DiMo-GUI: Advancing Test-time Scaling in GUI Grounding via Modality-Aware Visual Reasoning

Hang Wu[1,3], **Hongkai Chen**[3†], **Yujun Cai**[2], **Chang Liu**[3],
Qingwen Ye[3], **Ming-Hsuan Yang**[1], **Yiwei Wang**[1]
[1]University of California, Merced, [2]The University of Queensland,
[3]vivo Mobile Communication Co., Ltd
hangwu@ucmerced.edu, allenhkchen@gmail.com
https://wuhang03.github.io/DiMo-GUI-homepage/

Abstract

Grounding natural language queries in graphical user interfaces (GUIs) poses unique challenges due to the diversity of visual elements, spatial clutter, and the ambiguity of language. In this paper, we introduce DiMo-GUI, a training-free framework for GUI grounding that leverages two core strategies: dynamic visual grounding and modality-aware optimization. Instead of treating the GUI as a monolithic image, our method splits the input into textual elements and iconic elements, allowing the model to reason over each modality independently using general-purpose vision-language models. When predictions are ambiguous or incorrect, DiMo-GUI dynamically focuses attention by generating candidate focal regions centered on the model's initial predictions and incrementally zooms into subregions to refine the grounding result. This hierarchical refinement process helps disambiguate visually crowded layouts without the need for additional training or annotations. We evaluate our approach on standard GUI grounding benchmarks and demonstrate consistent improvements over baseline inference pipelines, highlighting the effectiveness of combining modality separation with region-focused reasoning.

1 Introduction

Graphical user interface (GUI) agents play an increasingly central role in modern computing, allowing a wide range of applications, from automated web navigation to intuitive control of operating systems (Anderson et al., 2018; Liu et al., 2024b). With the rise of large-scale vision-language models (VLMs), recent research has focused on leveraging both visual and textual modalities to build more intelligent and interactive agents. However, many existing frameworks rely predominantly on text-based reasoning (Yang et al., 2024a) or adopt simplistic visual grounding strategies (Lu et al., 2024; Gou et al., 2024). In practice, real-world GUIs often contain a large number of irrelevant or distracting elements—such as menu bars, advertisements, or extraneous buttons—that can overwhelm purely text-driven or naive visual approaches. This discrepancy between text-heavy inference and the complex visual nature of GUIs frequently results in errors, such as clicking incorrect buttons or navigating to unintended regions. Given that these agents are often tasked with high-level decision making, such low-level mistakes can accumulate, ultimately degrading overall performance and task success rates.

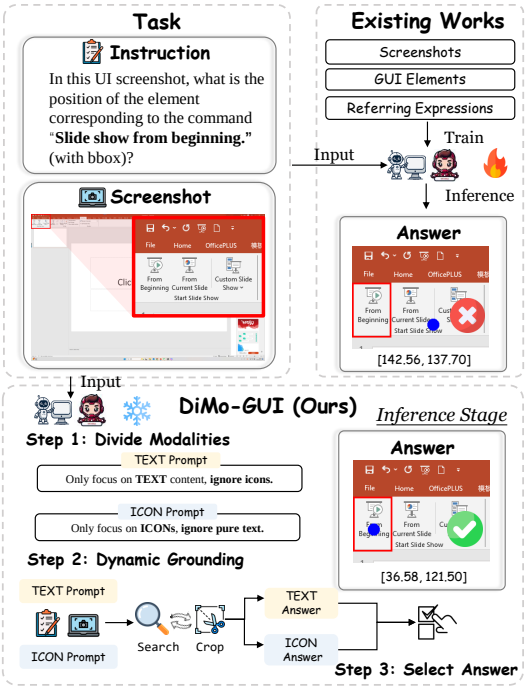

Figure 1: **Overview.** DiMo-GUI searches separately within the text and icon elements based on the instruction and the screenshot.

Recent work on GUI agents generally falls into two major paradigms: one centered on text-based

[†]The corresponding author.
[†]The work was done during the first author's internship at vivo Company.

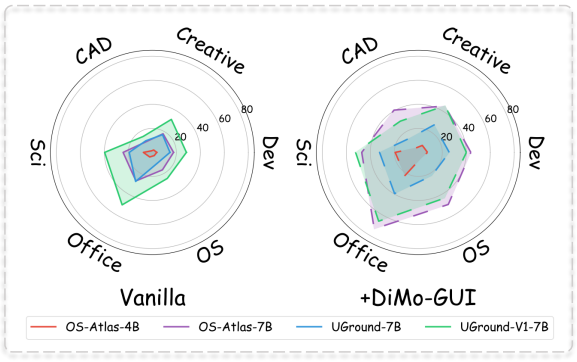

Figure 2: **Grounding Performance of DiMo-GUI.** By integrating DiMo-GUI, existing models can achieve significant performance improvements on the dataset.

reasoning and planning, and the other grounded in visual understanding through VLMs. Text-focused methods typically generate textual descriptions or bounding boxes for each visual element to inform action decisions (Lu et al., 2024). However, these approaches struggle with visually complex scenarios where text descriptions are ambiguous, incomplete, or fail to capture crucial visual cues—such as floating windows or dynamic pop-ups—even when assisted by accessibility trees. In contrast, vision-based pipelines (Gou et al., 2024; Qin et al., 2025) rely heavily on the grounding capabilities of VLMs, but are prone to errors such as clicking on empty or incorrect regions due to limitations in one-shot visual inference. Critically, these systems often lack an error correction mechanism; once a mistake occurs, it goes unaddressed, compounding over time and leading to cascading failures during multi-step interaction tasks.

To address these limitations, we propose DiMo-GUI, a training-free GUI grounding framework that integrates progressive zoom-in refinement and modality-specific processing. Instead of relying on a single forward pass, DiMo-GUI starts from a coarse prediction of the focal region and iteratively narrows the focus by refining bounding boxes around the target. Meanwhile, it separates textual and graphical components within the GUI and processes them with tailored strategies, allowing the agent to better handle diverse content types. As shown in Fig. 1, this design avoids the need for additional training, and can be plugged into existing GUI agents. Empirically, we find that this step-wise, disentangled grounding pipeline significantly improves robustness in visually cluttered or ambiguous environments, while maintaining compatibility with general-purpose VLMs.

We evaluate the proposed DiMo-GUI framework on the recently released ScreenSpot-Pro dataset by integrating it into several state-of-the-art models as reported in the original paper. Without modifying the model architecture or requiring any additional training, DiMo-GUI brings a clear performance improvement across key evaluation metrics as shown in Fig. 2. These results demonstrate the effectiveness and generalizability of our training-free design in enhancing GUI grounding performance in existing large-scale models. Our main contributions can be summarized as follows:

- We propose DiMo-GUI, a training-free framework that can be seamlessly integrated as a plug-and-play component into any GUI agent. Without requiring additional training or external data, DiMo-GUI effectively enhances grounding performance across various GUI tasks.
- DiMo-GUI introduces three key innovations: (1) a divide-and-conquer strategy that separates text and icon components for targeted processing, (2) a progressive zoom-in mechanism to increasingly focus on the target region, and (3) a dynamic halting system that enables timely decision-making and early DiMo-GUIping to reduce overthinking and unnecessary computational cost.
- Extensive and comprehensive experiments demonstrate that DiMo-GUI can significantly enhance the grounding performance of various GUI agents across multiple benchmarks with minimal computational overhead, showcasing the effectiveness and generalizability of the proposed framework.

2 Related Work

2.1 GUI Agents

Recent years have witnessed significant advances in GUI automation driven by large language models (LLMs). Early GUI agents predominantly focused on web interactions (Nakano et al., 2022; Hong et al., 2024) and have gradually expanded to mobile (Zhang et al., 2023; Wang et al., 2024a) and desktop environments (Zhang et al., 2024). A fundamental challenge across these applications is precise element localization. Traditional approaches relied on structured information like XML and DOM trees (Zhang et al., 2023), but faced limitations in accessibility and information redundancy. Alternative

methods using OCR (Du et al., 2020) or detection models (Liu et al., 2024a) introduced additional computational overhead. Recent advances in multimodal large language models (MLLMs) have enabled direct GUI element localization (Hong et al., 2024; Cheng et al., 2024; Lin et al., 2024), partially bridging the visual perception gap. (Tang et al., 2025) introduces a dual-system framework that combines fast prediction with systematic analysis to provide robust GUI foundation. OS-Atlas (Wu et al., 2024) and UGround (Gou et al., 2024) created large datasets and trained models to handle out-of-distribution tasks. (Zhou et al., 2025; Lee et al., 2025a; Yuan et al., 2025; Xia and Luo, 2025) explored improving grounding performance using reinforcement learning.

2.2 Test-time scaling

Test-time scaling dynamically adjusts computational resources during inference to enhance model performance, with recent studies showing it can outperform increased train-time computation through strategies like best-of-N sampling and external verification (Snell et al., 2024; Lee et al., 2025b; Hosseini et al., 2024). In localization tasks, test-time scaling has also been framed as a search problem (Wu and Xie, 2024). Inspired by its success in LLMs, similar techniques have been applied to GUI agents, such as leveraging action histories (Zhang and Zhang, 2023), gathering external information (Nakano et al., 2022), zooming in and searching (Nguyen, 2024), and adaptively refining focus regions (Luo et al., 2025).

3 Methodology

To address the limitations of existing GUI agents in handling high-resolution images and their imbalanced performance between text and icon understanding, we propose a novel framework called DiMo-GUI. As shown in the algorithm 1, our method integrates a dynamic zooming mechanism and a modality decoupling strategy. Specifically, DiMo-GUI dynamically narrows down the target region through iterative zooming on the input high-resolution screenshot, progressively refining the localization until the target coordinates are identified. In parallel, DiMo-GUI decouples text-based and icon-based GUI elements, processing each modality independently to reduce cross-modal interference. This design mitigates a common shortcoming of vision-language models (VLMs), which typically exhibit stronger capabilities in text understanding compared to visual icon interpretation.

Algorithm 1: Dual-Modality Grounding with Dynamic Zooming

Input: Full-resolution GUI image I, instruction Q
Output: Final grounded coordinate C^*

Step 1: Text modality grounding.
$C_{\text{text}} \leftarrow \texttt{DynamicGrounding}(I, Q, \text{“text”})$
Step 2: Icon modality grounding.
$C_{\text{icon}} \leftarrow \texttt{DynamicGrounding}(I, Q, \text{“icon”})$
Step 3: Candidate selection.
$C^* \leftarrow \texttt{Select}(C_{\text{text}}, C_{\text{icon}}, I, Q)$
return C^*

Function: `DynamicGrounding`(I, Q, modality m)
Initialize zoom region: $R \leftarrow I$
for $t = 1$ **to** *max_iters* **do**
  Predict coordinate:
  $C_t \leftarrow \texttt{PredictCoordinate}(R, Q, m)$
  if `DiMo-GUICondition`(C_t, t) **then**
    return C_t
  $R \leftarrow \texttt{CropAround}(R, C_t)$ // update region
return C_{max}

3.1 Dynamic Grounding Mechanism

High resolution remains one of the most significant challenges in GUI grounding, often leading to long inference times and excessive visual redundancy. A natural solution to this problem is to iteratively narrow down the target region, progressively refining the prediction of the target coordinates. To this end, DiMo-GUI introduces a dynamic zooming mechanism that enables efficient and focused localization. Specifically, the original high-resolution image is first passed to the model for an initial prediction. Based on the returned coordinates, a bounding box is cropped using the center point and a scaling factor of half the original image size. This cropped region is then used as input for the next round of inference. Iterative zooming in allows the model to capture finer details of the target element, making it easier to recognize. At the same time, it significantly reduces redundant regions in the image, thereby increasing the signal-to-noise ratio. This helps the model receive less visual interference and focus more effectively on identifying the target

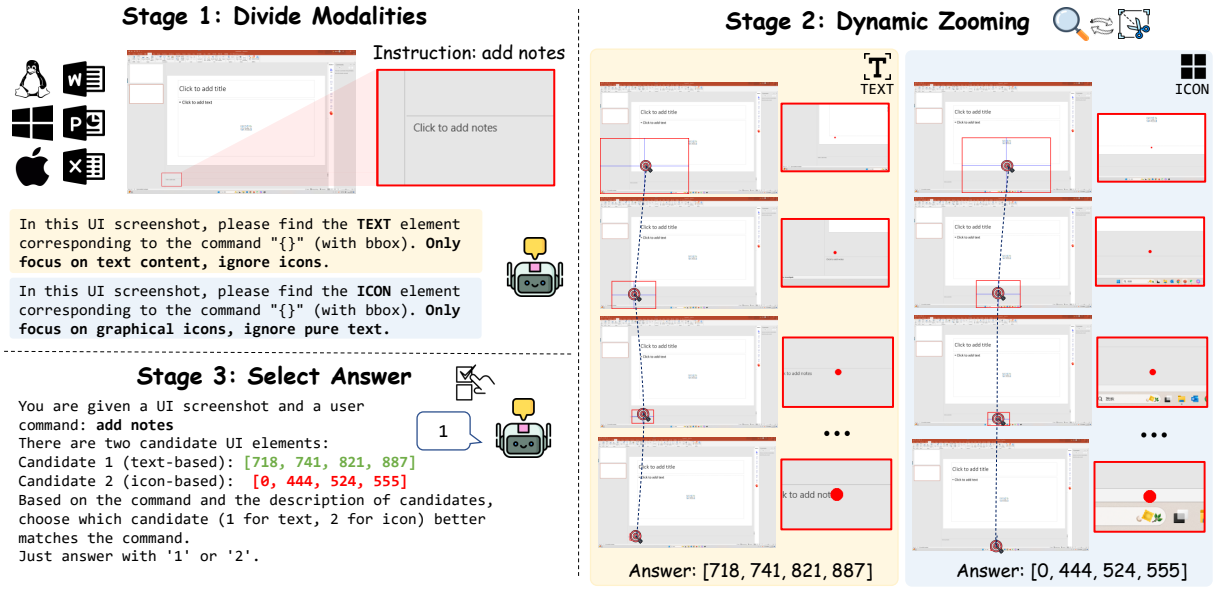

Figure 3: **Processing pipeline of DiMo-GUI.** DiMo-GUI decomposes the grounding process into three steps: (1) Divide Modalities: It processes textual and icon elements in the screenshot separately to prevent interference between the two modalities. (2) Dynamic Zooming: Based on an initial prediction, the model centers on the returned coordinates and crops a region half the size of the original image for more precise localization. (3) Decision Making: By analyzing the instruction along with the screenshot, the model determines whether the text-based or icon-based candidate is more likely to be the correct answer.

element. As the iterations proceed, the model's attention becomes increasingly concentrated, ultimately enabling accurate target localization with minimal computational overhead.

The number of iterations in the zooming process plays a critical role in determining the final grounding performance. Since different GUI screenshots and user instructions vary in complexity, it is evident that a fixed number of iterations is not optimal for all cases. To address this, we introduce a dynamic iteration mechanism that allows the model to autonomously decide whether to DiMo-GUI early during the progressive narrowing process. This approach not only reduces unnecessary iterations and improves inference efficiency but also prevents the model from "overthinking"—i.e., drifting into incorrect regions after having already located the correct target. Specifically, the method determines whether to continue zooming based on the spatial distance between the inference results before and after zooming. If the spatial distance between the predicted coordinates is smaller than one-sixth of the diagonal length of the pre-zoom image, it indicates that the target region has been localized with sufficient precision. In this case, further zooming is DiMo-GUIped, and the final coordinates are returned as the result. The above process is described as `DiMo-GUICondition`(C_t, t) in the algorithm 1, which decides whether to DiMo-GUI dynamic zooming in the t iteration based on the predicted coordinates C_t. Additionally, to prevent excessive zooming, we set an upper limit *max_iters* of seven zooming iterations.

3.2 Modality Decoupling Strategy

Another major challenge in GUI grounding lies in the uneven performance across different UI modalities, particularly between text-based and icon-based elements. Across multiple benchmarks, existing models consistently perform much better on text than on icons. This imbalance stems from two main issues: first, models often lack the ability to effectively recognize and understand icons, making it difficult to correctly associate them with the given instruction; second, models tend to overrely on textual information due to their stronger language processing capabilities, often focusing on related text even when it is not the correct target. To address this issue, we propose a Modality Decoupling Strategy based on a divide-and-conquer paradigm, which explicitly separates the handling of text and icons to reduce cross-modality interference and improve grounding reliability across both modalities.

Specifically, we perform two separate grounding passes over the image: one focusing exclusively on text elements and the other on icon elements. Each pass leverages the proposed dynamic

Grounding Model	Development			Creative			CAD			Scientific			Office			OS			Avg		
	text	icon	avg	text	icon	avg	text	icon	avg	text	icon	avg	text	icon	avg	text	icon	avg	text	icon	avg
QwenVL-7B (Bai et al., 2023)	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.7	0.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.1	0.0	0.1
GPT-4o (OpenAI, 2023)	1.3	0.0	0.7	1.0	0.0	0.6	2.0	0.0	1.5	2.1	0.0	1.2	1.1	0.0	0.6	0.0	0.0	0.0	1.3	0.0	0.8
SeeClick (Cheng et al., 2024)	0.6	0.0	0.3	1.0	0.0	0.6	2.5	0.0	1.9	3.5	0.0	2.0	1.1	0.0	0.5	2.8	0.0	1.5	1.8	0.0	1.1
Qwen2-VL-7B (Wang et al., 2024b)	2.6	0.0	1.3	1.5	0.0	0.9	0.5	0.0	0.4	6.3	0.0	3.5	3.4	1.9	3.0	0.9	0.0	0.5	2.5	0.2	1.6
ShowUI-2B (Lin et al., 2024)	16.9	1.4	9.4	9.1	0.0	5.3	2.5	0.0	1.9	13.2	7.3	10.6	15.3	7.5	13.5	10.3	2.2	6.6	10.8	2.6	7.7
CogAgent-18B (Hong et al., 2024)	14.9	0.7	8.0	9.6	0.0	5.6	7.1	3.1	6.1	22.2	1.8	13.4	13.0	0.0	6.5	5.6	0.0	3.1	12.0	0.8	7.7
Aria-UI (Yang et al., 2024b)	16.2	0.0	8.4	23.7	2.1	14.7	7.6	1.6	6.1	27.1	6.4	18.1	20.3	1.9	16.1	4.7	0.0	2.6	17.1	2.0	11.3
Claude Comp.Use (Hu et al., 2024)	22.0	3.9	12.6	25.9	3.4	16.8	14.5	3.7	11.9	33.9	15.8	25.8	30.1	16.3	26.2	11.0	4.5	8.1	23.4	7.1	17.1
UI-TARS-7B (Qin et al., 2025)	58.4	12.4	36.1	50.0	9.1	32.8	20.8	9.4	18.0	63.9	31.8	50.0	63.3	20.8	53.5	30.8	16.9	24.5	47.8	16.2	35.7
UI-TARS-72B(Qin et al., 2025)	63.0	17.3	40.8	57.1	15.4	39.6	18.8	12.5	17.2	64.6	20.9	45.7	63.3	26.4	54.8	42.1	15.7	30.1	50.9	17.5	38.1
OS-Atlas-4B (Wu et al., 2024)	7.1	0.0	3.7	3.0	1.4	2.3	2.0	0.0	1.5	9.0	**5.5**	7.5	5.1	3.8	4.4	5.6	0.0	3.1	5.0	1.7	3.7
+ *DiMo-GUI*	**13.6**	**1.4**	**7.7**	**9.6**	**2.8**	**6.7**	**4.1**	**4.7**	**4.2**	**30.6**	4.5	**19.3**	**24.3**	**15.1**	**22.2**	**7.5**	**2.2**	**5.1**	**14.6**	**4.0**	**10.6**
Δ	6.5	1.4	4.0	6.6	1.4	4.4	2.1	4.7	2.7	21.6	1.0	11.8	19.2	11.3	17.8	1.9	2.2	2.0	9.6	2.3	6.9
OS-Atlas-7B (Wu et al., 2024)	33.1	1.4	17.7	28.8	2.8	17.9	12.2	4.7	10.3	37.5	7.3	24.4	33.9	5.7	27.4	27.1	4.5	16.8	28.1	4.0	18.9
+ *DiMo-GUI*	**66.9**	**21.4**	**44.8**	**60.6**	**21.7**	**44.3**	**50.3**	**14.1**	**41.4**	**68.1**	**21.8**	**48.0**	**80.8**	**52.8**	**74.3**	**69.2**	**28.1**	**50.5**	**65.2**	**24.5**	**49.7**
Δ	33.8	20.0	27.1	31.8	18.9	26.4	38.1	9.4	31.1	30.6	14.5	23.6	46.9	47.1	46.9	42.1	23.6	33.7	37.1	20.5	30.8
UGround-7B (Gou et al., 2024)	26.6	2.1	14.7	27.3	2.8	17.0	14.2	1.6	11.1	31.9	2.7	19.3	31.6	11.3	27.9	17.8	0.0	9.7	25.0	2.8	16.5
+ *DiMo-GUI*	**44.2**	**6.2**	**25.8**	**39.9**	**7.7**	**26.4**	**17.3**	**3.1**	**13.8**	**50.7**	**8.2**	**32.3**	**46.9**	**15.1**	**39.6**	**32.7**	**10.1**	**22.4**	**38.1**	**7.9**	**26.6**
Δ	17.6	4.1	11.1	12.6	4.9	9.4	3.1	1.5	2.7	18.8	5.5	13.0	15.3	3.8	11.7	14.9	10.1	12.7	13.1	5.1	10.1
UGround-V1-7B (Gou et al., 2024)	51.9	3.4	28.4	48.0	9.1	31.7	20.0	1.6	15.3	57.6	16.4	39.8	61.6	13.2	50.4	37.4	7.9	25.0	45.6	8.4	31.4
+ *DiMo-GUI*	**57.8**	**21.4**	**40.1**	**60.1**	**18.1**	**42.5**	**45.7**	**18.8**	**39.1**	**75.7**	**28.2**	**55.1**	**79.7**	**37.7**	**70.0**	**51.4**	**30.3**	**41.8**	**61.7**	**24.3**	**47.4**
Δ	5.9	18.0	11.7	12.1	9.0	10.8	25.7	17.2	23.8	18.1	11.8	15.3	18.1	24.5	19.6	14.0	22.4	16.8	16.1	15.9	16.0

Table 1: **Comparison of various models on ScreenSpot-Pro.** Without requiring any additional training or external data, DiMo-GUI significantly boosts the grounding performance of existing models. It nearly doubles the performance metrics of OS-ATLAS-7B and UGroundV1-7B on the ScreenSpot-Pro benchmark, with substantial improvements observed across all subsets.

zooming mechanism to progressively refine the target location within its respective modality. After obtaining two candidate coordinate, C_{text} and C_{icon} from each modality, we feed them back into the model alongside the original instruction and full-resolution image. The model then evaluates both candidates and determines which coordinate is more likely to correspond to the correct target C^*, enabling more balanced and reliable grounding across modalities.

4 Experiments

We conducted evaluations of the DiMo-GUI framework on the most recent ScreenSpot-Pro (Li et al., 2025) and ScreenSpot (Cheng et al., 2024) benchmark datasets, and the results demonstrate its superior grounding performance compared to existing approaches.

4.1 Experimental Setup

Benchmarks and Models To thoroughly assess the grounding capabilities of DiMo-GUI, we conduct extensive experiments on two GUI grounding benchmarks: ScreenSpot (Cheng et al., 2024) and ScreenSpot-Pro (Li et al., 2025). ScreenSpot comprises 1,272 samples spanning mobile, desktop, and web platforms, emphasizing common interface scenarios and element types. However, due to its limited ability to represent professional software environments, ScreenSpot-Pro was introduced, featuring 23 professional applications with high-resolution interfaces and complex layouts.

On the two latest datasets mentioned above, we select the most recently reported state-of-the-art GUI agents as baseline models, *i.e.*, OS-Atlas (Wu et al., 2024) and UGround-V1 (Gou et al., 2024). OS-Atlas is a foundational action model that leverages a multi-platform GUI grounding dataset and addresses action naming conflicts during training to enhance performance across desktop, mobile, and web platforms for GUI agent development. UGround-V1 is a universal visual grounding model for GUI agents, trained on the largest dataset of 10M GUI elements and 1.3M screenshots, utilizing web-based synthetic data and a slight adaptation of the LLaVA architecture to accurately map referring expressions to pixel-level coordinates across diverse platforms. We then apply our DiMo-GUI framework to these models to evaluate its effectiveness in enhancing the performance of GUI agent systems.

4.2 Evaluation on Grounding Ability

We evaluate the effectiveness of the DiMo-GUI framework on the latest ScreenSpot-Pro dataset. As shown in Tab. 1, introducing the DiMo-GUI framework leads to significant performance breakthroughs for both OS-Atlas-7B and UGround-V1-7B, with OS-Atlas-7B achieving more than twice the performance of its original version. After integrating the framework, all subsets show notice-

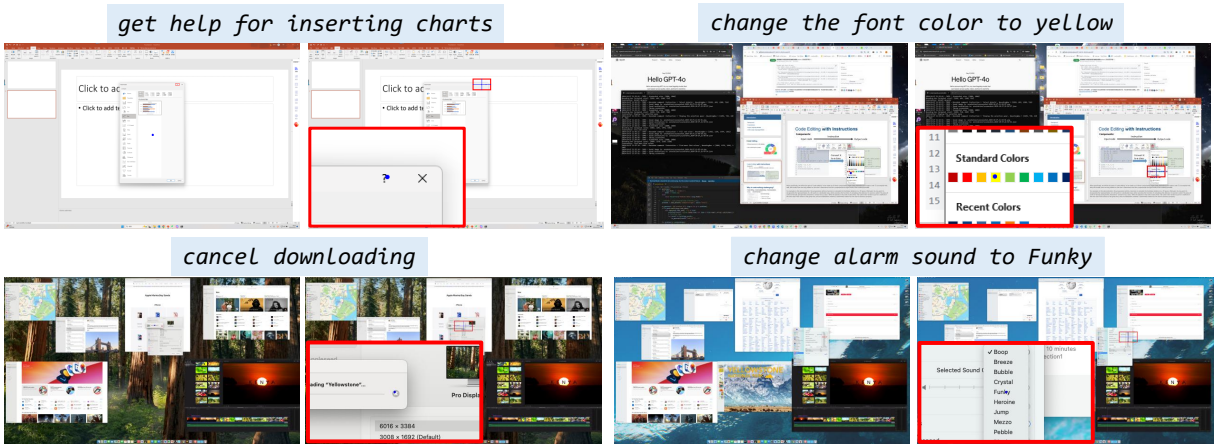

Figure 4: **Quantitative results on ScreenSpot-Pro.** On the left is the original model's prediction, where the red box represents the ground truth and the blue dot indicates the predicted coordinates. On the right is the result after integrating DiMo-GUI, where the model is able to localize more accurately according to the instruction.

able performance improvements, demonstrating that this training-free framework delivers surprisingly strong gains in GUI grounding with minimal cost. The qualitative results further demonstrate the effectiveness of the DiMo-GUI framework. When integrated with OS-Atlas-7B and UGround-V1-7B, we observe that in the early iterations, the models often fail to return accurate coordinates—primarily due to the overwhelming contextual redundancy caused by high-resolution input. However, after several rounds of iterative zooming, the models exhibit a significantly increased likelihood of pinpointing accurate coordinates within specific regions, indicating that DiMo-GUI effectively guides the model's attention to more relevant visual cues.

In addition, we conduct evaluations on the ScreenSpot dataset by integrating the DiMo-GUI framework into OS-Atlas-7B and UGround-V1-7B. As shown in Tab. 2, both models exhibit notable performance improvements, further validating the strong generalizability of this plug-and-play framework. Despite its minimal computational cost, DiMo-GUI consistently enhances grounding performance across diverse task scenarios.

4.3 Analysis

In this section, we analyze the experimental results presented above to investigate the key factors that influence GUI grounding performance. By examining the strengths and weaknesses of different models across various tasks, we aim to identify the main challenges and provide insights into how future research in this field can further improve grounding accuracy and generalization. Overall, the performance of current GUI grounding models is mainly affected by two key factors: ultra-high resolution of GUI screenshots and limited visual processing ability of VLMs.

Ultra-high resolution of GUI screenshots High resolution has always been a critical issue in visual tasks. Almost all visual tasks experience a decline in performance as resolution increases, as higher resolution brings in more redundant information, making the task more challenging. GUI grounding is no exception, especially since the UI elements that need to be localized are often small. As shown in Figure 1, performance in GUI grounding significantly drops as the resolution increases. An intuitive solution to this issue is zooming in, which is the dynamic zooming approach proposed in this paper. However, it can be observed that as the resolution of the screenshots increases, the probability of the model making errors in the first iteration also increases, which inevitably leads to failure in subsequent operations. On the contrary, blindly enlarging the image can also introduce negative effects—for instance, excessive magnification may lead to a loss of global information. Determining the appropriate degree of magnification plays a crucial role in the task of GUI grounding, making a dynamic zooming strategy essential.

Limited visual processing ability of VLMs Another reason for the poor performance of GUI grounding is the weak ability of grounding models to process visual information. Most current GUI agents and grounding models are based on existing multimodal large models, and a common issue with MLLMs is that their ability to process visual information is weaker than their ability to handle text.

GUI Agent MLLMs	Mobile		Desktop		Web		Average
	Text	Icon/Widget	Text	Icon/Widget	Text	Icon/Widget	
InternVL-2-4B (Chen et al., 2024)	9.2	4.8	4.6	4.3	0.9	0.1	4.3
Fuyu (Bavishi et al., 2023)	41.0	1.3	33.0	3.6	33.9	4.4	19.5
Qwen2-VL-7B (Wang et al., 2024b)	61.3	39.3	52.0	45.0	33.0	21.8	42.9
CogAgent (Hong et al., 2024)	67.0	24.0	74.2	20.0	70.4	28.6	47.4
SeeClick (Cheng et al., 2024)	78.0	52.0	72.2	30.0	55.7	32.5	53.4
OS-Atlas-4B (Wu et al., 2024)	85.7	58.5	72.2	45.7	82.6	63.1	70.1
UGround-7B (Gou et al., 2024)	82.8	60.3	82.5	63.6	80.4	70.4	73.3
OS-Atlas-7B (Wu et al., 2024)	93.0	72.9	91.8	62.7	**90.9**	74.3	82.5
+DiMo-GUI	**96.2** ↑3.2	**73.5** ↑0.6	**96.4** ↑4.6	**75.1** ↑12.4	89.7 ↓1.2	**75.4** ↑1.1	**85.7** ↑3.2
UGround-V1-7B (Gou et al., 2024)	**95.0**	83.3	**95.0**	77.8	92.1	77.2	87.6
+DiMo-GUI	94.8 ↓0.2	**85.3** ↑2.0	94.3 ↓0.7	**82.1** ↑4.3	**93.2** ↑1.1	**80.3** ↑3.1	**89.2** ↑1.6

Table 2: **GUI Grounding Results of different GUI Agents on ScreenSpot-v2**. Even though most models already achieve high quantitative scores on this dataset, introducing DiMo-GUI still leads to noticeable performance improvements across the vast majority of subsets.

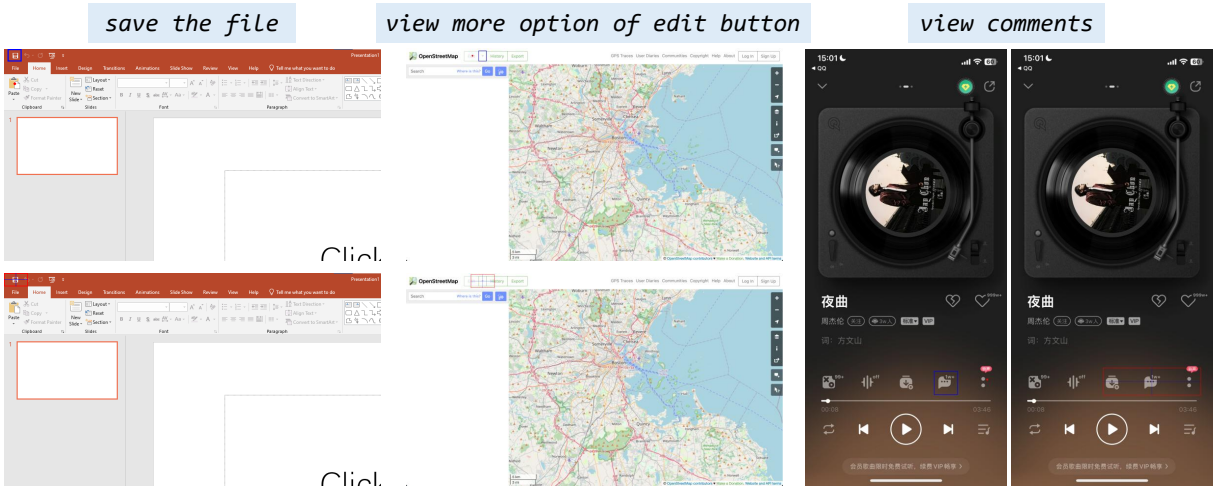

Figure 5: **Quantitative results on ScreenSpot-v2.** On the Screenspot benchmark, which features relatively low resolution and simple scenes, DiMo-GUI also enhances the model's localization capabilities.

This causes the models to be more inclined to trust textual information, a phenomenon known as hallucinations in MLLMs. Since locating, recognizing, and understanding icons is much more difficult than processing text, GUI agents tend to rely more on textual information during the grounding process. The direct consequence is that if a screenshot contains text related to the instruction, or even the same text, GUI agents will almost completely abandon the search for icons and instead use the text as the answer, even though it may not be helpful. The modality decoupling approach we propose effectively addresses this issue by allowing the model to better consider both text and icon modalities, which helps mitigate the drawbacks of the model's weaker ability to process visual information.

As illustrated in the specific example in Fig. 6, when the user instruction includes the word "edit," the agent tends to focus on elements related to editing during the search process. In this case, there happens to be a text element labeled "Edit" in the target region, which conveys a clearer semantic meaning compared to the adjacent icon. Consequently, the agent model is more likely to rely on this text element, as it is not only easier to recognize and understand but also highly relevant to the instruction. However, this text element does not actually fulfill the intended function of the instruction. Its seemingly clear semantics, in this context, become a source of distraction. When we modify the prompt to explicitly direct the agent to focus only on icon elements while ignoring text elements, the model DiMo-GUIs selecting the "Edit" text and instead searches for the appropriate icon. Interestingly, the "Edit" text then serves as valuable contextual information that aids the model in locating the target icon—transforming from a source of distraction into a helpful cue.

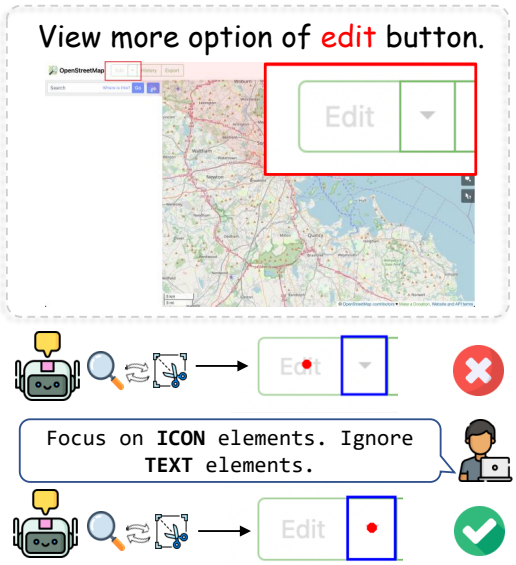

Figure 6: **Case Study.** GUI agents often mistake instruction-related text in the image as targets. Using a divide-and-conquer approach with explicit modality helps the agent locate the target accurately.

Table 3: Ablation on the number of iterative zooming steps. Performance improves with more iterations, but plateaus after 3 steps.

max_iter	0	1	2	3	4	5
acc (%)	18.4	18.7	40.2	46.7	48.8	**48.9**

4.4 Ablation Study

Ablation on Dynamic Zooming To validate the effectiveness of the proposed dynamic zooming strategy, we compare DiMo-GUI with two baselines: (1) a no-zooming baseline where the model directly predicts coordinates from the original screenshot without any refinement, and (2) a single-pass static zooming variant that only zooms into the region of interest once based on the initial prediction. As shown in Tab. 3, the grounding performance first improves and then declines with the monotonic increase in iterations.This aligns with intuition: in early stages, more zoom-in operations help the model focus on target regions by filtering out irrelevant details. However, excessive zooming can remove important context, hindering accurate grounding. Our proposed dynamic iterative zooming approach significantly improves grounding accuracy over both baselines, which demonstrates the importance of progressively refining the region of interest.

Ablation on Modality Decoupling We also investigate the impact of modality decoupling by comparing the full DiMo-GUI framework with a variant that treats all UI elements uniformly without distinguishing between text and icon modalities. The results in Tab. 4 show that modality-aware processing leads to consistent performance gains. This confirms our hypothesis that different modalities benefit from specialized zooming strategies, and that decoupling helps reduce visual ambiguity, particularly in scenarios where icons are harder to interpret than text.

Table 4: Ablation on Dynamic Grounding and Modalities Dividing.

Method	OS-Atlas-7B
vanilla	18.4
w DG	45.7
w MD	26.1
w DiMo-GUI	**49.7**

5 Conclusion

DiMo-GUI is a training-free, plug-and-play framework designed specifically for the GUI grounding task. It incorporates two key components: dynamic zooming and modality decoupling, which effectively address the challenges of handling high-resolution screenshots and the limited visual understanding capability of existing GUI agents. By progressively refining the focus region and treating text and icon modalities separately, DiMo-GUI significantly boosts grounding performance across various benchmarks and models, offering substantial improvements with minimal computational overhead.

6 Limitations

Currently our model employs a progressive expansion strategy without any error correction or backtracking mechanisms. This can lead to early-stage mistakes that propagate and become irrecoverable. In future work, we plan to incorporate backtracking mechanisms using structures such as trees or graphs, aiming to further improve the accuracy.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3674–3683.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou,

and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond.

Rohan Bavishi, Erich Elsen, Curtis Hawthorne, Maxwell Nye, Augustus Odena, Arushi Somani, and Sagnak Taşırlar. 2023. Introducing our multimodal models.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. 2024. InternVL: Scaling up vision foundation models and aligning for generic visual-linguistic tasks.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. 2024. Seeclick: Harnessing gui grounding for advanced visual gui agents. *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yuning Du, Chenxia Li, Ruoyu Guo, Xiaoting Yin, Weiwei Liu, Jun Zhou, Yifan Bai, Zilin Yu, Yehua Yang, Qingqing Dang, and Haoshuang Wang. 2020. PP-OCR: A practical ultra lightweight ocr system.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. 2024. Navigating the digital world as humans do: Universal visual grounding for gui agents. *International Conference on Learning Representations (ICLR)*.

Wenyi Hong, Weihan 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. 2024. CogAgent: A visual language model for gui agents.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron Courville, Alessandro Sordoni, and Rishabh Agarwal. 2024. V-star: Training verifiers for self-taught reasoners. *arXiv preprint arXiv:2402.06457*.

Siyuan Hu, Mingyu Ouyang, Difei Gao, and Mike Zheng Shou. 2024. The Dawn of GUI Agent: A preliminary case study with claude 3.5 computer use.

Hyunseok Lee, Jeonghoon Kim, Beomjun Kim, Jihoon Tack, Chansong Jo, Jaehong Lee, Cheonbok Park, Sookyo In, Jinwoo Shin, and Kang Min Yoo. 2025a. ReGUIDE: Data efficient gui grounding via spatial reasoning and search. *arXiv preprint arXiv:2505.15259*.

Hyunseok Lee, Seunghyuk Oh, Jaehyung Kim, Jinwoo Shin, and Jihoon Tack. 2025b. Revise: Learning to refine at test-time via intrinsic self-verification. *arXiv preprint arXiv:2502.14565*.

Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. 2025. ScreenSpot-Pro: Gui grounding for professional high-resolution computer use.

Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Weixian Lei, Lijuan Wang, and Mike Zheng Shou. 2024. ShowUI: One vision-language-action model for gui visual agent.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. 2024a. Grounding DINO: Marrying dino with grounded pre-training for open-set object detection.

Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, et al. 2024b. VisualAgentBench: Towards large multimodal models as visual foundation agents. *arXiv preprint arXiv:2408.06327*.

Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. 2024. Omniparser for pure vision based gui agent. *arXiv preprint arXiv:2408.00203*.

Tiange Luo, Lajanugen Logeswaran, Justin Johnson, and Honglak Lee. 2025. Visual test-time scaling for gui agent grounding. *arXiv preprint arXiv:2505.00684*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2022. WebGPT: Browser-assisted question-answering with human feedback.

Anthony Nguyen. 2024. Improved gui grounding via iterative narrowing. *arXiv preprint arXiv:2411.13591*.

OpenAI. 2023. Gpt-4 technical report.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. 2025. UI-TARS: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*.

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

Fei Tang, Yongliang Shen, Hang Zhang, Siqi Chen, Guiyang Hou, Wenqi Zhang, Wenqiao Zhang, Kaitao Song, Weiming Lu, and Yueting Zhuang. 2025. Think Twice, Click Once: Enhancing gui grounding via fast and slow systems. *arXiv preprint arXiv:2503.06470*.

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

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei

Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024b. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution.

Penghao Wu and Saining Xie. 2024. V*: Guided visual search as a core mechanism in multimodal llms. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13084–13094.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, et al. 2024. Os-atlas: A foundation action model for generalist gui agents. *International Conference on Learning Representations (ICLR)*.

Xiaobo Xia and Run Luo. 2025. GUI-R1: A generalist r1-style vision-language action model for gui agents. *arXiv preprint arXiv:2504.10458*.

Ke Yang, Yao Liu, Sapana Chaudhary, Rasool Fakoor, Pratik Chaudhari, George Karypis, and Huzefa Rangwala. 2024a. Agentoccam: A simple yet strong baseline for llm-based web agents. *arXiv preprint arXiv:2410.13825*.

Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. 2024b. Aria-UI: Visual grounding for gui instructions.

Xinbin Yuan, Jian Zhang, Kaixin Li, Zhuoxuan Cai, Lujian Yao, Jie Chen, Enguang Wang, Qibin Hou, Jinwei Chen, Peng-Tao Jiang, et al. 2025. Enhancing visual grounding for gui agents via self-evolutionary reinforcement learning. *arXiv preprint arXiv: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. 2024. UFO: A ui-focused agent for windows os interaction.

Chi Zhang, Zhao Yang, Jiaxuan Liu, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. 2023. AppAgent: Multimodal agents as smartphone users.

Zhuosheng Zhang and Aston Zhang. 2023. You only look at screens: Multimodal chain-of-action agents. *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Yuqi Zhou, Sunhao Dai, Shuai Wang, Kaiwen Zhou, Qinqlin Jia, et al. 2025. GUI-G1: Understanding r1-zero-like training for visual grounding in gui agents. *arXiv preprint arXiv:2505.15810*.