# Improved GUI Grounding via Iterative Narrowing

**Anthony Nguyen**
Algoma University
Ontario, Canada
anguyen@algomau.ca

## Abstract

Graphical User Interface (GUI) grounding plays a crucial role in enhancing the capabilities of Vision-Language Model (VLM) agents. While general VLMs, such as GPT-4V, demonstrate strong performance across various tasks, their proficiency in GUI grounding remains suboptimal. Recent studies have focused on fine-tuning these models specifically for zero-shot GUI grounding, yielding significant improvements over baseline performance. We introduce a visual prompting framework that employs an iterative narrowing mechanism to further improve the performance of both general and fine-tuned models in GUI grounding. For evaluation, we tested our method on a comprehensive benchmark comprising various UI platforms and provided the code to reproduce our results in this GitHub repository.

## 1 Introduction

GUI Grounding is the task of identifying the visual location on an interface image given a natural language query [4, 9]. This task is essential for improving the capabilities of Vision Language Model (VLM) agents, enabling them to better understand and interact with graphical user interfaces [10]. While general VLMs, such as GPT-4V, have demonstrated high performance across a variety of visual-linguistic tasks, their accuracy in GUI grounding tasks remains suboptimal [4, 10].

Existing solutions have attempted to bridge this performance gap by fine-tuning VLMs for GUI grounding, resulting in improvements over baseline models [5, 4, 9, 6]. However, these approaches often come with the need for extensive data and re-training.

Instead of training a model, we propose a visual prompting framework that employs an iterative narrowing mechanism to improve the grounding performance of current VLMs. Our approach treats the model's initial position predictions as an approximation. The framework iteratively refines the prediction by focusing on progressively smaller, cropped regions that center around the previous prediction. This refinement process can be repeated for multiple iterations.

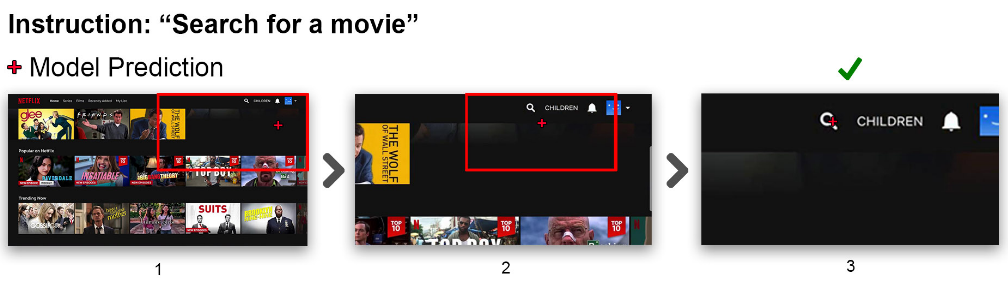

Figure 1: A visual demonstration of our method. On each iteration, the model prediction (red cross) determines the next region (red box) to iterate.

We evaluate the performance of our method on the ScreenSpot [4] benchmark. Our findings demonstrate that this iterative approach leads to substantial performance gains, providing a straightforward yet effective solution for enhancing GUI grounding capabilities.

To encourage reproducibility and facilitate further research, we provide our code at `https://github.com/ant-8/GUI-Grounding-via-Iterative-Narrowing`.

## 2 Related Work

### 2.1 Grounding Models

Cheng et al. [4] trained SeeClick, a GUI-specific grounding model and introduced the ScreenSpot benchmark, showing that grounding pre-training enhances task performance. Gou et al. [5] expanded on this by training a pixel-level model on a dataset of 10 million GUI elements, achieving up to 20% accuracy improvements and demonstrating effective human-like visual grounding. Wu et al. [9] introduced a cross-platform dataset of 13 million GUI elements and trained two models: OS-Atlas-Base-4B and OS-Atlas-Base-7B, improving performance across diverse benchmarks.

### 2.2 GUI Agents

Xie et al. [10] introduced the OSWorld benchmark, designed to evaluate an agent's ability to complete tasks within GUI environments. One of the key categories tested is the screenshot-only scenario, where the model is provided solely with an image of the screen, without any auxiliary information. This scenario emphasizes the importance of strong grounding capabilities, as a common failure case observed by the authors stems from inadequate grounding performance.

Wu et al. [9] expanded on this challenge by integrating their pre-trained grounding model, OS-Atlas-Base-7B, with GPT-4o, demonstrating significant improvements over GPT-4o alone on the OSWorld benchmark. Their results further validate the crucial role of effective grounding in improving task completion in GUI environments.

Hong et al. [6] also addressed GUI grounding by training and evaluating CogAgent on the screenshot-only category of OSWorld. CogAgent was developed to function both as a grounding model as well as a standalone visual agent.

### 2.3 Iterative Visual Reasoning Beyond Convolutions

Chen et al. [1] proposed an iterative visual reasoning framework that extends beyond traditional CNNs by incorporating a local module with spatial memory and a global graph-reasoning module for spatial and semantic interactions. The global module builds a graph connecting knowledge, image regions, and class assignments, enabling iterative updates through cross-feeding between the modules to refine predictions.

### 2.4 Iterative Localization in Proposal-Free Referring Expression Grounding

Sun et al. [7] tackled proposal-free referring expression grounding introducing a similar mechanism called *iterative shrinking*, which is guided by a reinforcement learning agent to localize target objects based on query sentences.

This iterative approach parallels our method, though Sun et al. uses reinforcement learning, while our method relies on VLM predictions to refine the target.

## 3 Methodology

Our method refines the prediction of target coordinates within an input image $\mathcal{I}$, guided by a text query $\mathcal{Q}$, through an iterative process of cropping and coordinate prediction.

Initially, the entire image $\mathcal{I}$ is treated as the cropping window, with dimensions equal to the original image size. At each iteration, the model predicts normalized coordinates $(x_k, y_k) \in [0, 1] \times [0, 1]$, representing a relevant location within the current cropping window. These coordinates are used to update the cropping window for the next iteration, progressively narrowing the region of focus.

The cropping window dimensions are adjusted dynamically based on the image orientation. For landscape-oriented images, the width and height are scaled by factors $\alpha_{\text{landscape}}$ and $\beta_{\text{landscape}}$, respectively, where both scale factors are less than one. Similarly, for portrait-oriented images, the width is scaled by $\alpha_{\text{portrait}}$, which is less aggressive than the landscape scaling, while the height is scaled by $\beta_{\text{portrait}}$.

After $n$ iterations, the final normalized coordinates are transformed into absolute coordinates relative to the original image dimensions. This transformation ensures that the output $(X, Y)$ corresponds directly to a location within the original image.

### 3.1 Pseudocode

The following pseudocode outlines the iterative process of predicting the target location in the image:

```
Algorithm 1 Iterative Cropping and Prediction
Input: Image $\mathcal{I}$, Query $\mathcal{Q}$, Number of iterations $n$
Output: Final coordinates $(X, Y)$ relative to the original image
    1. Initialize cropping window $\mathcal{C}_0$ dimensions $w_0 = W_{\text{orig}}, h_0 = H_{\text{orig}}$
    2. For $k = 1$ to $n$:
        (a) Using VLM: Predict normalized coordinates $(x_k, y_k) \in [0, 1] \times [0, 1]$ relative to $\mathcal{C}_{k-1}$,
            from $\mathcal{Q}$
        (b) If $\mathcal{I}$ is landscape-oriented:
            • $w_k \leftarrow \alpha_{\text{landscape}} \cdot w_{k-1}, \quad h_k \leftarrow \beta_{\text{landscape}} \cdot h_{k-1}$
        (c) Else (portrait-oriented):
            • $w_k \leftarrow \alpha_{\text{portrait}} \cdot w_{k-1}, \quad h_k \leftarrow \beta_{\text{portrait}} \cdot h_{k-1}$
        (d) Define cropping window $\mathcal{C}_k$:
            • Center: $(x_k, y_k)$
            • Dimensions: $(w_k, h_k)$
    3. Convert normalized $(x_n, y_n)$ to coordinates relative to the original image space $(X, Y)$
    4. Return Final coordinates $(X, Y)$
```

## 4 Experiments

### 4.1 Setup

We evaluated our method on the ScreenSpot [4] benchmark, a comprehensive assessment tool for single-step GUI grounding across multiple platforms. The ScreenSpot benchmark is structured into three primary categories: *mobile*, *web*, and *desktop*, each designed to reflect common user interface environments. Furthermore, each category is divided into two subcategories based on the type of target element: *text* or *icon*.

For the entire evaluation, we used $n = 3$ iterations. While this choice provided promising results, we did not conduct in-depth investigations to determine an optimal value for $n$ or how aggressive each crop should be shrunken. The cropping aggressiveness was controlled using scaling factors:

- For landscape-oriented images:
  - Width scaling factor: $\alpha_{\text{landscape}} = 0.5$
  - Height scaling factor: $\beta_{\text{landscape}} = 0.5$
- For portrait-oriented images:
  - Width scaling factor: $\alpha_{\text{portrait}} = \frac{1}{1.2} \approx 0.833$
  - Height scaling factor: $\beta_{\text{portrait}} = 0.5$

Future work may explore the impact of varying $n$ and the values of $\alpha_{\text{landscape}}$, $\beta_{\text{landscape}}$, $\alpha_{\text{portrait}}$, and $\beta_{\text{portrait}}$ on performance and inference time.

## 4.2 Evaluation Results

| **Models** | **Baseline** | **IN** |
|---|---|---|
| InternVL-2-4B | 4.32 | **6.53** |
| Qwen2-VL-7B | 42.89 | **69.1** |
| OS-Atlas-Base-7B | 82.47 | **83.33** |

Table 1: Overall average accuracy (%) comparing baseline against our method (IN) on the ScreenSpot benchmark.

| **Models** | **Text** | | **Icon/Widget** | |
|---|---|---|---|---|
| | Baseline | IN | Baseline | IN |
| InternVL-2-4B | 9.16 | **14.65** | 1.64 | **2.18** |
| Qwen2-VL-7B | 61.34 | **83.52** | 39.29 | **57.21** |
| OS-Atlas-Base-7B | 93.04 | 93.04 | 72.93 | **74.24** |

Table 2: Accuracy scores in the *mobile* category.

| **Models** | **Text** | | **Icon/Widget** | |
|---|---|---|---|---|
| | Baseline | IN | Baseline | IN |
| InternVL-2-4B | 4.64 | **8.76** | **4.29** | 2.14 |
| Qwen2-VL-7B | 52.01 | **84.54** | 44.98 | **60.71** |
| OS-Atlas-Base-7B | 91.75 | **92.27** | 62.86 | **77.14** |

Table 3: Accuracy scores in the *desktop* category.

| **Models** | **Text** | | **Icon/Widget** | |
|---|---|---|---|---|
| | Baseline | IN | Baseline | IN |
| InternVL-2-4B | 0.87 | **5.22** | 0.10 | **2.91** |
| Qwen2-VL-7B | 33.04 | **73.04** | 21.84 | **50.0** |
| OS-Atlas-Base-7B | **90.87** | 86.96 | **74.27** | 72.33 |

Table 4: Accuracy scores in the *web* category.

## 4.3 Weaknesses & Observations

A key weakness of our method is the loss of contextual information as iterations progress, posing challenges for identifying target elements that rely on spatially distant cues. GUI-specific models like OS-Atlas-Base-7B [9] can accurately identify such targets in the baseline, while our method often fails due to limited context (see Figure 2).

Our method improves precision but at the cost of underperforming in context-dependent scenarios. This limitation likely explains the greater gains observed in generalist VLMs (e.g., InternVL-2-4B [3, 2], Qwen2-VL-7B [8]) versus OS-Atlas-Base-7B [9], which already has strong precision abilities from training. Addressing this limitation by maintaining context throughout the iterations could further enhance performance for all models.

## 4.4 Preliminary Solutions and Future Work

To address the loss of contextual information, we conducted preliminary experiments where the model was provided with both the entire screenshot and the current crop during each iteration. This

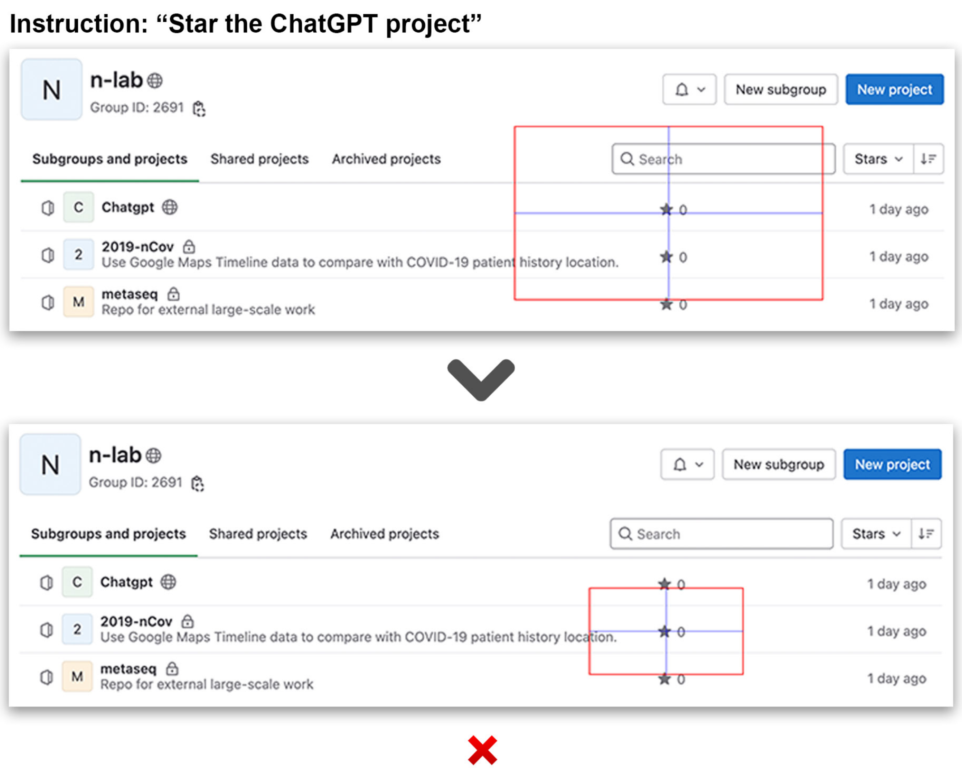

Figure 2: An example of a context-dependent failure case. At iteration $k < n$ (top image), the cropping leads to the loss of crucial contextual information—specifically, the association between each star button and its corresponding project. Thus, resulting in the wrong UI element being selected (bottom image).

approach was inspired by the work of Chen et al. [1], which introduced a mechanism to leverage both local and global modules that iteratively exchange information for enhanced reasoning. Our goal was to enable the model to retain a global understanding while refining its focus through local crops.

However, during these initial experiments, we observed that the Visual Language Model (VLM) frequently confused the local crop with the global context image. This confusion led to incorrect coordinate predictions, as the model sometimes generated coordinates referencing the entire global image rather than accurately focusing on the local context intended for refinement.

We believe that further fine-tuning could help the model more effectively distinguish between global and local contexts, ultimately improving its performance in context-dependent scenarios.

## 5 Conclusion

In this work, we introduced a visual prompting framework designed to perform an iterative narrowing mechanism. As a result, this enhances the GUI grounding capabilities of Vision-Language Models (VLMs). Through iterative refinement of predictions, our method improves accuracy by narrowing the focus on progressively smaller regions, allowing models to more precisely identify visual elements on a graphical user interface. Evaluation on the ScreenSpot [4] benchmark demonstrated that our method improves performance, especially for generalist VLMs like InternVL-2-4B [2, 3] and Qwen2-VL-7B [8], where substantial improvements were observed compared to baselines. However, the method shows limitations when handling spatially distant contextual cues, which affects performance in context-dependent scenarios.

Future work may focus on addressing these contextual limitations by incorporating both global and local context information more effectively. Promising directions may include refining the model's ability to differentiate between local crops and the entire image. There is potential to further push the boundaries of VLM precision in GUI grounding tasks, contributing to the development of more effective visual agents.

## References

[1] Xinlei Chen, Li-Jia Li, Li Fei-Fei, and Abhinav Gupta. Iterative visual reasoning beyond convolutions, 2018. URL https://arxiv.org/abs/1803.11189.

[2] 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. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. *arXiv preprint arXiv:2312.14238*, 2023.

[3] Zhe Chen, Weiyun Wang, Hao Tian, Shenglong Ye, Zhangwei Gao, Erfei Cui, Wenwen Tong, Kongzhi Hu, Jiapeng Luo, Zheng Ma, et al. How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites. *arXiv preprint arXiv:2404.16821*, 2024.

[4] Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. SeeClick: Harnessing GUI grounding for advanced visual GUI agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9313–9332, Bangkok, Thailand, August 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.acl-long.505.

[5] 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. *arXiv preprint arXiv:2410.05243*, 2024. URL https://arxiv.org/abs/2410.05243.

[6] 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. Cogagent: A visual language model for gui agents, 2023. URL https://arxiv.org/abs/2312.08914.

[7] Mingjie Sun, Jimin Xiao, and Eng Gee Lim. Iterative shrinking for referring expression grounding using deep reinforcement learning, 2021. URL https://arxiv.org/abs/2103.05187.

[8] 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. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[9] 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 preprint arXiv:2410.23218*, 2024.

[10] Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments, 2024.