approaches use binary rewards that treat elements as hit-or-miss targets, creating sparse signals that ignore the continuous nature of spatial interactions. Motivated by human clicking behavior that naturally forms Gaussian distributions centered on target elements, we introduce GUI Gaussian Grounding Rewards (GUI-G²), a principled reward framework that models GUI elements as continuous Gaussian distributions across the interface plane. GUI-G² incorporates two synergistic mechanisms: Gaussian point rewards model precise localization through exponentially decaying distributions centered on element centroids, while coverage rewards assess spatial alignment by measuring the overlap between predicted Gaussian distributions and target regions. To handle diverse element scales, we develop an adaptive variance mechanism that calibrates reward distributions based on element dimensions. This framework transforms GUI grounding from sparse binary classification to dense continuous optimization, where Gaussian distributions generate rich gradient signals that guide models toward optimal interaction positions. Extensive experiments across ScreenSpot, ScreenSpot-v2, and ScreenSpot-Pro benchmarks demonstrate that GUI-G², substantially outperforms state-of-the-art method UI-TARS-72B, with the most significant improvement of 24.7% on ScreenSpot-Pro. Our analysis reveals that continuous modeling provides superior robustness to interface variations and enhanced generalization to unseen layouts, establishing a new paradigm for spatial reasoning in GUI interaction tasks.

* This work was done when the first author was an intern at Ant Group.

† Corresponding author.

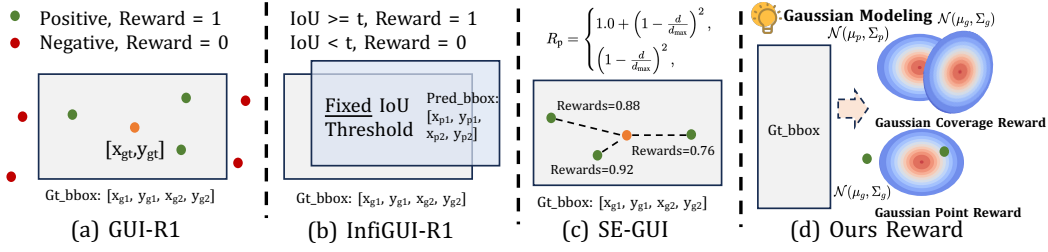


Figure 2: Comparison of reward modeling strategies. (a-c) Existing methods treat GUI elements as abstract points with binary or distance-based rewards, while (d) our Gaussian approach provides continuous point and coverage rewards that naturally align with human clicking behavior.

1 INTRODUCTION

Autonomous GUI agents are revolutionizing human-computer interaction by allowing users to control interfaces with natural language across various applications (Gou et al., 2024; Tang et al., 2025b; Cheng et al., 2024). As the core of these systems, GUI grounding, is the fundamental capability to accurately map natural language instructions to precise pixel coordinates on interface elements (Tang et al., 2025a; Cheng et al., 2024; Lin et al., 2024; Wu et al., 2025).

Recent advances in GUI grounding have increasingly adopted reinforcement learning frameworks (Lu et al., 2025; Luo et al., 2025; Liu et al., 2025d). However, current approaches rely on binary reward systems (Lu et al., 2025; Luo et al., 2025; Yuan et al., 2025; Zhou et al., 2025) that assign rewards of 1 for coordinates within target bounding boxes and 0 otherwise. This formulation treats GUI interactions as binary hit-or-miss problems, creating sparse learning signals where predictions one pixel outside target regions receive the same zero reward as complete failures (Figure 2, a-b). The binary paradigm ignores two critical aspects of interface interaction: first, clicking quality varies continuously with distance from element centers, and second, interface elements are inherently two-dimensional regions with spatial structure, not abstract points (Figure 2, a-c). This mismatch between discrete optimization and the continuous geometric nature of GUI interactions severely limits learning efficiency, particularly during early training when models need dense feedback to develop appropriate grounding behaviors.

This discrete approach contradicts empirical evidence from human behavior. Analysis of the AITW dataset (Rawles et al., 2023) reveals that users’ clicks naturally form Gaussian distributions centered on target elements (Figure 3, right), consistent with Fitts’ Law (Fitts, 1954; and, 1992). This pattern demonstrates that spatial targeting inherently follows continuous probability distributions, with click density decreasing smoothly from element centers to edges. Current binary mechanisms completely ignore this fundamental characteristic of human-computer interaction.

Building on this insight, we introduce GUI-G² (GUI Gaussian Grounding Rewards), a principled framework that fundamentally reconceptualizes GUI grounding by modeling clicking points as smooth probability distributions across the interface plane. Rather than treating elements as discrete hit-or-miss targets, GUI-G² represents them as continuous Gaussian distributions that provide rich spatial information and dense learning signals. This approach comprises two complementary mechanisms: First, we design point-based rewards that decrease smoothly with distance from element centers, encouraging precise localization while maintaining continuous gradients. Second, we introduce coverage-based rewards that measure the spatial overlap between predicted click distributions and target element regions, ensuring comprehensive element targeting.

To accommodate varying element scales, we introduce an adaptive variance mechanism that dynamically adjusts reward distributions according to element dimensions. This ensures consistent learning signals across GUI components while maintaining their distinct geometric properties. GUI-G² transforms GUI grounding from sparse binary optimization to dense continuous reasoning, enabling models to learn fine-grained spatial relationships and develop more robust interaction strategies.

Extensive evaluation on ScreenSpot (Cheng et al., 2024), ScreenSpot-v2 (Wu et al., 2024), and ScreenSpot-Pro (Li et al., 2025) benchmarks demonstrates that our approach achieves substantial

improvements over state-of-the-art methods, with accuracy gains up to 4.1%, 3.3%, and 24.7% respectively. Our analysis reveals superior robustness to interface variations and enhanced generalization to unseen layouts, confirming that continuous spatial modeling provides more fundamental and transferable representations than discrete alternatives. Comprehensive ablation studies validate the synergistic contributions of both Gaussian components and the critical importance of adaptive variance mechanisms for handling interface diversity.

Our contributions are threefold:

- We introduce GUI-G², a principled approach that models GUI interactions as continuous spatial processes, fundamentally transforming reward design from discrete binary signals to geometrically-aware continuous feedback that captures the inherent planar nature of interface elements.
- We propose a novel dual-component reward system comprising Gaussian point rewards for precise localization and Gaussian coverage rewards for regional assessment, enhanced with adaptive variance mechanisms that automatically calibrate distributions based on element dimensions.
- We demonstrate through extensive experiments that GUI-G² achieves substantial improvements, with accuracy of 92.0% on ScreenSpot, 93.3% on ScreenSpot-v2, and 47.5% on ScreenSpot-Pro, while exhibiting superior robustness and generalization compared to discrete reward approaches.

2 RELATED WORK

2.1 GUI AGENTS

GUI agents are intelligent systems that can understand and interact with graphical user interfaces through natural language instructions, enabling automated execution of complex computer tasks (Gou et al., 2024; Zhang et al., 2025a; Tang et al., 2025b; Sun et al., 2025; Shen et al., 2023; Hong et al., 2024; Yang et al., 2024). These approaches can be broadly categorized into two main paradigms: **(1) Expert Design-Driven Workflow Paradigm**: These approaches typically leverage closed-source multimodal large language models and construct workflows through expertly designed fine-grained modules such as planners (Wang et al., 2024b; Zhang et al., 2024) and grounders (Gou et al., 2024; Liu et al., 2024; Lin et al., 2024; Wu et al., 2024). The Mobile-Agent series (Wang et al., 2025; 2024a;b), AppAgent series (Zhang et al., 2023; Li et al., 2024; Jiang et al., 2025; Xie et al., 2025), and UFO series (Zhang et al., 2024; 2025b) all accomplish various tasks through these workflow-based approaches. These GUI agents typically consist of planners and grounders, where planners usually employ closed-source large language models such as GPT-4o (OpenAI, 2024) and Claude (Anthropic, 2024) for task planning. For grounding components, there are two main approaches: one utilizes HTML and DOM tree structures for screen understanding (Rawles et al., 2023; Zhang et al., 2023), while the other employs visual tools such as OCR (Du et al., 2020), SAM (Kirillov et al., 2023), and Omniparser (Lu et al., 2024) for more effective screen understanding and element localization. However, this reliance on pre-programmed workflows, driven by human expertise, makes frameworks inherently non-scalable, consuming substantial manual effort and proving difficult to extend to new domains (Qin et al., 2025). **(2) Data-Driven Training Paradigm**: These approaches employ specialized MLLMs trained specifically for GUI understanding and interaction through data-driven methodologies (Qin et al., 2025; Gou et al., 2024; Lin et al., 2024; Cheng et al., 2024; Tang et al., 2025a; Wu et al., 2024). These works achieve GUI-specific capabilities by collecting large-scale GUI corpora for fine-tuning to develop models tailored for GUI tasks. For example, UI-TARS (Qin et al., 2025) develops an end-to-end native GUI agent through large-scale GUI screenshots for enhanced perception and action traces for unified action modeling across platforms. However, due to the limitations of supervised fine-tuning (Chu et al., 2025), these methods still face generalization challenges when encountering novel interface scenarios (Luo et al., 2025; Lu et al., 2025).

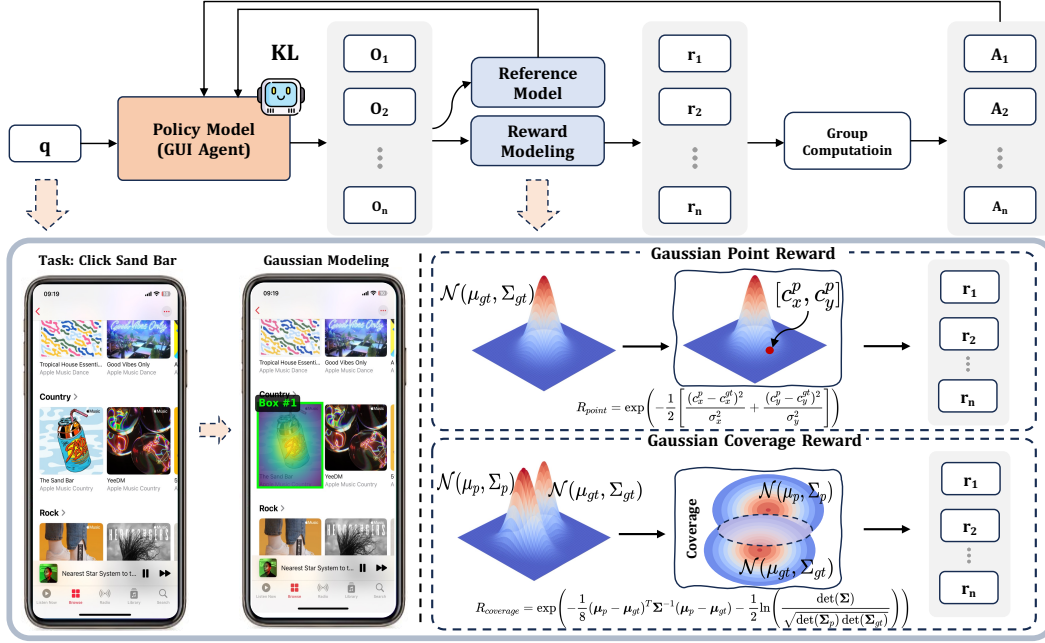


Figure 3: **GUI Gaussian Grounding Rewards (GUI-G²)**. Our framework transforms GUI grounding through continuous Gaussian modeling. Given a task instruction and screenshot, the policy model generates multiple predictions that are evaluated using our dual reward mechanism. Gaussian Point Rewards assess localization precision while Gaussian Coverage Rewards measure spatial overlap, together providing dense learning signals that guide policy optimization.

2.2 REINFORCEMENT FINE-TUNING

Since the release of DeepSeek-R1 (DeepSeek-AI, 2025), rule-based reward reinforcement learning has been applied across various domains, such as video understanding (Feng et al., 2025) and multimodal reasoning (Shen et al., 2025). Researchers have begun applying this approach to GUI tasks. GUI-R1 (Luo et al., 2025) and UI-R1 (Lu et al., 2025) apply verifiable reward paradigms to GUI tasks, representing pioneering efforts in this direction while demonstrating the potential of RFT. InfiGUI-R1 (Liu et al., 2025d) similarly follows the R1 paradigm, employing two-stage training to inject reasoning capabilities into the model. GUI-G1 (Zhou et al., 2025) reanalyzes existing problems in current R1-based GUI agents and designs controllable box size rewards for GUI grounding tasks, while incorporating difficulty coefficient factors based on box size using the GRPO (Shao et al., 2024) algorithm to enable better learning. SE-GUI (Yuan et al., 2025) proposes self-evolution approaches and continuous rewards to guide model learning. However, most previous methods treat GUI elements as discrete point requiring perfect targeting and provide only sparse hit-or-miss feedback, struggling to provide effective guidance for model learning during the early stages of training. We address the limitations by proposing a Gaussian continuous reward mechanism, which provides dense and informative feedback to guide model learning more effectively.

3 METHOD

We introduce GUI-G² (GUI Gaussian Grounding Rewards), a principled framework that reformulates GUI grounding rewards from discrete binary signals to continuous Gaussian distributions. As illustrated in Figure 3, our approach comprises three key innovations: (1) Gaussian point rewards that model localization precision, (2) Gaussian coverage rewards that capture spatial overlap, and (3) an adaptive variance mechanism that scales with element dimensions. This continuous formulation addresses the fundamental limitation of binary rewards by providing learning signals for near-misses through smooth gradients throughout the spatial domain.

3.1 PROBLEM FORMULATION

GUI grounding maps natural language instructions to pixel-level targets on graphical interfaces. Given a screenshot s and instruction i , the model must predict a bounding box $\mathbf{b}^p = [x_1^p, y_1^p, x_2^p, y_2^p]$ that localizes the element described by i , where (x_1, y_1) and (x_2, y_2) denote the top-left and bottom-right corners respectively. The ground truth is annotated as $\mathbf{b}^{gt} = [x_1^{gt}, y_1^{gt}, x_2^{gt}, y_2^{gt}]$.

In the reinforcement learning formulation, the model generates a sequence of tokens representing the predicted bounding box coordinates. The standard evaluation criterion checks whether the predicted center $(c_x^p, c_y^p) = (\frac{x_1^p + x_2^p}{2}, \frac{y_1^p + y_2^p}{2})$ falls within \mathbf{b}^{gt} . Our reward function $R(\mathbf{b}^p, \mathbf{b}^{gt})$ transforms this discrete success metric into continuous spatial feedback. Unlike binary rewards that provide no gradient for near-misses, GUI-G² generates dense learning signals that vary smoothly with prediction quality, enabling more efficient policy optimization through richer supervision.

3.2 GAUSSIAN REWARD MODELING

We model GUI elements as 2D Gaussian distributions to capture the continuous nature of spatial interactions. This approach transforms discrete bounding boxes into smooth probability distributions that naturally encode spatial uncertainty and provide rich gradient information.

Gaussian Representation. For each GUI element with bounding box $\mathbf{b} = [x_1, y_1, x_2, y_2]$, we construct a 2D Gaussian distribution:

$$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{2\pi\sqrt{|\boldsymbol{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right) \quad (1)$$

where $\mathbf{x} = (x, y)$ represents a position in the 2D interface space, $\boldsymbol{\mu} = (c_x, c_y) = (\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2})$ is the element’s geometric center, and $\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}$ is a diagonal covariance matrix. The diagonal structure assumes independence between x and y dimensions, simplifying computation while maintaining expressiveness.

Gaussian Point Rewards. The point reward evaluates localization precision by measuring how well the predicted center aligns with the target element’s Gaussian distribution. Given a predicted bounding box with center $\boldsymbol{\mu}_p = (c_x^p, c_y^p)$ and ground truth center $\boldsymbol{\mu}_{gt} = (c_x^{gt}, c_y^{gt})$, we compute:

$$R_{point} = \mathcal{N}(\boldsymbol{\mu}_p; \boldsymbol{\mu}_{gt}, \boldsymbol{\Sigma}_{gt}) = \exp\left(-\frac{1}{2}\left[\frac{(c_x^p - c_x^{gt})^2}{\sigma_x^{gt^2}} + \frac{(c_y^p - c_y^{gt})^2}{\sigma_y^{gt^2}}\right]\right) \quad (2)$$

This formulation provides several key properties. First, the reward reaches its maximum value of 1 when the predicted center perfectly aligns with the ground truth. Second, it decreases smoothly and exponentially with distance, ensuring continuous gradients throughout the spatial domain. Third, the rate of decay is controlled by the variance parameters, allowing flexible adaptation to different element characteristics.

Gaussian Coverage Rewards. While point rewards optimize for center alignment, GUI interactions often succeed when clicking anywhere within element boundaries. Coverage rewards capture this regional aspect by measuring the spatial overlap between predicted and target Gaussian distributions. We quantify this overlap using the Bhattacharyya coefficient:

$$BC(\mathcal{N}_p, \mathcal{N}_{gt}) = \int \sqrt{\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p) \cdot \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{gt}, \boldsymbol{\Sigma}_{gt})} d\mathbf{x} \quad (3)$$

For Gaussian distributions, this integral has a closed-form solution:

$$R_{coverage} = \exp\left(-\frac{1}{8}(\boldsymbol{\mu}_p - \boldsymbol{\mu}_{gt})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_p - \boldsymbol{\mu}_{gt}) - \frac{1}{2} \ln\left(\frac{\det(\boldsymbol{\Sigma})}{\sqrt{\det(\boldsymbol{\Sigma}_p) \det(\boldsymbol{\Sigma}_{gt})}}\right)\right) \quad (4)$$

where $\boldsymbol{\Sigma} = \frac{\boldsymbol{\Sigma}_p + \boldsymbol{\Sigma}_{gt}}{2}$ is the average covariance. The first term penalizes center misalignment weighted by the combined uncertainty, while the second term measures size and shape similarity between distributions.

Adaptive Variance Mechanism. GUI elements span diverse scales, from tiny icons to full-screen panels. Fixed variance parameters would either over-constrain large elements or under-constrain small ones. We introduce an adaptive mechanism that scales variance with element dimensions:

$$\sigma_x = \alpha \cdot (x_2 - x_1), \quad \sigma_y = \alpha \cdot (y_2 - y_1) \quad (5)$$

where α is a scaling factor that controls the relative influence of element size on the standard deviations. The intuition behind this scaling is straightforward: larger elements naturally tolerate greater spatial uncertainty in user interactions. A small icon requires precise targeting within a few pixels, while a large button or panel can be successfully activated across a much wider region. By making the Gaussian spread proportional to element size, we ensure that the reward function respects this natural interaction pattern. The adaptive mechanism applies to both point and coverage rewards, ensuring consistent behavior across the interface hierarchy.

3.3 REINFORCEMENT LEARNING WITH GUI-G²

To leverage the complementary strengths of precise localization and spatial coverage, we combine both reward components:

$$R_{total} = \nu \cdot R_{point} + \gamma \cdot R_{coverage} \quad (6)$$

where ν and γ balance the contribution of each component. The point reward drives the model toward accurate center positioning, while the coverage reward ensures appropriate spatial extent. This dual objective mirrors human interaction patterns: users aim for element centers but can successfully interact anywhere within boundaries.

We integrate GUI-G² into Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which estimates advantages using multiple sampled responses. For each instruction, we sample N predictions and compute their rewards under GUI-G². The advantage for response i is:

$$A_i = \frac{R_{total}(\tau_i) - \text{mean}(\{R_{total}(\tau_j)\}_{j=1}^N)}{\text{std}(\{R_{total}(\tau_j)\}_{j=1}^N)} \quad (7)$$

This normalization ensures stable gradients across different element types and sizes. The policy optimization objective becomes:

$$\mathcal{J}(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta_{old}}} \left[\sum_t \min(r_t(\theta) A_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) A_t) - \beta \mathbb{D}_{KL}[\pi_{\theta} \parallel \pi_{ref}] \right] \quad (8)$$

where $r_t(\theta) = \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta_{old}}(a_t | s_t)}$ is the probability ratio, ϵ controls the trust region, and β weights the KL regularization. The continuous nature of GUI-G² rewards fundamentally transforms the optimization landscape. While binary rewards create a discontinuous surface with sharp cliffs at bounding box edges, our Gaussian formulation produces smooth gradients everywhere in the spatial domain. This smoothness is crucial during early training: when predictions are far from targets, the exponentially decaying Gaussian signals provide clear directional guidance toward improvement.

4 EXPERIMENTS

4.1 EXPERIMENT SETUP

Implementation Details. We implement GUI-G² using Qwen2.5-VL-7B-Instruct (Bai et al., 2025) as the base model within the VLM-R1 framework (Shen et al., 2025). Training is conducted on 8 NVIDIA A100-80G GPUs for one epoch with the following hyperparameters: learning rate 1e-6, global batch size 8, 8 sampled responses per instruction, and KL penalty $\beta = 0.04$. For the Gaussian reward mechanism, we set $\alpha = 0.5$. We employ Flash Attention 2 (Dao, 2023) and use bfloat16 precision with gradient checkpointing. During inference, we use deterministic generation with temperature 0. Unless otherwise specified, we set ν and γ to 1.0. More training details are provided in Table 7. The training and inference prompt templates are shown in A.2.

Training Dataset and Evaluation Benchmarks. Our training data comprises approximately 100K GUI grounding instances sampled from four major datasets: Widget Captioning (Cheng et al., 2024),

Model	ScreenSpot v1 Accuracy (%)						SSv1 Avg.	SSv2 Avg.
	Mobile		Desktop		Web			
	Text	Icon	Text	Icon	Text	Icon		
Proprietary Models								
GPT-4o	30.5	23.2	20.6	19.4	11.1	7.8	18.8	20.1
Claude Computer Use	-	-	-	-	-	-	83.0	-
General Open-source Models								
Qwen2-VL-7B	61.3	39.3	52.0	45.0	33.0	21.8	42.9	-
Qwen2.5-VL-3B	-	-	-	-	-	-	55.5	80.9
Qwen2.5-VL-7B	-	-	-	-	-	-	84.7	88.8
GUI-specific Models (SFT)								
CogAgent-18B	67.0	24.0	74.2	20.0	70.4	28.6	47.4	-
SeeClick-9.6B	78.0	52.0	72.2	30.0	55.7	32.5	53.4	55.1
UGround-7B	82.8	60.3	82.5	63.6	80.4	70.4	73.3	76.3
OS-Atlas-7B	93.0	72.9	91.8	62.9	90.9	74.3	82.5	-
ShowUI-2B	92.3	75.5	76.3	61.1	81.7	63.6	75.1	77.3
FOCUS-2B	90.1	78.2	80.9	65.0	81.7	68.5	77.4	-
Aguvis-7B	95.6	77.7	93.8	67.1	88.3	75.2	84.4	80.5
Aguvis-72B	94.5	85.2	95.4	77.9	91.3	85.9	89.2	-
UI-TARS-2B	93.0	75.5	90.7	68.6	84.3	74.8	82.3	84.7
UI-TARS-7B	94.5	85.2	95.9	85.7	90.0	83.5	89.5	91.6
UI-TARS-72B	94.9	82.5	89.7	88.6	88.7	85.0	88.4	90.3
GUI-Actor-7B	94.9	82.1	91.8	80.0	91.3	85.4	88.3	92.1
JEDI-3B	-	-	-	-	-	-	-	88.6
JEDI-7B	-	-	-	-	-	-	-	91.7
GUI-specific Models (RL)								
UI-R1-3B	95.6	84.7	90.2	59.3	85.2	73.3	83.3	85.4
UI-R1-E-3B	97.1	83.0	95.4	77.9	91.7	85.0	89.2	89.5
GUI-R1-3B	-	-	93.8	64.8	89.6	72.1	-	-
GUI-R1-7B	-	-	91.8	73.6	91.3	75.7	-	-
InfGUI-R1-3B	97.1	81.2	94.3	77.1	91.7	77.6	87.5	-
GUI-G1-3B	98.6	85.8	96.4	80.7	91.4	82.3	90.3	-
SE-GUI-7B	-	-	-	-	-	-	88.2	90.3
LPO-8B	-	-	-	-	-	-	-	90.5
Ours								
GUI-G ² -7B	96.7	90.8	95.9	88.6	90.9	86.9	92.0	93.3

Table 1: Performance comparison on ScreenSpot v1 and v2. **Bold** highlights the best results, “-” indicates missing values due to unavailable results in the original paper, unreleased model checkpoints, and inference code.

UI RefExp (Bai et al., 2021), ShowUI-web (Lin et al., 2024), and OmniAct (Kapoor et al., 2024), covering diverse interface types across mobile, desktop, and web platforms. We evaluate on three benchmarks: ScreenSpot (Cheng et al., 2024) and ScreenSpot-v2 (Wu et al., 2024) for general GUI grounding, and ScreenSpot-Pro (Li et al., 2025) for high-resolution professional software interfaces. Following standard protocol (Cheng et al., 2024; Lin et al., 2024), predictions are considered correct when the predicted center falls within the ground truth bounding box.

Reward Type	Mobile		Desktop		Web		Avg
	Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
Sparse Reward							
Point	97.9	87.2	88.7	72.1	84.9	79.8	87.4
IoU	95.9	86.7	87.1	69.3	88.4	77.3	85.8
Point + IoU	97.2	86.7	88.1	68.6	88.9	78.8	86.5
Dense Reward							
GUI-G ² -7B	98.3	91.9	95.4	89.3	94.0	87.7	93.3

Table 2: Comparison of sparse and dense reward methods on ScreenSpot-v2.

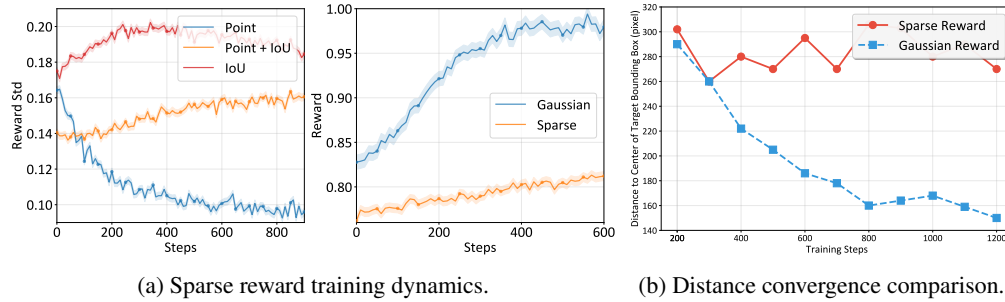


Figure 4: Reward comparison analysis. **Left:** Training dynamics of sparse reward variants (Point, IoU, Point+IoU) showing reward standard deviation and convergence patterns. **Right:** Distance to target center over training steps, where **Gaussian** rewards demonstrate monotonic convergence while **Sparse** rewards exhibit erratic fluctuations.

4.2 MAIN RESULTS

We evaluate GUI-G²-7B against existing methods across three benchmarks: ScreenSpot, ScreenSpot-v2, and ScreenSpot-Pro. Tables 1 and 3 show that our method achieves state-of-the-art performance among reinforcement learning approaches.

GUI-G²-7B reaches 92.0% on ScreenSpot, 93.3% on ScreenSpot-v2, and 47.5% on ScreenSpot-Pro, consistently outperforming all RL baselines. The most significant improvement occurs on ScreenSpot-Pro, where we surpass UI-TARS-72B by 9.4% (47.5% vs. 38.1%) while using 10 \times fewer parameters. This efficiency gain demonstrates that continuous Gaussian rewards enable smaller models to outperform much larger counterparts through more effective optimization.

Compared to other continuous reward methods, GUI-G² shows clear advantages. While LPO-8B and SE-GUI-7B also employ distance-based continuous rewards, they achieve only 90.5% and 90.3% respectively on ScreenSpot-v2, falling short of our 93.3%. This performance gap stems from a key insight: these methods treat GUI elements as point targets with distance decay, missing the planar nature of interface interactions. Our dual Gaussian formulation explicitly models both precise localization through point rewards and spatial extent through coverage rewards, capturing the complete interaction space that distance-only methods overlook.

The consistent improvements across diverse interface types validate the generalizability of our approach. On ScreenSpot-Pro’s high-resolution professional software, we achieve 64.7% on text elements compared to UI-TARS-72B’s 50.9%, indicating that Gaussian rewards particularly benefit tasks requiring fine spatial precision. These comprehensive improvements establish continuous Gaussian modeling as a principled foundation for GUI grounding, transforming sparse binary optimization into dense spatial learning that aligns with natural interaction patterns.

4.3 REWARD DESIGN ANALYSIS

Binary vs. Continuous Rewards. We investigate the fundamental differences between binary and continuous reward mechanisms by implementing three sparse baselines: Point rewards that activate when predicted centers fall within target boxes, IoU rewards that trigger when overlap exceeds 0.5, and their combination. To analyze convergence behavior, we select 10 challenging samples from ScreenSpot-v2 where initial predictions are incorrect, then track the average distance from predicted to ground truth centers across 8 sampled responses every 200 training steps.

Figure 4 exposes the critical limitations of sparse signals. Throughout training, binary rewards generate erratic optimization trajectories with severe fluctuations in both reward values and spatial convergence. The Point baseline achieves relative stability but plateaus early, while IoU rewards demonstrate particularly poor learning dynamics due to their restrictive activation threshold. Most strikingly, sparse methods show no consistent reduction in distance to target centers, oscillating wildly between 200-400 pixels without meaningful progress.

Model	CAD		Dev		Creative		Scientific		Office		OS		Avg.		Avg.
	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	
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)															
SeeClick-9.6B	2.5	0.0	0.6	0.0	1.0	0.0	3.5	0.0	1.1	0.0	2.8	0.0	1.8	0.0	1.1
FOCUS-2B	7.6	3.1	22.8	1.7	23.7	1.7	25.0	7.1	23.2	7.7	17.8	2.5	19.8	3.9	13.3
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
Aria-UI	7.6	1.6	16.2	0.0	23.7	2.1	27.1	6.4	20.3	1.9	4.7	0.0	17.1	2.0	11.3
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
UI-TARS-72B	18.8	12.5	62.9	17.2	57.1	15.4	64.6	20.9	63.3	26.4	42.1	15.7	50.9	17.6	38.1
JEDI-3B	27.4	9.4	61.0	13.8	53.5	8.4	54.2	18.2	64.4	32.1	38.3	9.0	49.8	13.7	36.1
JEDI-7B	38.0	14.1	42.9	11.0	50.0	11.9	72.9	25.5	75.1	47.2	33.6	16.9	52.6	18.2	39.5
GUI-Actor-7B	-	-	-	-	-	-	-	-	-	-	-	-	-	-	44.6
GUI-specific Models (RL)															
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
UI-R1-E-3B	37.1	12.5	46.1	6.9	41.9	4.2	56.9	21.8	65.0	26.4	32.7	10.1	-	-	33.5
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	-	-	-
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	-	-	-
InfGUI-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-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
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
SE-GUI-7B	51.3	42.2	68.2	19.3	57.6	9.1	75.0	28.2	78.5	43.4	49.5	25.8	63.5	21.0	47.3
Ours															
GUI-G ² -7B	55.8	12.5	68.8	17.2	57.1	15.4	77.1	24.5	74.0	32.7	57.9	21.3	64.7	19.6	47.5

Table 3: Performance comparison of different models across various task categories based on Text, Icon, and Average scores on ScreenSpot-Pro. "-" indicates unreported results in original papers.

Finding 1. Sparse rewards create unstable training dynamics with IoU rewards showing particularly poor learning efficiency due to restrictive thresholds.

In contrast, GUI-G² exhibits smooth monotonic convergence from 290px to 150px, demonstrating that continuous Gaussian signals fundamentally transform the optimization landscape. Table 2 quantifies this advantage: GUI-G² achieves 93.3% accuracy, surpassing the best sparse baseline (Point: 87.4%) by 5.9%. This substantial gap emerges because Gaussian rewards provide informative gradients at every spatial position, enabling models to learn from predictions at any distance from targets. Binary rewards create a discrete cliff at bounding box edges where gradient information vanishes, leaving models without guidance for improving near-miss predictions. Our continuous formulation eliminates these optimization barriers, creating smooth paths toward target elements from any starting position.

Finding 2. Continuous Gaussian rewards enable monotonic convergence and achieve +5.9% performance improvement through dense spatial feedback signals.

Inside vs. Outside Boundary Rewards: Why Continuous Everywhere Matters. A natural question arises: should rewards be provided only within target boundaries or everywhere in the interface? We implement an Inside Gaussian (IG) baseline that applies our Gaussian formulation only when predictions fall within ground truth boxes, reverting to zero otherwise. Figure 6 shows

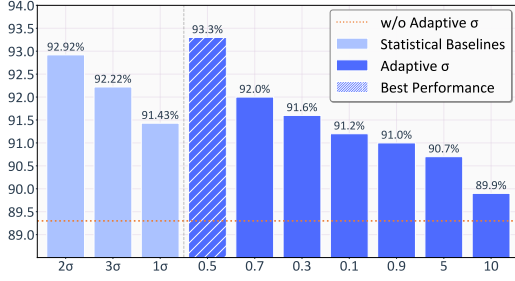


Figure 5: Hyperparameter sensitivity analysis for adaptive sigma (σ). Performance peaks at $\alpha = 0.5$ with 93.3% accuracy on Screenspot-v2.

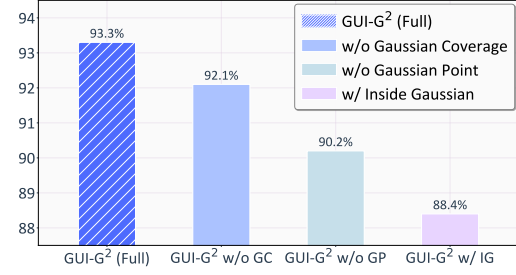


Figure 6: Ablation of Gaussian Component. Both Point and Coverage components contribute to the final 93.3% performance.

Model	ScreenSpot Accuracy (%)				ScreenSpot-v2 Accuracy (%)			
	Mobile	Desktop	Web	Avg.	Mobile	Desktop	Web	Avg.
SE-GUI-7B	85.6	91.4	86.5	88.2	95.2	87.1	87.0	90.3
GUI-G ² -7B	94.0	92.8	89.0	92.0	95.6	92.8	91.1	93.3

Table 4: **Gaussian vs. Distance-based Dense Rewards.** Our GUI-G² outperforms SE-GUI-7B on both ScreenSpot and ScreenSpot-v2 datasets, demonstrating the effectiveness of Gaussian-based dense rewards over distance-based dense reward mechanisms.

GUI-G² outperforms IG by 4.9% (93.3% vs. 88.4%), despite using identical Gaussian formulations. This reveals that restricting rewards to successful predictions, even with continuous formulations, recreates the fundamental problem of sparse signals. By providing Gaussian feedback throughout the entire interface plane, GUI-G² enables models to learn from every prediction, creating smooth optimization paths from any starting position to target elements.

Finding 3. Providing continuous Gaussian rewards everywhere in the spatial domain, rather than only within target boundaries, improves performance by 4.9% through eliminating optimization discontinuities.

Dual Gaussian Components: Point Precision and Spatial Coverage. GUI-G²’s dual formulation addresses complementary aspects of interface interaction. Ablation studies in Figure 6 demonstrate that removing either component significantly degrades performance: 92.1% without coverage rewards and 90.2% without point rewards, compared to 93.3% with both. Point rewards alone optimize for precise center localization but ignore that users can successfully click anywhere within element boundaries. Coverage rewards alone measure spatial overlap but lack the precision to guide models toward optimal clicking positions. The 1.2% improvement from their combination confirms that effective GUI grounding requires modeling both aspects simultaneously, reflecting how humans naturally aim for element centers while accepting clicks anywhere within boundaries.

GUI-G² vs. Distance-Based Rewards. SE-GUI-7B represents an alternative continuous approach using normalized Euclidean distance. Table 4 shows GUI-G² consistently outperforms SE-GUI by 3.8% on ScreenSpot and 3.0% on ScreenSpot-v2. This gap highlights two fundamental differences. First, SE-GUI treats elements as point targets, computing distances to centers without considering spatial extent. Second, it applies different formulas inside versus outside bounding boxes, creating gradient discontinuities at boundaries. Our unified Gaussian formulation provides smooth gradients everywhere while explicitly modeling both localization precision and spatial coverage, better capturing the continuous nature of GUI interactions.

4.4 ABLATION STUDIES

Adaptive Variance Mechanism. GUI elements span diverse scales from tiny icons to full-screen panels, and human clicking tolerance naturally varies with element dimensions: larger elements

Configuration	ν	γ	Acc (%)
GUI-G ²	1.0	1.0	93.3
GUI-G ² [Format]	1.0	1.0	93.2
GUI-G ² [GP]	0.8	0.2	92.2
GUI-G ² [GC]	0.2	0.8	91.8

Table 5: Reward weighting configurations.

Configuration	Accuracy (%)	Tokens
Thinking	88.7	130
No Thinking	93.3	16
Δ	+4.6	-114
Relative	+5.2%	-87.7%

Table 6: Thinking vs. No Thinking Analysis.

accommodate greater spatial uncertainty, while small icons require precise targeting. To handle this diversity, we propose an adaptive variance mechanism that scales reward distributions based on element size and validate it against multiple baselines. We implement the following configurations: (i) 1σ Principle: $\sigma_x = \text{width}/2$, $\sigma_y = \text{height}/2$; (ii) 2σ Principle: $\sigma_x = \text{width} \times 2$, $\sigma_y = \text{height} \times 2$; (iii) 3σ Principle: $\sigma_x = \text{width} \times 3$, $\sigma_y = \text{height} \times 3$; and (iv) w/o adaptive σ : using fixed variance for all elements. As shown in Table 5, our adaptive mechanism with $\alpha = 0.5$ achieves peak performance at 93.3%, substantially outperforming fixed variance approaches (87.8%) by +5.5 percentage points. Remarkably, this optimal value aligns with the 2σ statistical principle (92.92%), demonstrating that effective GUI grounding emerges from balanced spatial tolerance that neither over-constrains nor under-constrains interaction boundaries. The 1σ principle (91.43%) proves overly restrictive by failing to capture natural clicking variability, while the 3σ principle (92.22%) shows that excessive tolerance dilutes localization precision. Our adaptive mechanism’s superiority over even the optimal 2σ baseline reveals that personalized calibration based on individual element characteristics provides the optimal balance between spatial flexibility and targeting precision.

Balancing Point and Coverage Rewards. To evaluate the impact of different weighting schemes between Gaussian point and Gaussian coverage rewards, we perform ablation experiments with the following configurations: (i) GUI-G²: our original model with $R = 1.0 \times R_{\text{point}} + 1.0 \times R_{\text{coverage}}$; (ii) GUI-G² [GP]: Point-dominant weighting with $R = 0.8 \times R_{\text{point}} + 0.2 \times R_{\text{coverage}}$; (iii) GUI-G² [GC]: Coverage-dominant weighting with $R = 0.2 \times R_{\text{point}} + 0.8 \times R_{\text{coverage}}$; (iv) GUI-G² [Format]: our original model with additional format reward that assigns reward 1 when the model outputs exactly four numerical coordinates in the required format $[x1, y1, x2, y2]$ and 0 otherwise. As shown in Table 5, equal weighting (1.0 each) achieves optimal performance at 93.3%, outperforming both point-dominant (92.2%) and coverage-dominant (91.8%) configurations. This demonstrates that effective GUI grounding requires simultaneous optimization of precise localization and spatial overlap modeling with balanced importance.

Finding 4. Balanced weighting of point and coverage rewards (1.0 each) achieves optimal performance, while format rewards provide minimal benefit.

Thinking vs. No Thinking Grounding. Most previous methods adopt the R1-style reasoning paradigm directly (Luo et al., 2025; Liu et al., 2025d), following the success of reasoning-based models in other domains. However, this widespread adoption raises a fundamental question: is explicit reasoning truly beneficial for GUI grounding tasks? To investigate whether GUI grounding is suitable for thinking-based optimization, we conduct controlled experiments comparing thinking versus non-thinking approaches. Both configurations utilize GUI-G² reward mechanism, with the thinking model additionally receiving format rewards for proper usage of thinking tag. Training prompts detailed in Appendix A.2. As shown Table 6, our experiments demonstrate that **explicit reasoning significantly impairs GUI grounding performance**. The non-thinking approach achieves 93.3% on ScreenSpot-v2, substantially outperforming the thinking approach at 88.7%—a +5.3% improvement while using 76.9% fewer tokens. This counterintuitive finding suggests that GUI grounding is fundamentally a perceptual task relying on immediate visual pattern recognition rather than step-by-step analysis. The performance degradation likely occurs because reasoning tokens compete with visual representations for attention, interfering with crucial visual features essential for accurate element localization.

Finding 5. Explicit reasoning processes significantly harm GUI grounding performance.

5 CONCLUSION

In this work, we propose GUI-G², a principled reward modeling framework that reconceptualizes GUI grounding as a continuous spatial optimization task. Unlike traditional reinforcement learning approaches that rely on sparse binary rewards, GUI-G² leverages Gaussian point rewards and Gaussian coverage rewards to provide dense, geometrically-aware feedback signals. By modeling GUI elements as 2D Gaussian distributions and introducing an adaptive variance mechanism, our method captures both fine-grained localization precision and spatial coverage characteristics, which enables more efficient learning and better generalization. Evaluated on three benchmarks—ScreenSpot, ScreenSpot-v2, and ScreenSpot-Pro. GUI-G²-7B outperforms state-of-the-art models, achieving up to 24.7% improvement over UI-TARS-72B on high-resolution professional interfaces. These results establish GUI-G² as a robust and effective solution for spatial reasoning in GUI interaction tasks.

REFERENCES

- I. Scott MacKenzie and. Fitts’ law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7(1):91–139, 1992. doi: 10.1207/s15327051hci0701_3. URL https://doi.org/10.1207/s15327051hci0701_3.
- Anthropic. Claude computer use. Available at: <https://www.anthropic.com/news/developing-computer-use>, 2024.
- Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, and Blaise Agüera y Arcas. Uibert: Learning generic multimodal representations for ui understanding, 2021. URL <https://arxiv.org/abs/2107.13731>.
- Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL <https://arxiv.org/abs/2502.13923>.
- Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. SeeClick: Harnessing gui grounding for advanced visual gui agents, 2024. URL <https://arxiv.org/abs/2401.10935>.
- Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. Sft memorizes, rl generalizes: A comparative study of foundation model post-training, 2025. URL <https://arxiv.org/abs/2501.17161>.
- Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning, 2023. URL <https://arxiv.org/abs/2307.08691>.
- DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL <https://arxiv.org/abs/2501.12948>.
- Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and Haoshuang Wang. Pp-ocr: A practical ultra lightweight ocr system, 2020. URL <https://arxiv.org/abs/2009.09941>.
- Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Junfei Wu, Xiaoying Zhang, Benyou Wang, and Xiangyu Yue. Video-r1: Reinforcing video reasoning in mllms, 2025. URL <https://arxiv.org/abs/2503.21776>.
- P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 74:381–391, 1954.
- Boyuan Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for gui agents, 2024. URL <https://arxiv.org/abs/2410.05243>.

- Wenyi Hong, Weihang Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents, 2024. URL <https://arxiv.org/abs/2312.08914>.
- Wenjia Jiang, Yangyang Zhuang, Chenxi Song, Xu Yang, Joey Tianyi Zhou, and Chi Zhang. Appagentx: Evolving gui agents as proficient smartphone users. 2025. URL <https://arxiv.org/abs/2503.02268>.
- Raghav Kapoor, Yash Parag Butala, Melisa Russak, Jing Yu Koh, Kiran Kamble, Waseem Alshikh, and Ruslan Salakhutdinov. Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web. 2024. URL <https://arxiv.org/abs/2402.17553>.
- Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023. URL <https://arxiv.org/abs/2304.02643>.
- Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. Screenspot-pro: Gui grounding for professional high-resolution computer use, 2025.
- Yanda Li, Chi Zhang, Wanqi Yang, Bin Fu, Pei Cheng, Xin Chen, Ling Chen, and Yunchao Wei. Appagent v2: Advanced agent for flexible mobile interactions, 2024. URL <https://arxiv.org/abs/2408.11824>.
- Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Weixian Lei, Lijuan Wang, and Mike Zheng Shou. Showui: One vision-language-action model for gui visual agent, 2024. URL <https://arxiv.org/abs/2411.17465>.
- Xiao Liu, Bo Qin, Dongzhu Liang, Guang Dong, Hanyu Lai, Hanchen Zhang, Hanlin Zhao, Iat Long Iong, Jiadai Sun, Jiaqi Wang, Junjie Gao, Junjun Shan, Kangning Liu, Shudan Zhang, Shuntian Yao, Siyi Cheng, Wentao Yao, Wenyi Zhao, Xinghan Liu, Xinyi Liu, Xinying Chen, Xinyue Yang, Yang Yang, Yifan Xu, Yu Yang, Yujia Wang, Yulin Xu, Zehan Qi, Yuxiao Dong, and Jie Tang. Autoglm: Autonomous foundation agents for guis. 2024. URL <https://arxiv.org/abs/2411.00820>.
- Xuyang Liu, Yiyu Wang, Junpeng Ma, and Linfeng Zhang. Video compression commander: Plug-and-play inference acceleration for video large language models, 2025a. URL <https://arxiv.org/abs/2505.14454>.
- Xuyang Liu, Ziming Wang, Yuhang Han, Yingyao Wang, Jiale Yuan, Jun Song, Bo Zheng, Linfeng Zhang, Siteng Huang, and Honggang Chen. Global compression commander: Plug-and-play inference acceleration for high-resolution large vision-language models, 2025b. URL <https://arxiv.org/abs/2501.05179>.
- Xuyang Liu, Zichen Wen, Shaobo Wang, Junjie Chen, Zhishan Tao, Yubo Wang, Xiangqi Jin, Chang Zou, Yiyu Wang, Chenfei Liao, Xu Zheng, Honggang Chen, Weijia Li, Xuming Hu, Conghui He, and Linfeng Zhang. Shifting ai efficiency from model-centric to data-centric compression, 2025c. URL <https://arxiv.org/abs/2505.19147>.
- Yuhang Liu, Pengxiang Li, Congkai Xie, Xavier Hu, Xiaotian Han, Shengyu Zhang, Hongxia Yang, and Fei Wu. Infigui-r1: Advancing multimodal gui agents from reactive actors to deliberative reasoners. 2025d. URL <https://arxiv.org/abs/2504.14239>.
- Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based gui agent, 2024. URL <https://arxiv.org/abs/2408.00203>.
- Zhengxi Lu, Yuxiang Chai, Yaxuan Guo, Xi Yin, Liang Liu, Hao Wang, Han Xiao, Shuai Ren, Guanqing Xiong, and Hongsheng Li. Ui-r1: Enhancing efficient action prediction of gui agents by reinforcement learning. 2025. URL <https://arxiv.org/abs/2503.21620>.
- Run Luo, Lu Wang, Wanwei He, and Xiaobo Xia. Gui-r1 : A generalist r1-style vision-language action model for gui agents. 2025. URL <https://arxiv.org/abs/2504.10458>.

OpenAI. Introducing gpt-4o. Available at: <https://openai.com/index/hello-gpt-4o>, 2024.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjuan Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, Xiaojun Xiao, Kai Cai, Chuang Li, Yaowei Zheng, Chaolin Jin, Chen Li, Xiao Zhou, Minchao Wang, Haoli Chen, Zhaojian Li, Haihua Yang, Haifeng Liu, Feng Lin, Tao Peng, Xin Liu, and Guang Shi. Ui-tars: Pioneering automated gui interaction with native agents, 2025. URL <https://arxiv.org/abs/2501.12326>.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. Android in the wild: A large-scale dataset for android device control. 2023. URL <https://arxiv.org/abs/2307.10088>.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. Spurious rewards: Rethinking training signals in rlvr, 2025. URL <https://arxiv.org/abs/2506.10947>.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL <https://arxiv.org/abs/2402.03300>.

Haozhan Shen, Peng Liu, Jingcheng Li, Chunxin Fang, Yibo Ma, Jiajia Liao, Qiaoli Shen, Zilun Zhang, Kangjia Zhao, Qianqian Zhang, Ruochen Xu, and Tiancheng Zhao. Vlm-r1: A stable and generalizable r1-style large vision-language model, 2025. URL <https://arxiv.org/abs/2504.07615>.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face, 2023. URL <https://arxiv.org/abs/2303.17580>.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, Ben Kao, Guohao Li, Junxian He, Yu Qiao, and Zhiyong Wu. Os-genesis: Automating gui agent trajectory construction via reverse task synthesis, 2025. URL <https://arxiv.org/abs/2412.19723>.

Fei Tang, Yongliang Shen, Hang Zhang, Siqi Chen, Guiyang Hou, Wenqi Zhang, Wenqiao Zhang, Kaitao Song, Weiming Lu, and Yueting Zhuang. Think twice, click once: Enhancing gui grounding via fast and slow systems. 2025a. URL <https://arxiv.org/abs/2503.06470>.

Fei Tang, Haolei Xu, Hang Zhang, Siqi Chen, Xingyu Wu, Yongliang Shen, Wenqi Zhang, Guiyang Hou, Zeqi Tan, Yuchen Yan, Kaitao Song, Jian Shao, Weiming Lu, Jun Xiao, and Yueting Zhuang. A survey on (m)llm-based gui agents. 2025b. URL <https://arxiv.org/abs/2504.13865>.

Jiaqi Tang, Yu Xia, Yi-Feng Wu, Yuwei Hu, Yuhui Chen, Qing-Guo Chen, Xiaogang Xu, Xiangyu Wu, Hao Lu, Yanqing Ma, Shiyin Lu, and Qifeng Chen. Lpo: Towards accurate gui agent interaction via location preference optimization, 2025c. URL <https://arxiv.org/abs/2506.09373>.

Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration, 2024a. URL <https://arxiv.org/abs/2406.01014>.

Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception, 2024b. URL <https://arxiv.org/abs/2401.16158>.

Zhenhailong Wang, Haiyang Xu, Junyang Wang, Xi Zhang, Ming Yan, Ji Zhang, Fei Huang, and Heng Ji. Mobile-agent-e: Self-evolving mobile assistant for complex tasks, 2025. URL <https://arxiv.org/abs/2501.11733>.

- Qianhui Wu, Kanzhi Cheng, Rui Yang, Chaoyun Zhang, Jianwei Yang, Huiqiang Jiang, Jian Mu, Baolin Peng, Bo Qiao, Reuben Tan, et al. Gui-actor: Coordinate-free visual grounding for gui agents. *arXiv preprint arXiv:2506.03143*, 2025.
- Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. Os-atlas: A foundation action model for generalist gui agents, 2024. URL <https://arxiv.org/abs/2410.23218>.
- Tianbao Xie, Jiaqi Deng, Xiaochuan Li, Junlin Yang, Haoyuan Wu, Jixuan Chen, Wenjing Hu, Xinyuan Wang, Yuhui Xu, Zekun Wang, Yiheng Xu, Junli Wang, Doyen Sahoo, Tao Yu, and Caiming Xiong. Scaling computer-use grounding via user interface decomposition and synthesis, 2025. URL <https://arxiv.org/abs/2505.13227>.
- Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. Aria-ui: Visual grounding for gui instructions, 2024. URL <https://arxiv.org/abs/2412.16256>.
- Xinbin Yuan, Jian Zhang, Kaixin Li, Zhuoxuan Cai, Lujian Yao, Jie Chen, Enguang Wang, Qibin Hou, Jinwei Chen, Peng-Tao Jiang, and Bo Li. Enhancing visual grounding for gui agents via self-evolutionary reinforcement learning. 2025. URL <https://arxiv.org/abs/2505.12370>.
- Chaoyun Zhang, Liqun Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. Ufo: A ui-focused agent for windows os interaction, 2024. URL <https://arxiv.org/abs/2402.07939>.
- Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue Liu, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. Large language model-brained gui agents: A survey. 2025a. URL <https://arxiv.org/abs/2411.18279>.
- Chaoyun Zhang, He Huang, Chiming Ni, Jian Mu, Si Qin, Shilin He, Lu Wang, Fangkai Yang, Pu Zhao, Chao Du, Liqun Li, Yu Kang, Zhao Jiang, Suzhen Zheng, Rujia Wang, Jiaxu Qian, Minghua Ma, Jian-Guang Lou, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Ufo2: The desktop agentos. 2025b. URL <https://arxiv.org/abs/2504.14603>.
- Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. Appagent: Multimodal agents as smartphone users, 2023. URL <https://arxiv.org/abs/2312.13771>.
- Yuqi Zhou, Sunhao Dai, Shuai Wang, Kaiwen Zhou, Qinglin Jia, and Jun Xu. Gui-gl: Understanding rl-zero-like training for visual grounding in gui agents. 2025. URL <https://arxiv.org/abs/2505.15810>.

A APPENDIX

A.1 ANALYSIS OF SPURIOUS REWARDS

Recent studies have shown that even spurious rewards can stimulate reinforcement learning training processes (Shao et al., 2025), raising important questions about reward design robustness. To better explore the impact of artificial reward signals on GUI grounding performance and validate the necessity of our proposed Point-to-Plane Gaussian reward mechanism, we conducted controlled experiments with two distinct fake reward strategies: (i) Random $U(0, 1)$ Reward: rewards are randomly sampled from a uniform distribution $U(0, 1)$ (including boundary values 0 and 1); (ii) Binary Random Reward: rewards are randomly assigned as either 0 or 1 with equal probability, creating maximum variance in sparse feedback patterns.

As shown in Figure 8 and 9, our experimental results reveal three critical key findings: **(1) GUI Grounding Cannot Benefit Spurious Rewards for Effective Learning:** Both random reward strategies exhibit progressive performance degradation with consistent downward trends.

Continuous random rewards decline from 90.6% to 87.9% (-2.7%) and binary random rewards drop from 88.6% to 84.5% (-4.1%) over 3000 steps. These spurious rewards fail to provide effective learning signals for GUI grounding tasks, leading to gradual performance deterioration. Unlike other domains where random rewards may provide training benefits, GUI grounding tasks cannot benefit from arbitrary reward signals, demonstrating that meaningful spatial feedback is essential for effective learning. The failure of spurious rewards validates the effectiveness of our Point-to-Plane Gaussian reward mechanism. **(2) Continuous Random Rewards Show Superior Initial Performance:** The $U(0, 1)$ strategy maintains higher initial accuracy (90.6% vs 88.6%) with more gradual degradation. This difference stems from the fundamental learning signal availability: continuous random rewards consistently provide non-zero feedback at every training step, ensuring gradient flow and parameter updates throughout the learning process. In contrast, the binary strategy introduces complete signal absence (zero rewards) with 50% probability, creating intermittent learning interruptions. During early training phases, these zero rewards introduce excessive noise that immediately disrupts gradient estimation and blocks policy updates, causing faster knowledge degradation. The continuous feedback mechanism, despite being random, maintains smoother gradient dynamics compared to the sporadic learning signals in binary rewards, highlighting the critical importance of consistent reward availability in reinforcement learning systems.

Hyperparameter	Value
num_generations	8
per_device_train_batch_size	8
gradient_accumulation_steps	1
bf16	true
torch_dtype	bfloat16
data_seed	42
gradient_checkpointing	true
attn_implementation	flash_attention_2
num_train_epochs	1
max_pixels	12845056
β	0.04
α	0.5
ν	1.0
γ	1.0

Figure 7: Hyperparameter settings used in the training experiments.

A.2 EVALUATION DETAILS

Compared Methods. To better evaluate the advantages of our P2G reward mechanism, we assess existing methods that employ the RL paradigm for training as follows:

- **UI-R1** (Lu et al., 2025): Employs traditional sparse point rewards, assigning a reward of 1 when predicted coordinates [x,y] fall within the ground truth bounding box, and 0 otherwise.
- **GUI-R1** (Luo et al., 2025): Adopts the same sparse point reward strategy as UI-R1, specifically designed as a binary reward mechanism for GUI Grounding tasks.
- **GUI-G1** (Zhou et al., 2025): Combines sparse point rewards with IoU rewards for joint optimization, and introduces an adaptive reward function based on predicted bounding box size to handle GUI elements of different scales.

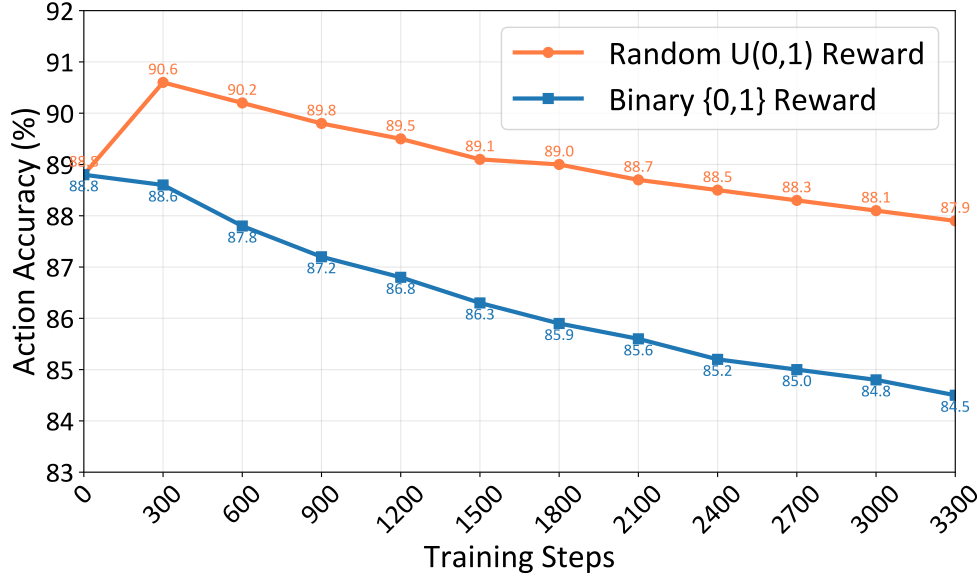


Figure 8: Performance comparison between random reward strategies on ScreenSpot-V2. Both continuous random $U(0, 1)$ rewards and binary random rewards show progressive degradation, demonstrating that GUI grounding requires spatially-meaningful reward signals rather than arbitrary feedback.

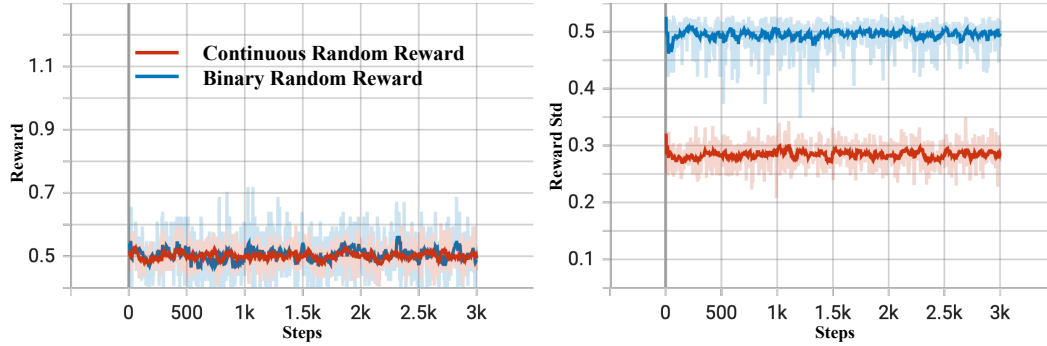


Figure 9: Reward distribution and standard deviation analysis during training. The reward variance patterns illustrate the fundamental differences between continuous and binary random reward mechanisms in reinforcement learning dynamics.

- **InfGUI-R1** (Liu et al., 2025d): Simultaneously utilizes sparse point rewards and sparse IoU rewards for optimization, enhancing GUI element localization accuracy through dual sparse reward mechanisms.
- **SE-GUI** (Yuan et al., 2025): Adopts a continuous reward function based on normalized distance, providing different reward values according to whether the predicted point is within the target bounding box and its distance from the center point.
- **LPO** (Tang et al., 2025c): Implements a dynamic location reward mechanism that provides continuous reward feedback based on spatial accuracy by calculating the Euclidean distance between executed coordinates and target coordinates.

This comprehensive comparison encompasses diverse reward paradigms ranging from sparse binary mechanisms to continuous distance-based formulations, enabling thorough validation of our proposed GUI-G² approach across different methodological frameworks.

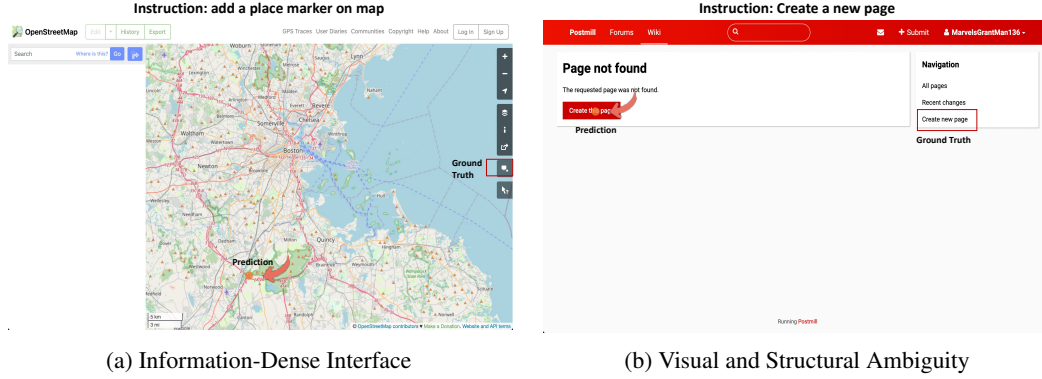


Figure 10: Analysis of GUI grounding failure cases.

Thinking Prompt

{problem} Output the thinking process in `<think>` `</think>` and final answer in `<answer>` `[x1,y1,x2,y2]` `</answer>` tags.

No Thinking Prompt

Outline the position corresponding to the instruction: {problem}. The output should be only `[x1,y1,x2,y2]`.

A.3 ERROR ANALYSIS

We analyzed failure cases from the ScreenSpot-V2 dataset and found that icon recognition remains a critical challenge in GUI grounding. Our analysis of error distribution reveals that icon errors account for 76.9% of total failures (63 icon errors vs 19 text errors across all platforms). This stark disparity underscores that semantic interpretation of visual symbols represents a fundamental bottleneck, as icons require models to infer abstract functionality from visual representations rather than explicit textual information. Beyond icon challenges, we identified two primary failure patterns: information-dense interface bottlenecks, where environments with high information density such as online maps or complex software interfaces (Figure 10a) overwhelm the model’s processing capabilities due to overlapping elements and complex visual hierarchies; and visual and structural ambiguity, where the model becomes confused when multiple UI elements share similar visual features or spatial arrangements (Figure 10b), leading to incorrect selections among viable candidates when task descriptions lack sufficient specificity. These findings highlight that current GUI grounding limitations stem primarily from semantic understanding challenges rather than spatial localization capabilities, suggesting that future research should prioritize enhanced visual-semantic reasoning to bridge the gap between visual perception and functional intent.

A.4 FUTURE WORK

As GUI agents scale to handle complex high-resolution interfaces, computational overhead may become a limiting factor for practical deployment. Future work could explore model compression techniques (Liu et al., 2025c) and acceleration frameworks for large vision-language models (Liu et al., 2025a;b) to reduce inference costs while preserving grounding performance. Such optimizations would enable broader adoption of GUI agents across diverse computing environments.