BAMI: Training-Free Bias Mitigation in GUI Grounding

Borui Zhang[1*] Bo Zhang[1*] Bo Wang[1*] Wenzhao Zheng[1] Yuhao Cheng[2]
Liang Tang[2] Yiqiang Yan[2] Jie Zhou[1] Jiwen Lu[1,†]

[1]Tsinghua University, China [2]Lenovo Research, China

Abstract

*GUI grounding is a critical capability for enabling GUI agents to execute tasks such as clicking and dragging. However, in complex scenarios like the ScreenSpot-Pro benchmark, existing models often suffer from suboptimal performance. Utilizing the proposed **Masked Prediction Distribution (MPD)** attribution method, we identify that the primary sources of errors are twofold: high image resolution (leading to precision bias) and intricate interface elements (resulting in ambiguity bias). To address these challenges, we introduce **Bias-Aware Manipulation Inference (BAMI)**, which incorporates two key manipulations, coarse-to-fine focus and candidate selection, to effectively mitigate these biases. Our extensive experimental results demonstrate that BAMI significantly enhances the accuracy of various GUI grounding models in a training-free setting. For instance, applying our method to the TianXi-Action-7B model boosts its accuracy on the ScreenSpot-Pro benchmark from 51.9% to 57.8%. Furthermore, ablation studies confirm the robustness of the BAMI approach across diverse parameter configurations, highlighting its stability and effectiveness.* [1]

1. Introduction

The advent of multimodal large language models (MLLMs) [2, 11] has made it increasingly feasible for GUI agents to automate tasks across desktop and mobile platforms. At the core of these agents lies *GUI Grounding*: given a pair of *natural language instructions* and a *screenshot*, the task is to accurately localize the coordinates of the target element within a high-resolution graphical interface, thereby enabling subsequent atomic actions such as clicking, typing, or dragging. Early approaches often relied on structured interface representations, such as XML or DOM trees [4, 8]. However, these structures are frequently unavailable or inconsistent with the visual rendering in real-world scenarios. Consequently, research has shifted toward the visual paradigm of *instruction + screenshot*, where MLLMs directly output coordinates [6, 16, 21, 30, 31], providing a more robust perceptual foundation for agents. In comparison to general natural image tasks, GUI scenarios present unique challenges due to their **high resolution** and **dense elements**, where semantics are determined by a combination of icons, text, and contextual cues. These characteristics make accurate localization significantly more challenging. For instance, in ScreenSpot-Pro [12], a benchmark dataset covering professional software across multiple domains, the localization accuracy of most models remains below 50%.

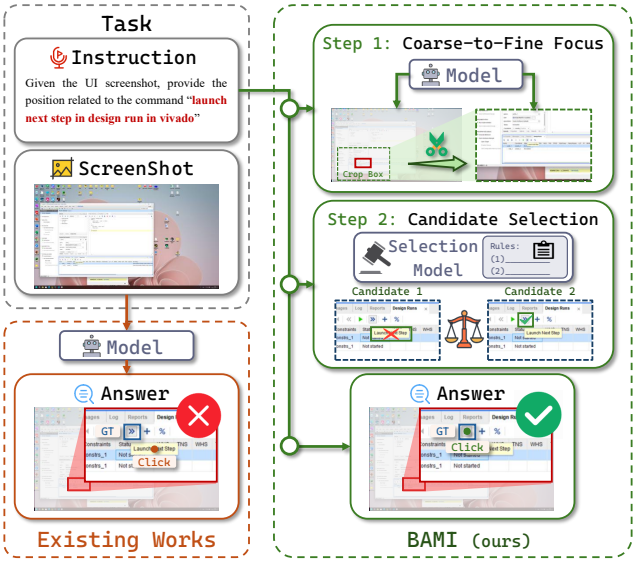


Figure 1. Compared with conventional grounding models, **BAMI** achieves accurate localization without additional training via structured inference with bias-aware manipulations.

The performance of multimodal grounding models remains underutilized. In particular, performance improvements can be achieved **without additional training** by optimizing inference methods. From an error-driven perspective, we categorize grounding failures into two primary types: **(1)**

*Equal contribution, order decided by coin flip.
†Corresponding author: Jiwen Lu (lujiwen@tsinghua.edu.cn).
[1]Code is available at https://github.com/Neur-IO/BAMI.

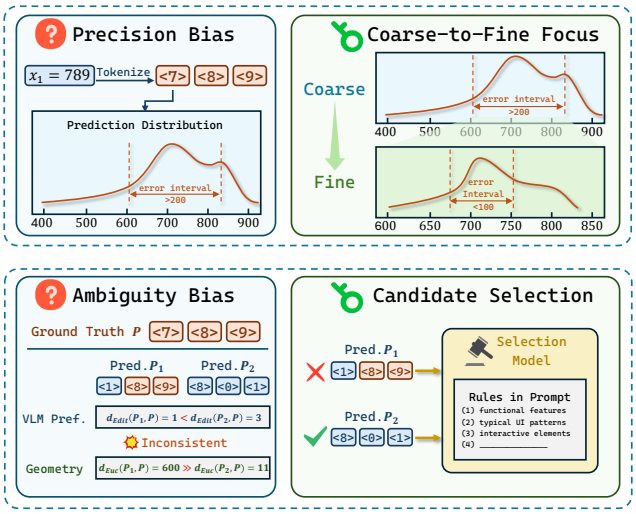


Figure 2. **Bias Mitigation Strategy.** To address accuracy bias and ambiguity bias, BAMI introduces two manipulations: coarse-to-fine focus and candidate selection.

Knowledge deficiency: The model fails to recognize the target due to a lack of relevant knowledge. **(2) Inductive bias**: The model has the necessary knowledge but makes errors due to its inherent selection bias, which manifests in two typical forms, namely *precision bias* and *ambiguity bias*. To diagnose these causes of failure, we introduce a **Masked Prediction Distribution (MPD)** method. This approach randomly occludes parts of the screenshot, makes repeated predictions, and aggregates the frequency of hotspots or candidate points across the image. This aggregation reveals how the model distributes its focus across the image. Statistical analysis of 50 error samples shows that approximately 14% of failures stem from knowledge deficiency, while 74% are attributed to inductive bias.

In this paper, we propose **Bias-Aware Manipulation Inference (BAMI)**. The key idea is to transform the one-step localization task into a recursive, multi-step structured inference process through predefined bias-aware manipulations (Figure 1 and 2). To mitigate precision bias, we decompose localization into hierarchical **coarse-to-fine focus**, where each step refines the candidate region identified in the previous round. This progressive refinement reduces the search space and improves the resolution of the predicted coordinates. To address ambiguity bias, we incorporate an external **Candidate Selection**. By defining selection rules specific to the localization task and injecting these rules into the model as prompts, we correct the model's erroneous selection preferences. Importantly, our method does not require any additional model training and can be directly applied to a variety of existing open-source backbones. We evaluate BAMI on multiple open-source backbones (e.g., OS-Atlas-7B [30], UI-TARS-7B [21], and TianXi-Action-7B [26]) and several datasets (e.g., ScreenSpot-Pro [12], ScreenSpot-V2 [30]). BAMI consistently improves accuracy on complex samples (Figure 3). Ablation studies further confirm the effects of **coarse-to-fine focus** and **candidate selection**. Our results demonstrate that extending and structuring the reasoning path during inference provides a cost-effective means of unlocking the full grounding potential of existing models. The main contributions of this work are as follows:

- **Diagnosis of Grounding Failures**: We introduce the MPD method to diagnose common grounding failures, such as knowledge deficiency and inductive bias.
- **Precision Bias Mitigation**: We transform single-step localization into a multi-step progressive search through hierarchical cropping, which effectively reduces precision bias in high-resolution and small-object scenarios.
- **Ambiguity Bias Correction**: To address discrepancies between MLLM's edit distance and spatial coordinate distance, we introduce an external selection and correct the MLLM's selection bias using predefined prompt rules.
- **Training-free Improvements**: We validate BAMI across various backbones and benchmarks, demonstrating consistent improvements and emphasizing the general value of test-time reasoning design in GUI Grounding.

2. Related Work

Training on pre-trained MLLMs [2] has been demonstrated to significantly enhance GUI grounding capabilities. Early approaches predominantly relied on conventional instruction fine-tuning. With the introduction of DeepSeek-R1 [7], reinforcement learning fine-tuning has attracted growing attention. Meanwhile, several studies have found that specially designed inference methods help tap into the potential of MLLMs in terms of localization capabilities.

2.1. Instruction Fine-tuning

The simplest approach is to fine-tune pre-trained MLLMs (e.g., Qwen2.5-VL [2]) on task-specific GUI instruction datasets. Early work such as AGUVIS [31] introduced vision-based models for GUI grounding. To address high-resolution GUI screenshots, CogAgent [9] introduced a cross-resolution efficient attention mechanism. ShowUI [13] applied token pruning based on GUI interface structure, improving both efficiency and performance. OmniParser [16] converted GUI pixels into structured tokens that could be parsed by LLMs. In terms of dataset construction, SeeClick [3] proposed an automated pipeline for managing GUI data. UGround [6] built a large-scale dataset with 10M elements, improving generalization. With the advent of larger-scale datasets and more powerful models, new large-scale systems such as UI-TARS [21] and Phi-Ground [34] have pushed the SOTA performance across benchmarks.

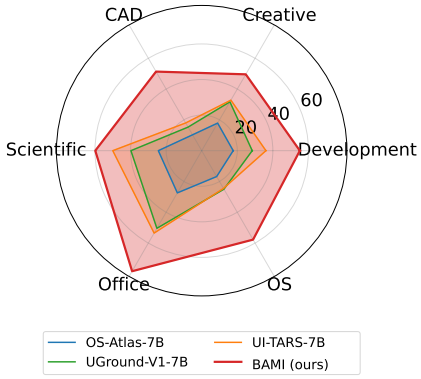


Figure 3. **Accuracy comparison on ScreenSpot-Pro.** BAMI consistently improves performance across all model backbones.

2.2. Reinforcement Learning

Given the fine-grained nature of GUI localization, instruction fine-tuning alone is often insufficient for achieving high precision. DeepSeek-R1 [7] introduced the GRPO method, demonstrating the potential of reinforcement learning in enhancing spatial reasoning for GUI grounding tasks. Following this, UI-R1 [17] and GUI-R1 [19] were among the first to apply GRPO in GUI tasks. InfiGUI-R1 [15] focused on reward function design, emphasizing IoU-based metrics to improve localization accuracy. GUI-G1 [35] introduced box-attribute constraints to regulate bounding-box geometry, while GUI-G2 [25] modeled spatial distributions using Gaussian functions. TianXi-Action [26] focused on generating high-quality reinforcement learning data. Collectively, these studies affirm the efficacy of reinforcement learning in enhancing spatial reasoning in GUI tasks.

2.3. Inference Enhancement

Significant attention has been given to optimizing inference strategies to exploit the capabilities of MLLMs. One line of work extends reasoning chains in the language space; however, experiments [33] have found this direction suboptimal for GUI scenarios, sometimes even hindering performance. Alternatively, several works have targeted inference enhancement in the image space. ScreenSeekeR [12] and R-VLM [20] introduced multi-stage pipelines, first performing region-level localization followed by refinement within local regions, thus improving accuracy. DiMo-GUI [29] proposed a divide-and-conquer strategy, separating reasoning over icons and text to reduce cross-modal interference. GUI-RC [5] employed intersection operations to aggregate multiple predictions, improving robustness. While conventional MLLMs have demonstrated the effectiveness of inference enhancement techniques for general tasks [14], their direct application to GUI tasks is often limited by inductive biases specific to spatial reasoning. This paper identifies two critical inductive biases —**precision bias and ambiguity bias**— that remain prominent in GUI grounding. We propose BAMI to address these through bias-aware inference.

3. Pilot Study

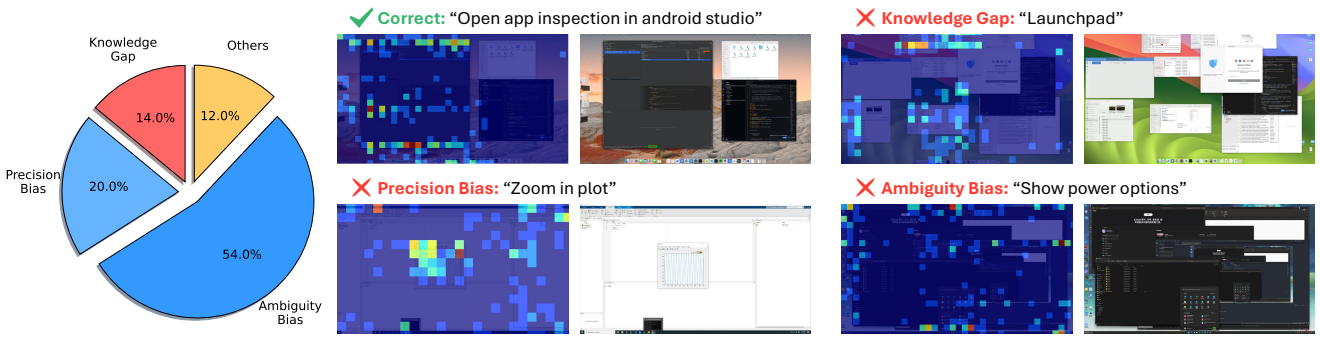
On ScreenSpot-Pro [12], a challenging GUI grounding benchmarks, the accuracy of state-of-the-art grounding models on these benchmarks has significantly decreased, falling below 50%. To gain deeper insights into the underlying performance bottlenecks, we conducted a systematic pilot study addressing two primary questions: *(1) What are the root causes of errors made by GUI grounding models? (2) How can these errors be mitigated from a model mechanism perspective without the need for retraining?*

3.1. Error Attribution via MPD

This section uses the ScreenSpot-Pro dataset [12] as a benchmark to analyze potential error patterns in GUI grounding models and explore corresponding mitigation strategies.

Problem Formulation For a GUI grounding model f, given a query q and a GUI screenshot $I \in \mathbb{R}^{H\times W\times 3}$, the model generates a text sequence t containing the target bounding box in the standard format: <|box_start|>(x_1, y_1, x_2, y_2)<|box_end|>. The coordinates $(x_1, y_1, x_2, y_2) = r(t)$ are extracted using a regular expression parser r, where (x_1, y_1) and (x_2, y_2) represent the top-left and bottom-right coordinates of the bounding box, respectively. The center coordinates of the bounding box are computed as: $(x_c, y_c) = \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}\right)$. A prediction is considered correct if the center coordinate (x_c, y_c) lies within the ground-truth bounding box; otherwise, it is deemed an error.

Attribution Method Traditional gradient-based attribution methods (e.g., GradCAM [22], Integrated Gradients [24]) are not well-suited for the discrete text-to-coordinate conversion process. As an alternative, we initially considered using Shapley values [18, 23] for attribution analysis. For an n-dimensional input feature, the Shapley value for the i-th feature is defined as: $\phi_i = \sum_{S\subseteq\{1,2,\ldots,n\}\setminus\{i\}} \frac{|S|!(n-|S|-1)!}{n!}\left[f(S\cup\{i\}) - f(S)\right]$, where S denotes a subset of features. However, due to the high resolution of GUI screenshots, estimating the Shapley values [1] for a single sample takes approximately 10 hours on a single RTX 4090 GPU, which is computationally impractical. To address this, we propose the **Masked Prediction Distribution (MPD)** method, which efficiently observes the spatial distribution of model predictions under random perturbations (see Supplementary Algorithm 1). Regions with densely distributed predicted points indicate high model confidence in those areas. We set the number of perturbations to 300 per sample and can obtain heatmaps within 20 minutes per sample.



(a) (b)

Figure 4. **Error Attribution Analysis.** (a) Proportions of attribution types. (b) Attribution analysis of model predictions. The deep red regions in the heatmap indicate potential prediction locations, demonstrating how the MPD can clearly identify the sources of model errors.

Table 1. Proportions and detailed analysis of different error types.

Error Type	Description
Knowledge Gap (14%)	Model fails to recognize target information. 7 error samples.
Precision Bias (20%)	Model identifies target but exhibits systematic offset. 10 samples.
Ambiguity Bias (54%)	Model distracted by similar regions or misleading semantics. 27 samples.
Others (12%)	Unclassified patterns. 6 samples.

Error Pattern Analysis Based on the experimental results of TianXi-Action-7B [26] on ScreenSpot-Pro, we conducted an attribution analysis on 50 error samples, with the findings summarized in Table 1. Notably, both precision bias and ambiguity bias are categorized as inductive bias issues, collectively accounting for 74% of the error samples. This indicates that if we can effectively mitigate inductive bias, the model's performance will be significantly improved.

3.2. Mitigation Strategy: Inductive Bias Correction

Based on the error pattern analysis, we explored potential mitigation methods for different error types. Knowledge gap errors reflect limitations in the model's training data or architecture, which are difficult to address with inference-time techniques. In contrast, inductive bias errors (precision bias and ambiguity bias) can potentially be mitigated through optimization of the inference mechanism.

Limitations of Language-Space Enhancement Inspired by reasoning techniques in large language models (e.g., Chain-of-Thought [27]), we first attempted to enhance GUI grounding performance by augmenting linguistic information. **(1) Query Expansion Strategy:** For queries with insufficient or ambiguous descriptions, we used a language model to expand and refine the original query, generating more precise instruction information. **(2) Context Expansion Strategy:** We utilized a multimodal large language model (e.g., Qwen2.5-VL [2]) to generate a structured description of the GUI, including the geometric location, text content, and other information of UI elements, and concatenated this with the original query as model input. However, experimental results indicated that merely extending the language sequence did not significantly improve model accuracy, and even introduced additional errors in some cases. This phenomenon aligns with recent findings [33] that traditional linguistic reasoning models are difficult to directly transfer to precise grounding tasks.

Root Causes of Precision Bias An in-depth analysis of precision bias revealed that multimodal models typically adopt discretized coordinate representations for images with resolution $H \times W$. For instance, in Qwen series models, a coordinate value of $x_1 = 789$ is split into independent digit characters (<7>, <8>, <9>) and further converted into their corresponding token IDs. This discretization inherently limits the model's maximum precision to the unit digit level.

Root Causes of Ambiguity Bias The cross-entropy training objective for multimodal models optimizes the **edit distance** of token sequences rather than the **Euclidean distance**. Let the ground-truth coordinate be $x_{\text{GT}} = 789$, and consider two predicted candidates: $x' = 189$ and $x'' = 801$. A direct comparison of the two metrics yields:

$$d_{\text{edit}}(x_{\text{GT}}, x') = 1 < d_{\text{edit}}(x_{\text{GT}}, x'') = 3$$
$$d_{\text{euc}}(x_{\text{GT}}, x') = 600 > d_{\text{euc}}(x_{\text{GT}}, x'') = 12$$

This inconsistency in metrics causes a fundamental conflict between the model's optimization objective in token space and the need for accuracy in real-world spatial localization. Therefore, external correction mechanisms combining token sequence optimization with geometric constraints are necessary to address this systematic bias.

4. Method

Based on the experimental results from the pilot study, we design the **BAMI** method in this section. The method tar-

Algorithm 1 BAMI (with N crop iterations and M candidates per iteration)

Require: Query q, screenshot I, correction model m, and grounding model f
Ensure: Grounding point (x, y)

Initialize the input image as $I^1 = I$
for all $t \in \{1, 2, \cdots, N\}$ **do**
  Initialize the candidate box set $\Phi^t = \emptyset$
  for all $i \in \{1, 2, \cdots, M\}$ **do**
    Masking all pixels in the candidate set to get input image $I_i^t = \text{MASK}(I^{t-1}, \Phi^t)$
    Predict the candidate box $b_i^t = f(q, I_i^t)$ and update $\Phi^t \leftarrow \Phi^t \cup \{b_i^t\}$
  end for
  Select the preferred box $\tilde{b}^t = m(q, I^t, \Phi^t)$
  Crop the input image $I^{t+1} = \text{CROP}(I^t, \tilde{b}^t)$
end for
Compute the center point of $\tilde{b}^N$ as (x, y)

gets both accuracy bias and ambiguity bias, and proposes different manipulations to improve GUI grounding.

4.1. Accuracy Bias: Coarse-to-Fine Focus

The root cause of accuracy bias lies in the discretization process of multimodal large models during coordinate localization. Since the prediction accuracy of the model is typically limited to the pixel level, and its output is difficult to be perfectly accurate, prediction errors may sometimes reach tens or even hundreds of pixels. To effectively eliminate accuracy bias, inspired by human observation strategies, we propose a **coarse-to-fine focus** manipulation. Specifically, we first use the grounding model to predict a coarse localization coordinate (x^t, y^t). Then, based on this coarse coordinate, we crop the original image to a scale of $\lambda < 1$, and input the cropped image back into the grounding model for fine localization, obtaining a more precise coordinate (x^{t+1}, y^{t+1}). Although this process can be iterated multiple times, we find that there is a trade-off in the hyperparameters. **(1) Iteration count:** After a certain number of iterations, the performance improvement of the model tends to plateau; **(2) Crop ratio:** A large cropping ratio may lead to the loss of crucial information, while a small cropping ratio may prevent the model from accurately localizing the target.

4.2. Ambiguity Bias: Candidate Correction

Multimodal large models (MLLMs) represent coordinates as text sequences for autoregressive generation. While this design simplifies the training process, it introduces a discrepancy between the training and inference phases. For example, the coordinate "789" is encoded into the text sequence `<7><8><9>`, and the model minimizes the edit distance of this text sequence using cross-entropy loss. In practice, however, the impact of digit position errors is asymmetric: an error in the hundreds place is two orders of magnitude more significant than that in the ones place. This results in a substantial mismatch between edit distance and Euclidean distance, and no straightforward mapping exists to convert the former to the latter. To eliminate ambiguity bias, we first generate multiple mutually exclusive candidate bounding boxes through multi-round masked prediction operations. Subsequently, we utilize a correction model to re-select from these candidate boxes. We investigate both online APIs (e.g. GPT-5) and locally trained models (e.g. Qwen3-VL-8B) for this role. Notably, the key to this operation lies in prompt design; naive prompt designs fail to leverage the correction model effectively. To enable the correction model to rectify the erroneous ordering tendency of the grounding model, we incorporate **key principles** consistent with GUI priors into the prompt. Examples of these principles are provided below, and detailed prompt design is available in the appendix.

Prompt

```
- (Functional Preference) Focus on the functional purpose of the highlighted elements
- (Memory Comparison) Consider standard patterns (e.g., buttons for actions)
- (Interactive Components) Prioritize interactive elements over static text/labels
```

4.3. BAMI: Bias-Aware Manipulation Inference

By integrating the two manipulations outlined above, we propose **BAMI**, as illustrated in Figure 5. To enhance the diversity of candidate boxes, we mask the pixels within already predicted candidate boxes prior to each new prediction step, thereby ensuring the mutual exclusivity between newly generated candidate boxes and existing results. To mitigate precision bias, BAMI adopts a coarse-to-fine focus strategy in its outer loop, enabling gradual refinement of focus toward more accurate coordinate positions step by step. Simultaneously, to address ambiguity bias, BAMI employs a candidate selection strategy in each iteration, selecting the most suitable box from multiple candidates as the final output. The algorithm of BAMI is detailed in Algorithm 1.

5. Experiment

5.1. Experimental Setup

Models The proposed BAMI method aims to enhance the accuracy of Grounding models without retraining. We tested this method on several state-of-the-art grounding models, including OS-Atlas-7B [30], UI-TARS-1.5-7B [21], and TianXi-Action-7B [26]. All models were implemented using the Transformers framework [28] for inference. The input to the models consists of both the query and the screenshot. OS-Atlas and TianXi-Action output bounding box coordinates, while UI-TARS outputs click coordinates.

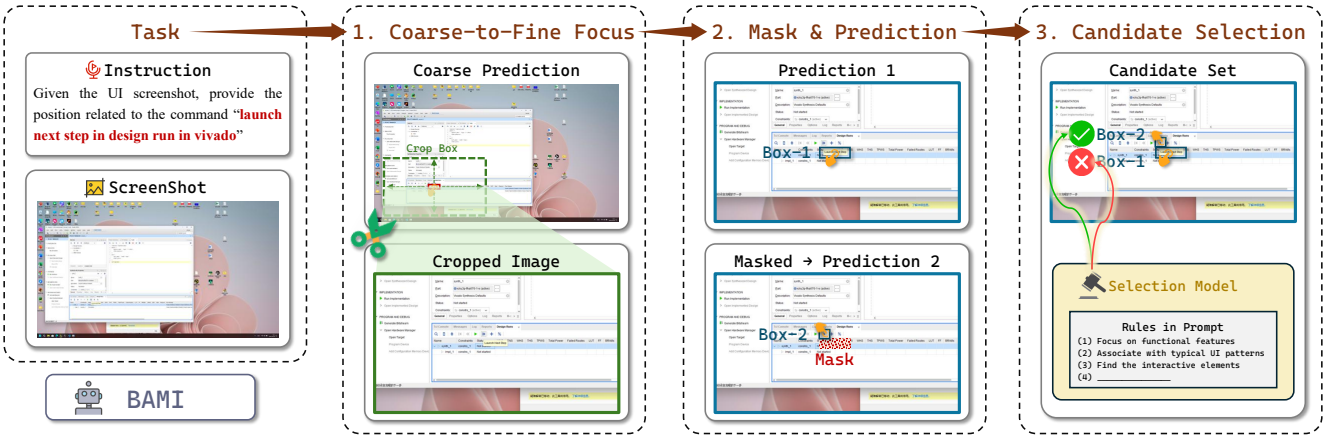


Figure 5. **Illustration of BAMI. Step 1:** Based on the initial prediction results of the grounding model, BAMI performs cropping around these initial predictions at a predefined ratio. **Step 2:** The model conducts multiple predictions on the cropped images; after each prediction, the pixels within the predicted bounding box are randomly masked to ensure the diversity of multiple prediction results. **Step 3:** Using predefined rules and an external knowledge model, the model ranks multiple candidate coordinates and selects the final coordinates.

Data We evaluate BAMI on ScreenSpot-V2 [30], and ScreenSpot-Pro [12]. ScreenSpot-V2 are mainly used to assess grounding accuracy in simple scenarios, covering mobile, web, and desktop. ScreenSpot-Pro focuses on complex scenarios, consisting of high-resolution screenshots of professional software, where each sample contains multiple software elements, and the targets are typically small, making it a particularly challenging task.

Hyperparameters To balance efficiency and accuracy, two iterations were adopted for the coarse-to-fine focusing process. For high-resolution screenshots, the crop ratio λ was set to the range $[0.5, 0.7]$. To eliminate ambiguity bias, a masking mechanism was employed, which generates $2 \sim 3$ candidate results per iteration; subsequently, a correction model was used to select the result most relevant to the query. We evaluate both online (GPT-5) and offline (Qwen3-VL-8B) variants. All experiments were conducted on a single RTX 4090 GPU.

5.2. Comparison with SOTA

We first evaluated state-of-the-art grounding methods on the complex ScreenSpot-Pro dataset, with results summarized in Table 2. All models were categorized into three groups based on their training or deployment paradigms: supervised fine-tuning (SFT) training, reinforcement learning (RL) training, and test-time inference. Among 7B-scale models, our BAMI method achieved the best performance on the ScreenSpot-Pro dataset, attaining an accuracy of 57.8%. Built upon TianXi-Action-7B [26], our model delivers a 5.9% accuracy improvement. We present the consistent improvements of BAMI across different base models in Table 3. Additionally, we conducted experiments on the ScreenSpot-V2 dataset, with detailed results in the supplementary material, which demonstrates that our method outperforms all baseline models to varying extents. Furthermore, Table 4 (left) dissects two key manipulations of BAMI, where "PB" and "AB" denote precision bias and ambiguity bias elimination, respectively. It can be observed that both manipulations independently yield significant improvements in model accuracy.

5.3. Ablation Studies

This section presents a series of ablation experiments to validate the effectiveness of the accuracy bias and ambiguity bias elimination manipulations in BAMI and explores the impact of different parameter settings on model performance.

5.3.1. Accuracy Bias Elimination

Impact of Crop Ratio and Iteration Number We first investigate the effects of the number of crops and the crop ratio λ on the elimination of accuracy bias, as illustrated in Figure 6a. For the crop ratio, the number of iterations is fixed at 2. When the crop ratio exceeds 40%, the elimination effect on precision bias becomes relatively significant. If the crop ratio is set too aggressively (i.e., less than 40%), it may lead to the cropping of crucial contextual information, compromising model performance. For the number of iterations, the crop ratio is fixed at 50%. It can be observed that 2 iterations are sufficient to eliminate precision bias. Excessive iterations may result in an overly large overall cropping ratio, which conversely degrades the performance.

Impact of Target Type We also investigate whether BAMI exhibits selectivity across different targets, as illustrated in Figure 6b. We calculated the Euclidean distance between predictions and ground truth, where the blue line represents

Table 2. Comparison with various models on ScreenSpot-Pro.

Grounding Model	Development		Creative		CAD		Scientific		Office		OS		Avg.
	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	
Proprietary Models													
GPT-4o [11]	2.0	0.0	1.3	0.0	1.0	0.0	2.1	0.0	1.1	0.0	0.0	0.0	0.8
Claude Computer Use [10]	14.5	3.7	22.0	3.9	25.9	3.4	33.9	15.8	30.1	16.3	11.0	4.5	17.1
General Open-source Models													
Qwen2.5-VL-3B [2]	9.1	7.3	22.1	1.4	26.8	2.1	38.2	7.3	33.9	15.1	10.3	1.1	16.1
Qwen2.5-VL-7B [2]	16.8	1.6	46.8	4.1	35.9	7.7	49.3	7.3	52.5	20.8	37.4	6.7	26.8
GUI-specific Models (SFT)													
SeeClick-9.6B [3]	2.5	0.0	0.6	0.0	1.0	0.0	3.5	0.0	1.1	0.0	2.8	0.0	1.1
CogAgent-18B [9]	7.1	3.1	14.9	0.7	9.6	0.0	22.2	1.8	13.0	0.0	5.6	0.0	7.7
OS-Atlas-7B [30]	12.2	4.7	33.1	1.4	28.8	2.8	37.5	7.3	33.9	5.7	27.1	4.5	18.9
ShowUI-2B [13]	2.5	0.0	16.9	1.4	9.1	0.0	13.2	7.3	15.3	7.5	10.3	2.2	7.7
UGround-7B [6]	14.2	1.6	26.6	2.1	27.3	2.8	31.9	2.7	31.6	11.3	17.8	0.0	16.5
UGround-V1-7B [6]	15.8	1.2	51.9	2.8	47.5	9.7	57.6	14.5	60.5	13.2	38.3	7.9	31.1
UI-TARS-7B [21]	20.8	9.4	58.4	12.4	50.0	9.1	63.9	31.8	63.3	20.8	30.8	16.9	35.7
TianXi-Action-7B [26]	**76.0**	**21.4**	**61.6**	**19.6**	**45.2**	**18.8**	**80.6**	**31.8**	**84.2**	**54.7**	**57.9**	**33.7**	**51.9**
GUI-specific Models (RL)													
UI-R1-3B [17]	11.2	6.3	22.7	4.1	27.3	3.5	42.4	11.8	32.2	11.3	13.1	4.5	17.8
UI-R1-E-3B [17]	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-R1-7B [19]	23.9	6.3	49.4	4.8	38.9	8.4	55.6	11.8	58.7	26.4	42.1	16.9	-
InfiGUI-R1-3B [15]	33.0	14.1	51.3	12.4	44.9	7.0	58.3	20.0	65.5	28.3	43.9	12.4	35.7
GUI-G1-3B [35]	39.6	9.4	50.7	10.3	36.6	11.9	61.8	30.0	67.2	32.1	23.5	10.6	37.1
SE-GUI-7B [32]	51.3	42.2	68.2	19.3	57.6	9.1	75.0	28.2	78.5	43.4	49.5	25.8	47.3
GUI-G2-7B [25]	**55.8**	**12.5**	**68.8**	**17.2**	**57.1**	**15.4**	**77.1**	**24.5**	**74.0**	**32.7**	**57.9**	**21.3**	**47.5**
Test-Time Methods													
GUI-RC [5]	-	-	-	-	-	-	-	-	-	-	-	-	41.2
DiMo-GUI-7B [29]	66.9	21.4	60.6	21.7	50.3	14.1	68.1	21.8	80.8	52.8	69.2	28.1	49.7
BAMI-7B	**81.8**	**26.9**	**68.2**	**23.8**	**58.4**	**29.7**	**77.8**	**36.4**	**83.6**	**60.4**	**72.9**	**33.3**	**57.8**

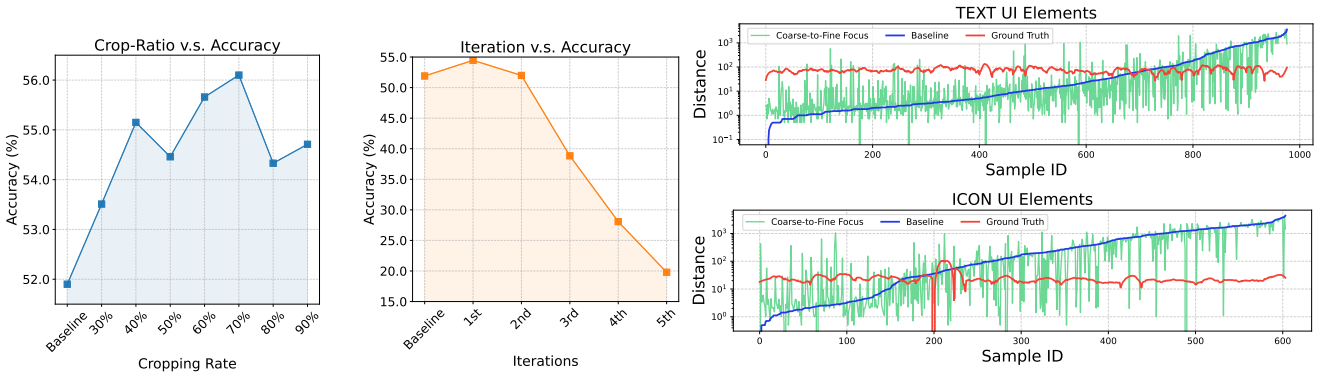
Table 3. Comparison with different baseline models on ScreenSpot-Pro.

Grounding Model	Development		Creative		CAD		Scientific		Office		OS		Avg.
	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	
UGround-7B [6]	14.2	1.6	26.6	2.1	27.3	2.8	31.9	2.7	31.6	11.3	17.8	0.0	16.5
+ BAMI	**48.7**	**5.5**	**46.5**	**7.7**	**18.3**	**4.7**	**54.9**	**14.6**	**52.5**	**18.9**	**42.8**	**9.4**	**30.0**
OS-Atlas-7B [30]	12.2	4.7	33.1	1.4	28.8	2.8	37.5	7.3	33.9	5.7	27.1	4.5	18.9
+ BAMI	**66.2**	**16.6**	**58.6**	**16.1**	**36.0**	**10.9**	**55.6**	**17.3**	**68.4**	**22.6**	**56.8**	**17.1**	**41.6**
UI-TARS-1.5-7B [21]	50.0	14.5	56.6	13.3	37.6	12.5	66.0	22.7	76.3	34.0	55.6	16.9	40.8
+ BAMI	**71.4**	**22.1**	**68.2**	**21.7**	**49.8**	**14.1**	**77.8**	**23.6**	**82.5**	**41.5**	**69.1**	**24.2**	**51.9**
TianXi-Action-7B [26]	76.0	21.4	61.6	19.6	45.2	18.8	80.6	31.8	84.2	54.7	57.9	33.7	51.9
+ BAMI (GPT-5)	**81.8**	**26.9**	**68.2**	**23.8**	**58.4**	**29.7**	**77.8**	**36.4**	**83.6**	**60.4**	**72.9**	**33.3**	**57.8**
+ BAMI (Local)	**80.5**	**26.9**	**66.7**	**21.0**	**53.8**	**28.1**	**78.5**	**34.6**	**83.1**	**62.3**	**71.3**	**31.6**	**56.2**

the baseline and the green line represents BAMI. The red line denotes the distance between corner points and the center of the ground truth bounding box. If a prediction line lies below the red line, the prediction is highly likely correct. We sorted the baseline results in ascending order for ease of observation. For both text and icon types, the bias distance of BAMI's predictions is mostly smaller than that of the baseline. This indicates that BAMI has no selective preference for predicted targets and possesses universality.

5.3.2. Ambiguity Bias Elimination

Impact of Prompt Design A key reason for ambiguity bias lies in the fact that MLLMs prioritize candidate outcomes based on edit distance. Therefore, it is crucial to inject priority priors in the Euclidean space through prompt design. We conducted ablation experiments on two important prompt



(a) Ablations on crop ratio and iteration number.

(b) Impact of different target types.

Figure 6. **Ablations on accuracy bias elimination.** (a) Effect of crop ratio and iteration count. (b) Performance across different target types.

Table 4. Ablation studies on BAMI components and prompt design.

Manipulations				Prompt Design			
Setting	**PB**	**AB**	**Acc.**	**Setting**	**CoT**	**KP**	**Acc.**
Baseline			51.9	Baseline			51.9
+ C2F Focus	✓		55.2	+ Vanilla			55.7
+ Cand. Sel.		✓	54.3	+ w/ CoT	✓		57.0
+ BAMI	✓	✓	**57.8**	**+ w/ CoT&KP**	✓	✓	**57.8**

Table 5. Impact of different correction models (online and offline).

Category	**Correction Model**	**Accuracy**
Baseline		51.9
Online APIs	Doubao-Seed-1.6-Flash	55.3
	GLM-4.5V	55.9
	Qwen-VL-Max	56.4
	Gemini-2.5-Pro	57.2
	GPT-5	**57.8**
Local Model	Qwen3-VL-8B (Ours)	56.2

structures in BAMI, as shown in Table 4 (right). "CoT" denotes the chain-of-thought-style prompt, which aims to enable the correction model to make selections in a more granular manner. "KP" stands for key principle, a critical component for injecting coordinate space priority priors into the selection process. Experimental results demonstrate that injecting Euclidean space priors into the correction model significantly enhances the accuracy of BAMI.

Impact of Correction Model Selection We investigated the impact of correction model selection, as presented in Table 5. Among online APIs, GPT-5 and Gemini-2.5-Pro achieved the best performance, enabling an overall accuracy of over 57%. All tested models contributed to performance improvement, indicating BAMI's robustness across different correction models. To address privacy requirements and enable independent deployment, we trained a local Qwen3-VL-8B correction model (8B parameters, matching grounding models' scale) via LoRA fine-tuning on 128K dual-box samples. As shown in Table 5 and Table 3, it achieves 56.2Training details are in the supplementary material.

6. Discussion

This paper investigates how to improve GUI grounding in a training-free manner. Through **MPD**, we identify that most errors on complex GUIs stem from inductive bias, mainly accuracy bias and ambiguity bias. **BAMI** extends the model's reasoning process through structured inference with two critical manipulations, coarse-to-fine focus and candidate selection, which substantially alleviate these biases. On ScreenSpot-Pro, BAMI improves TianXi-Action-7B from 51.9% to 57.8%, showing clear advantages over contemporary training-free methods in both effectiveness and efficiency. To address privacy concerns and avoid relying on the extra knowledge implicitly introduced by external APIs, we further train a local correction-model variant based on Qwen3-VL-8B using public data, which reaches 56.2%. In particular, this local model uses a parameter scale comparable to the grounding models themselves, demonstrating that effective correction can be achieved without requiring significantly larger architectures. Beyond this practical variant, we will further investigate how the model's inductive preferences diverge from real-world GUI scenarios, with the goal of developing more generalizable solutions.

Acknowledgement

This work was supported in part by the Beijing Natural Science Foundation under Grant No. L247009, and the National Natural Science Foundation of China under Grant 62125603, Grant 62336004, Grant 62321005.

References

[1] Marco Ancona, Cengiz Oztireli, and Markus Gross. Explaining deep neural networks with a polynomial time algorithm for shapley value approximation. In *ICML*, pages 272–281, 2019.

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

[3] Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. *arXiv*, abs/2401.10935, 2024.

[4] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *NeurIPS*, 36:28091–28114, 2023.

[5] Yong Du, Yuchen Yan, Fei Tang, Zhengxi Lu, Chang Zong, Weiming Lu, Shengpei Jiang, and Yongliang Shen. Test-time reinforcement learning for gui grounding via region consistency. *arXiv*, abs/2508.05615, 2025.

[6] Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for gui agents. In *ICLR*, 2025.

[7] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv*, abs/2501.12948, 2025.

[8] Izzeddin Gur, Hiroki Furuta, Austin V Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. In *ICLR*, 2024.

[9] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *CVPR*, 2024.

[10] Siyuan Hu, Mingyu Ouyang, Difei Gao, and Mike Zheng Shou. The dawn of gui agent: A preliminary case study with claude 3.5 computer use. *arXiv*, abs/2411.10323, 2024.

[11] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv*, abs/2410.21276, 2024.

[12] 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. *arXiv*, abs/2504.07981, 2025.

[13] Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Stan Weixian Lei, Lijuan Wang, and Mike Zheng Shou. Showui: One vision-language-action model for gui visual agent. In *CVPR*, pages 19498–19508, 2025.

[14] Shi Liu, Kecheng Zheng, and Wei Chen. Paying more attention to image: A training-free method for alleviating hallucination in lvlms. In *European Conference on Computer Vision*, pages 125–140. Springer, 2024.

[15] 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. *arXiv*, abs/2504.14239, 2025.

[16] Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. Omniparser for pure vision based gui agent. *arXiv*, abs/2408.00203, 2024.

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

[18] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *NeurIPS*, 30, 2017.

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

[20] Joonhyung Park, Peng Tang, Sagnik Das, Srikar Appalaraju, Kunwar Yashraj Singh, R Manmatha, and Shabnam Ghadar. R-vlm: Region-aware vision language model for precise gui grounding. *arXiv*, abs/2507.05673, 2025.

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

[22] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017.

[23] Lloyd S Shapley et al. A value for n-person games. 1953.

[24] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, pages 3319–3328, 2017.

[25] Fei Tang, Zhangxuan Gu, Zhengxi Lu, Xuyang Liu, Shuheng Shen, Changhua Meng, Wen Wang, Wenqi Zhang, Yongliang Shen, Weiming Lu, et al. Gui-g2: Gaussian reward modeling for gui grounding. *arXiv*, abs/2507.15846, 2025.

[26] Liang Tang, Shuxian Li, Yuhao Cheng, Yukang Huo, Zhepeng Wang, Yiqiang Yan, Kaer Huang, Yanzhe Jing, and Tiaonan Duan. Sea: Self-evolution agent with step-wise reward for computer use. *arXiv*, abs/2508.04037, 2025.

[27] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *NeurIPS*, 35:24824–24837, 2022.

[28] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv*, abs/1910.03771, 2019.

[29] Hang Wu, Hongkai Chen, Yujun Cai, Chang Liu, Qingwen Ye, Ming-Hsuan Yang, and Yiwei Wang. Dimo-gui: Advancing test-time scaling in gui grounding via modality-aware visual reasoning. *arXiv*, abs/2507.00008, 2025.

[30] Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, et al. Os-atlas: A foundation action model for generalist gui agents. *arXiv*, abs/2410.23218, 2024.

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

[32] Xinbin Yuan, Jian Zhang, Kaixin Li, Zhuoxuan Cai, Lujian Yao, Jie Chen, Enguang Wang, Qibin Hou, Jinwei Chen, Peng-Tao Jiang, et al. Enhancing visual grounding for gui agents via self-evolutionary reinforcement learning. *arXiv*, abs/2505.12370, 2025.

[33] Li Zhang, Longxi Gao, and Mengwei Xu. Does chain-of-thought reasoning help mobile gui agent? an empirical study. *arXiv*, abs/2503.16788, 2025.

[34] Miaosen Zhang, Ziqiang Xu, Jialiang Zhu, Qi Dai, Kai Qiu, Yifan Yang, Chong Luo, Tianyi Chen, Justin Wagle, Tim Franklin, et al. Phi-ground tech report: Advancing perception in gui grounding. *arXiv*, abs/2507.23779, 2025.

[35] Yuqi Zhou, Sunhao Dai, Shuai Wang, Kaiwen Zhou, Qinglin Jia, and Jun Xu. Gui-g1: Understanding r1-zero-like training for visual grounding in gui agents. *arXiv*, abs/2505.15810, 2025.