DeCoPatch: Revealing Causal Latent Subspaces in Vision-Language Models for GUI Grounding

Yongkang Zhang[1*†], Linjia Kang[1*], Zhimin Wang[1,2*], Duo Wu[1,2], and Zhi Wang[1✉]

[1] Shenzhen International Graduate School, Tsinghua University
[2] Pengcheng Laboratory
yongkangzhang273@gmail.com
{klj25,wangzhim25,wu-d24}@mails.tsinghua.edu.cn
wangzhi@sz.tsinghua.edu.cn
Github: https://github.com/calmee4/decopatch

Abstract. Vision–Language Models (VLMs) have recently emerged as a powerful paradigm for autonomously perceiving and interacting with complex and dynamic Graphical User Interface (GUI) environments. However, their interaction capability remains fundamentally limited by insufficient precision in spatial grounding. Intriguingly, simply overlaying visual markers on screenshots has been extensively shown to substantially enhance the grounding performance of models. Despite its empirical effectiveness, the underlying mechanism—specifically, why these explicit visual cues can influence the model's internal spatial understanding—remains largely unexplained. We find that the causal influence of visual markers on grounding performance is mediated by low-rank latent subspaces associated with specific neurons in the late decoder layers. Based on this key insight, we propose *DeCoPatch*, a novel decode-time causal intervention method that precisely modulates the activations of key neurons relevant to GUI grounding. *DeCoPatch* follows a two-stage design: it first identifies top-k critical neurons during the pre-fill stage, effectively isolating a low-rank latent subspace, and then reinforces these neurons during the decoding stage to causally steer the model's spatial grounding behavior. Extensive experiments on four challenging benchmarks demonstrate that *DeCoPatch* consistently enhances the GUI grounding performance of diverse VLMs with almost no additional computational overhead.

Keywords: Vision-Language Models · Causal Intervention · GUI Grounding

---

* Equal contribution.
† Yongkang's work was done during his internship at Tsinghua University.
✉ Corresponding author.

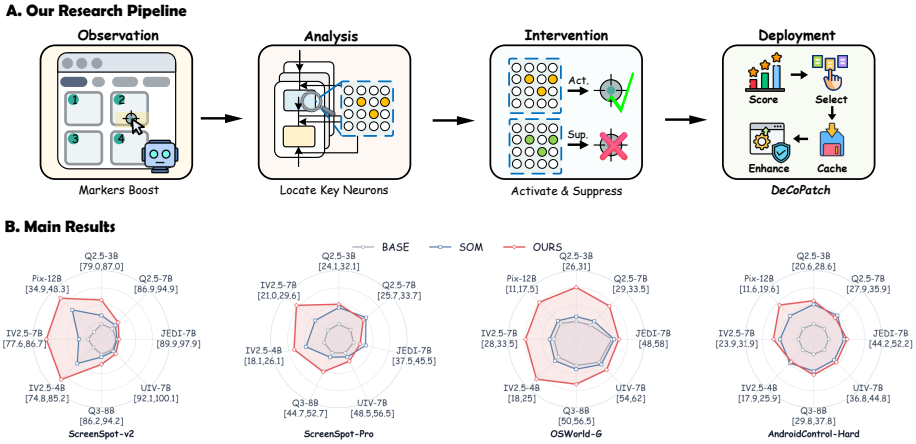

Fig. 1: We introduce *DeCoPatch*, a decode-time causal intervention method that modulates the activations of key neurons for GUI grounding. We first observe that visual markers significantly improve grounding performance. We then conduct an in-depth analysis of the model's internal mechanisms and hypothesize that this effect is mediated by a low-rank subspace formed by specific neurons. By applying causal interventions to these key neurons, we validate this hypothesis and subsequently develop the *DeCoPatch* method. Extensive experiments on four challenging benchmarks show that *DeCoPatch* consistently improves GUI grounding performance across diverse VLMs.

1 Introduction

Vision-Language Models (VLMs) [2, 3, 6, 16, 22] have enabled agents to autonomously perceive and interact with Graphical User Interface (GUI) environments, offering a promising paradigm for intelligent digital automation [13,17,18, 38]. Nevertheless, existing VLMs still struggle with precise grounding in highly dynamic GUI environments, which limits the performance of agents on complex GUI tasks [5, 9, 10, 14, 23, 35]. These limitations largely arise from insufficient spatial understanding and reasoning of VLMs. Since GUI tasks often involve multi-step interactions, even a minor grounding error (*e.g.*, clicking a neighboring button) can place the model in an incorrect interface state and ultimately lead to total task failure [7,41]. To mitigate this issue, recent works [21,25,26,28,34] have introduced visual markers (*e.g.*, SoM [32]) as explicit spatial cues, which have been widely demonstrated to be a simple and effective approach [30, 40, 42, 44]. By explicitly highlighting target regions on screenshots, these marker-based cues help VLMs more reliably locate interface elements, thereby significantly enhancing the models' capability to perform complex GUI interactions.

However, these works largely treat visual markers as an empirical trick. It remains unclear how such explicit spatial cues influence VLMs' latent representations in complex GUI scenarios and improve their ability to accurately localize interface elements. A deeper understanding of this mechanism is not only crucial for further improving VLMs' performance in GUI grounding under complex

layouts and multi-step interactions, but also offers the possibility of more precise causal interventions on the model's internal reasoning processes.

In this work, we conduct a systematic investigation from an interpretability perspective, as depicted in Figure 1. We find that the causal influence of visual markers is not diffusely distributed throughout the model, but instead is highly concentrated within a low-rank latent subspace in the late decoder layers.

Given this key insight, we propose *DeCoPatch*, a novel decode-time causal intervention that precisely modulates the activations of key internal neurons. The computation of *DeCoPatch* is divided into two stages: (1) *prefill* stage, where it identifies and caches a top-k subset of critical neurons within the late feed-forward layers and generates a mask. Neurons are selected based on our designed vision–text ratio scores, ensuring that only those most relevant for spatial grounding are targeted. By focusing on this top-k subset, the intervention is effectively confined to a low-rank latent subspace. (2) *decoding* stage, where the neuron mask is applied to each token for activation patching, allowing the model to selectively reinforce key neural pathways and improve precise localization of interface elements.

This targeted causal intervention enables precise modulation of the model's internal visual reasoning.

Contributions:

- We provide a deep mechanistic analysis of visual-marker gains for GUI grounding, showing that their causal influence is concentrated in a low-rank latent subspace within the late decoder layers of VLMs.
- We propose *DeCoPatch*, a novel causal intervention method that precisely modulates key neurons through activation patching at inference time, significantly enhancing performance on GUI grounding tasks.
- We conduct a comprehensive evaluation on four challenging benchmarks, demonstrating that *DeCoPatch* robustly enhances the GUI grounding capabilities of various VLMs with negligible computational overhead.

2 Related Work

GUI Grounding. Recent GUI agents emphasize grounding as the capability to map natural language instructions to actionable screen locations [36]. SeeClick [7] improves GUI grounding through grounding-centric pretraining on automatically curated screenshot data, leading to stronger downstream GUI agent performance. UGround [14] trains a universal grounding model on large-scale synthetic GUI data, significantly boosting cross-platform performance. OSWorld-G [31] builds a large-scale multi-task dataset to improve generalization to complex layouts and diverse interaction scenarios. SEEACT [43] employs adaptive multimodal inference strategies to improve action grounding robustness. RegionFocus [24] applies visual test-time scaling to zoom into relevant regions and reduce background noise. GUI-RC [9] exploits spatial consistency across multiple sampled predictions to identify consensus regions without additional training. GUI-Actor [28] introduces a coordinate-free architecture with an

attention-based action head that aligns visual tokens to target regions, improving spatial-semantic alignment and generalization. Despite their effectiveness, these approaches largely overlook interpretable analysis of the internal mechanisms that govern grounding behavior in VLMs. Our work explores this direction by leveraging mechanism-level insights to enable improvements in grounding performance.

Causal Intervention. Research on mechanistic interpretability enables precise causal interventions by identifying functional neurons, which is increasingly critical for understanding the complex internal states of models. Dai *et al*. [8] introduce the concept of knowledge neurons and a method to locate and edit specific knowledge within the pretrained model without fine-tuning. AlphaEdit [11] introduces null-space projection to locating-then-editing methods, ensuring parameter updates don't disrupt preserved knowledge during sequential model editing. Wang *et al*. [27] identify task-specific skill neurons in pre-trained Transformers, proving they are primarily developed during pre-training and are essential for model performance. Zhang *et al*. [39] systematically evaluate activation patching methodologies, and provide evidence-based best practices for more reliable mechanistic interpretability. H-Neurons [12] localizes a sparse subset of neurons originating in pre-training that causally trigger hallucinations and over-compliance in models. Despite these advances, our work is the first to leverage causal intervention techniques to enhance the grounding performance of VLMs.

3 Motivation & Mechanistic Analysis

As described in Sec. 1, our goal is to understand why visual markers improve GUI grounding, and how to reproduce these gains without rendering physical overlays. We conduct an in-depth analysis on ScreenSpot-v2 [29] using three representative visual markers—Grid [21], Intersections [21], and SoM [32]—across multiple VLMs, including Qwen2.5-VL-7B [4], Qwen3-VL-8B [3], and UI-Venus-Ground-7B [15].

We organize our analysis around three key questions: (Q1) Are there critical neurons within the model that are highly correlated with the performance gains brought by visual markers? (Q2) How can we apply effective causal interventions to these critical neurons in order to improve grounding performance? (Q3) Can we distill this mechanism into a simple online intervention method that operates without explicit physical markers?

3.1 Localization of Marker-Sensitive Neurons (Q1)

Motivation. To localize marker-induced changes within the model, we focus on the implicit representations in the decoder. This choice is motivated by the common view that, from feature extraction to text generation, the outputs of VLMs typically rely on the late layers [33, 37].

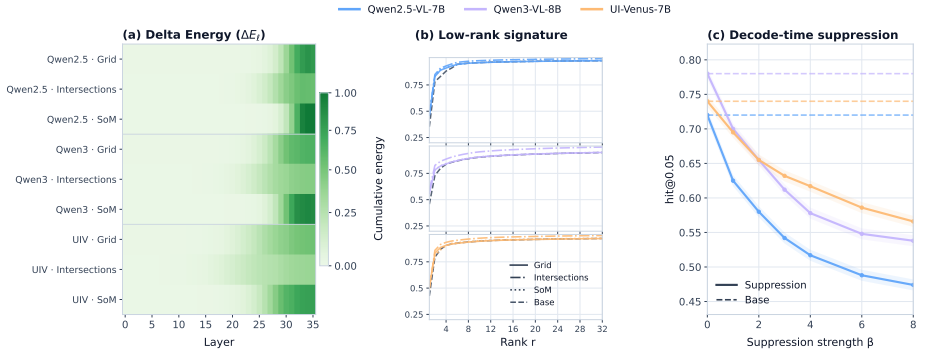

Fig. 2: Results of mechanistic analysis. (a) Layerwise delta-energy heatmaps for three marker variants, with values normalized to $[0, 1]$ for comparability, show concentrated effects in late decoder layers (Q1). (b) The delta-energy matrix ΔH_ℓ exhibits a consistent low-rank signature, with most cumulative energy captured by early ranks (Q2). (c) Increasing decode-time suppression strength β progressively reduces marker-induced gains toward the base reference. Curves are shown up to $\beta = 8$, where suppression behavior is already near saturation (Q2).

Experimental Setup. We denote the hidden state of token t of decoder layer l as $h_{\ell,t} \in \mathbb{R}^d$. Let $t^\star$ denote the decode-time decision token (i.e., the position where the action is executed), and let $h_{\ell,t^\star}$ be the marker-induced hidden state at layer ℓ. For each example i, we define the decision-token difference: $\Delta h_\ell^{(i)} = h_{\ell,t^\star}^{(i,\text{marker})} - h_{\ell,t^\star}^{(i,\text{base})}$. For each layer ℓ, we stack these differences across N examples into $\Delta H_\ell \in \mathbb{R}^{N\times d}$ and compute its singular value decomposition $\Delta H_\ell = U_\ell \Sigma_\ell V_\ell^\top$. We define per-layer delta energy to summarize the magnitude of marker-induced changes (Fig. 2(a)):

$$\Delta E_\ell = \frac{1}{N}\|\Delta H_\ell\|_F^2. \tag{1}$$

Insight 1: Visual markers primarily affect late decoder layers. We observe that the delta energy ΔE_ℓ induced by the presence or absence of markers is concentrated in the layer 25-35 of the decoder, as shown in Fig. 2(a). This pattern suggests that the impact from visual markers arises primarily in the late layers of decoder, where interventions on corresponding neurons may steer decision-related representations. These observations form the foundation of *DeCoPatch*, which achieves precise control through patching for late-layer activations.

3.2 Causal Intervention for Grounding Improvement (Q2)

Motivation. Visual markers introduce a structured, localized feature perturbation rather than a global shift in the input distribution. We therefore hypothesize that the marker-induced changes in hidden representations concentrate within a

small set of key subspaces responsible for grounding improvement, manifesting as a low-rank structure.

Experimental Setup. To quantify the low-rank structure of the marker-induced representation changes, we compute the cumulative spectral energy for rank r with the following equation:

$$C_\ell(r) = \frac{\sum_{j=1}^{r} \Sigma_\ell[j,j]^2}{\sum_j \Sigma_\ell[j,j]^2}. \tag{2}$$

Furthermore, we intervene only at the decision token during decoding and restrict interventions to the last-l decoder layers $\mathcal{L}$ selected by ΔE_ℓ. For a chosen rank r, the top-r right singular vectors $V_{\ell,r} \in \mathbb{R}^{d\times r}$ define a marker-sensitive latent subspace with projection matrix:

$$P_\ell = V_{\ell,r} V_{\ell,r}^\top, \tag{3}$$

For each $\ell \in \mathcal{L}$, we apply a specific suppression intervention:

$$h_{\ell,t^\star}^{(i,\text{marker})} \leftarrow h_{\ell,t^\star}^{(i,\text{marker})} - \beta\, P_\ell \Delta h_\ell^{(i)}, \tag{4}$$

where $\beta \geq 0$ controls the suppression strength. We use the hit@0.05 metric defined as Eq. 5 to evaluate the model's GUI grounding performance. Given a screenshot and a textual instruction as input, the model predicts the coordinate of the target UI element $\hat{p} \in [0,1]^2$. Since different benchmarks provide annotations in varying formats, we uniformly normalize all bounding boxes to the unit square. Let b denote the normalized bounding box of the target element. We define $d(\hat{p}, b)$ as the Euclidean distance from $\hat{p}$ to the central point within b.

$$\text{hit@}\delta = \mathbb{E}[\mathbb{I}(d(\hat{p}, b) \leq \delta)], \qquad \delta \geq 0. \tag{5}$$

Insight 2: Marker effects reside in a low-rank latent subspace. We observe that the presence of visual markers leads to faster saturation of the cumulative spectral energy $C_\ell(r)$ at small values of r (Fig. 2(b)). This indicates that most marker-induced representational changes can be explained by a small number of leading singular directions. Since the squared singular values $\sigma_{\ell,j}^2$ quantify the variance captured by each singular direction, the rapid concentration of spectral energy implies that the marker-induced variations are largely confined to a low-dimensional subspace, exhibiting a clear low-rank structure. This finding suggests that the effect of visual markers does not diffuse throughout the full high-dimensional representation space, but instead operates through a structured and restricted low-rank latent subspace. Furthermore, when we perform direction-specific suppression within this low-rank subspace during decoding, the grounding performance drops significantly (Fig. 2(c)). This causal intervention provides direct evidence that the performance gains induced by markers are mediated within a low-rank latent subspace at the late decoder layers.

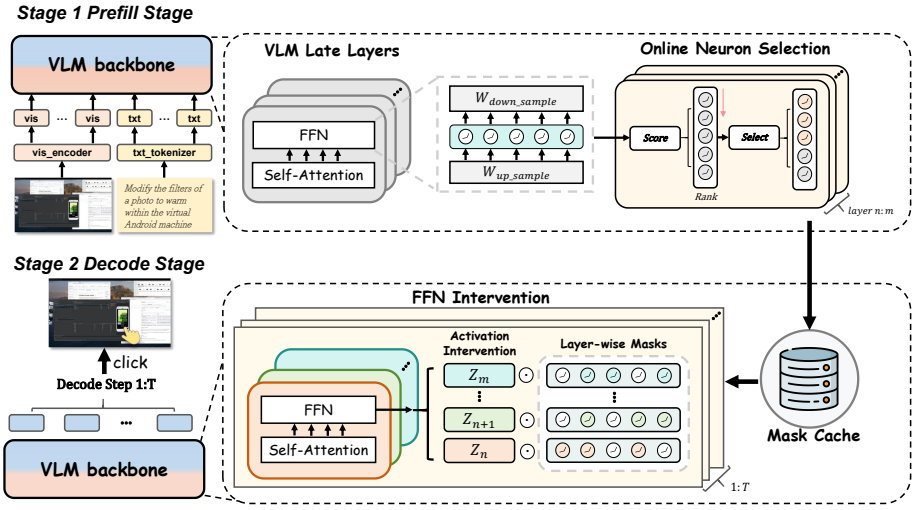

Fig. 3: Overview of *DeCoPatch*. In the prefill stage, we score decoder FFN neurons and cache a sparse top-k subset with the highest scores, effectively isolating a low-rank latent subspace relevant to GUI grounding. During decoding, activation scaling is applied to these selected neurons, enabling targeted causal intervention that steers the model's spatial grounding behavior.

3.3 From Analysis to Design (Q3)

Based on the above mechanistic analysis, we summarize two core design principles: (1) interventions should be applied to the late layers of the decoder, where marker-induced delta energy ΔE_l concentrates, and (2) causal interventions should be performed on key neurons in a low-rank latent subspace, enabling more targeted modulation. *DeCoPatch* operationalizes these principles into a practical framework: it selects the top-k key neurons online in the late decoder layers (Sec. 4.1) and applies patch-based causal interventions to their activations (Sec. 4.2).

4 Methodology

4.1 Online Neuron Selection

To optimize visual grounding, neurons that are relevant to visual information in the model may need to be prioritized. Since visual information is primarily introduced during the prefill stage, capturing these activation patterns early allows for the strategic selection of neurons that can be cached and reused during decoding, thereby reducing computational costs without sacrificing generative performance.

During the prefill phase, we partition the tokens into vision tokens V and text tokens T. For the j-th neuron in the ℓ-th decoder layer, we compute the modality-wise activation energy as:

$$e^v_{\ell,j} = \frac{1}{|V|} \sum_{t \in V} z^2_{\ell,t,j}, \quad e^t_{\ell,j} = \frac{1}{|T|} \sum_{t \in T} z^2_{\ell,t,j}, \tag{6}$$

where $z_{\ell,t,j}$ represents the activation value. The Vision-Text Ratio (VTR) score is then defined as the log ratio of these energies:

$$s^{\text{vr}}_{\ell,j} = \log \left(\frac{e^v_{\ell,j} + \varepsilon}{e^t_{\ell,j} + \varepsilon} \right). \tag{7}$$

We select the top-k neurons based on this VTR $s^{\text{vr}}_{\ell,j}$. These selections are cached and subsequently reused throughout the decoding process to maintain consistency and efficiency.

4.2 Decode-Time Activation Patching

We use the term *patch* to refer to a small-scale intervention on internal activations performed during inference. For the ℓ-th decoder layer, let $z_{\ell,t} \in \mathbb{R}^{d_{ff}}$ denote the intermediate activation corresponding to token t, immediately prior to the down-projection of Feed-Forward Neural Network (FFN). *DeCoPatch* then applies a dimension-wise scaling mask $m_\ell \in \mathbb{R}^{d_{ff}}$:

$$z'_{\ell,t,j} = m_{\ell,j} \odot z_{\ell,t,j}. \tag{8}$$

For selected dimensions S_ℓ of size k, we set $m_{\ell,j} = \alpha$ for $j \in S_\ell$ and $m_{\ell,j} = 1$ otherwise, where α is a fixed patch strength.

Additive View of Activation Patching. Although implemented as scaling, the patch can be read as an additive intervention:

$$z'_{\ell,t} = m_\ell \odot z_{\ell,t} = z_{\ell,t} + (m_\ell - \mathbf{1}) \odot z_{\ell,t}. \tag{9}$$

When this modified activation is passed through the FFN's down-projection $W^\downarrow_\ell$, it produces an additive vector in the residual stream as:

$$W^\downarrow_\ell z'_{\ell,t} = W^\downarrow_\ell z_{\ell,t} + W^\downarrow_\ell \big((m_\ell - \mathbf{1}) \odot z_{\ell,t} \big). \tag{10}$$

Because only k dimensions are changed, the added residual lies in the low-rank subspace spanned by k columns of $W^\downarrow_\ell$.

Table 1: Performance comparison on OSWorld-G. Results are reported as accuracy (%). Higher is better. Bold numbers indicate the best result within each model group.

Model	Method	TextMatch	ElemRecog	Layout	FineManip	Refusal	Overall
Qwen2.5-VL-3B	base	41.38	28.79	34.78	13.42	0.00	27.30
	som	43.30	28.79	34.39	15.44	0.00	27.66
	rand	41.38	29.39	33.99	14.09	0.00	26.95
	ours	**43.68**	**30.30**	**35.18**	14.77	0.00	**29.79**
Qwen2.5-VL-7B	base	44.83	31.82	41.11	17.45	0.00	30.32
	som	45.59	**32.73**	40.71	18.12	0.00	30.67
	rand	45.98	30.91	40.32	16.78	0.00	30.32
	ours	**46.36**	31.21	**42.69**	**19.46**	0.00	**32.09**
Jedi-7B-1080p	base	65.13	54.55	56.92	46.31	**5.56**	53.19
	som	65.52	55.45	57.71	46.98	**5.56**	53.55
	rand	64.37	56.06	56.13	44.97	3.70	52.48
	ours	**66.67**	**56.67**	**58.89**	**47.65**	3.70	**54.26**
UI-Venus-Ground-7B	base	73.95	59.70	60.87	44.97	0.00	57.80
	som	74.33	60.61	**61.66**	45.64	0.00	58.16
	rand	74.71	58.48	61.26	46.31	0.00	57.45
	ours	**75.48**	**61.52**	60.47	**47.65**	0.00	**59.04**
Qwen3-VL-8B	base	63.22	45.91	55.34	41.61	**7.41**	52.48
	som	63.98	47.58	**57.13**	41.61	**7.41**	52.84
	rand	62.45	46.36	55.34	40.94	5.56	51.95
	ours	**65.13**	**48.48**	56.92	**42.62**	**7.41**	**54.26**
InternVL2.5-4B	base	25.67	21.82	27.67	9.40	0.00	20.57
	som	**27.20**	**23.33**	28.85	10.74	0.00	21.10
	rand	24.90	22.73	26.88	10.07	0.00	20.39
	ours	26.44	22.42	**29.64**	**12.08**	0.00	**23.76**
InternVL2.5-7B	base	35.25	30.61	37.15	14.77	0.00	29.79
	som	35.63	**32.12**	38.34	**17.45**	0.00	30.14
	rand	36.02	29.70	36.76	15.44	0.00	29.43
	ours	**36.55**	31.21	**38.74**	16.11	0.00	**32.09**
Pixtral-12B	base	15.71	12.73	17.39	6.04	0.00	13.12
	som	16.48	**14.55**	**19.76**	6.71	0.00	13.48
	rand	14.94	12.42	16.60	6.04	0.00	12.77
	ours	**16.48**	14.24	17.39	**7.38**	0.00	**15.96**

5 Experiments

5.1 Experimental Setup

Benchmarks. We evaluate the effectiveness of *DeCoPatch* for GUI grounding on four challenging benchmarks, including PC benchmarks OSWorld-G [31] and ScreenSpot-Pro [20], mobile benchmark AndroidControl-Curated [19], and cross-platform benchmark ScreenSpot-v2 [29], to comprehensively assess its generalization across different device environments.

Models. We selected six general-purpose VLMs (InternVL2.5-4B/7B [6], Qwen2.5-VL-3B [4], Qwen2.5-VL-7B [4], Qwen3-VL-8B [3], Pixtral-12B [1]) along with two state-of-the-art grounding VLMs (Jedi-7B-1080p [31], UI-Venus-Ground-7B [15]). For each model, we evaluated performance under four intervention

Table 2: Performance comparison on ScreenSpot-Pro. Results are reported as accuracy (%). Higher is better. Bold numbers indicate the best result within each model group.

Model	Method	CAD		Dev		Creative		Scientific		Office		OS		Avg
		T	I	T	I	T	I	T	I	T	I	T	I	
Qwen2.5-VL-3B	base	22.99	6.90	38.13	4.01	40.47	5.28	44.09	10.24	46.96	16.52	32.65	4.08	25.90
	som	**25.29**	8.43	40.80	**8.03**	42.82	**8.80**	46.46	11.81	48.70	18.26	34.69	**6.12**	27.84
	rand	23.37	6.51	37.46	4.35	39.88	5.87	43.70	9.84	46.52	16.96	33.16	4.59	26.01
	ours	**25.29**	**9.20**	**41.47**	6.69	**45.16**	7.04	**48.03**	**14.17**	**50.43**	**19.13**	**35.71**	**6.12**	**28.23**
Qwen2.5-VL-7B	base	17.62	2.30	51.17	4.68	36.66	8.21	48.03	7.87	53.04	18.26	34.69	8.16	27.46
	som	**23.75**	**6.13**	**52.17**	**9.36**	41.64	**10.56**	**51.97**	**11.02**	**56.96**	22.61	**36.73**	9.18	**29.76**
	rand	18.01	2.68	50.84	4.35	37.24	8.50	47.64	8.27	52.61	17.83	35.20	8.67	27.64
	ours	20.69	4.21	**52.17**	8.03	**43.99**	9.97	49.61	9.45	53.91	**23.04**	34.69	**10.20**	29.37
Jedi-7B-1080p	base	38.31	13.79	42.14	10.70	49.85	11.73	72.44	25.20	73.91	46.09	32.65	16.33	39.29
	som	**41.00**	**15.33**	42.81	**12.04**	**51.03**	12.90	73.23	26.77	**75.65**	**48.70**	**34.69**	17.35	**40.76**
	rand	37.93	14.18	42.47	10.37	49.27	12.02	72.05	24.80	74.35	45.65	33.16	16.84	39.18
	ours	39.08	14.56	**44.48**	11.37	50.15	**14.37**	**75.20**	**27.95**	**75.65**	47.83	33.67	**20.41**	40.09
UI-Venus-Ground-7B	base	60.15	21.46	73.91	24.08	**63.05**	14.08	74.80	30.71	73.04	38.26	47.96	22.45	50.30
	som	61.69	23.37	74.58	**25.42**	62.76	16.72	**77.56**	32.68	74.78	40.87	48.98	23.47	50.82
	rand	59.77	22.22	73.58	23.75	62.76	14.37	75.20	30.31	74.35	38.70	48.47	21.94	50.16
	ours	**63.22**	**24.90**	**75.25**	24.75	62.17	**19.65**	76.38	**33.07**	**76.52**	**43.91**	**50.00**	**26.53**	**51.38**
Qwen3-VL-8B	base	48.66	16.86	61.87	28.76	67.16	23.17	71.65	34.65	70.43	40.00	55.10	25.51	46.50
	som	49.43	16.86	62.54	28.76	67.74	23.46	71.65	34.65	**72.17**	**41.74**	55.10	27.55	47.07
	rand	47.89	16.48	61.54	28.09	66.86	22.58	71.26	33.86	69.57	39.57	54.59	25.00	46.14
	ours	**51.34**	**18.39**	**64.21**	**30.10**	**68.33**	**24.05**	**73.23**	**36.22**	71.30	**41.74**	**56.12**	**29.59**	**48.96**
InternVL2.5-4B	base	19.54	4.21	33.44	3.01	31.96	3.81	39.76	7.09	41.74	12.17	28.57	3.06	19.82
	som	21.46	5.36	35.45	4.68	33.43	5.28	42.13	9.45	43.48	13.91	30.61	4.59	22.01
	rand	19.16	3.83	32.78	2.68	31.09	3.52	39.37	6.69	40.87	11.30	27.55	2.55	19.29
	ours	**23.75**	**7.66**	**37.46**	**6.02**	**35.78**	**6.45**	**43.31**	**10.63**	**45.22**	**15.65**	**32.65**	**6.12**	**23.44**
InternVL2.5-7B	base	24.14	5.36	39.80	4.01	37.83	4.40	45.67	9.06	48.70	14.35	33.16	3.57	22.81
	som	25.29	7.66	41.47	5.35	39.88	6.45	47.64	11.02	50.43	16.52	34.69	5.61	24.81
	rand	23.37	4.98	39.13	3.34	37.24	4.11	45.28	8.27	48.26	13.91	32.65	3.06	22.32
	ours	**29.12**	**10.34**	**44.82**	**8.70**	**42.52**	**9.38**	**50.79**	**13.39**	**53.04**	**18.70**	**37.76**	**7.14**	**27.82**

settings: *base* (no modifications), *SoM* (adding SoM physical markers to the original screenshot as input), *random* (random neuron activation), and *ours* (causal intervention under our method). This setup enables a systematic comparison of different intervention strategies on GUI grounding tasks and validates the effectiveness of *DeCoPatch*.

Implementation Details. In the default setting, we intervene on the last 6 decoder FFN layers with $k = 64$ selected neurons per layer and $\alpha = 1.3$. Random control uses the same layers, k, and α, while replacing selected indices with uniformly sampled neurons.

5.2 Main Results

Performance on OSWorld-G. OSWorld-G [31] is a comprehensive PC benchmark for GUI grounding that includes 564 finely annotated examples spanning

Table 3: Performance comparison on AndroidControl-Curated. Results are reported as Type / Grounding / Success Rate (%). Higher is better. Bold numbers indicate the best result within each model group.

Model	Method	AndroidControl-Curated-Easy			AndroidControl-Curated-Hard		
		Type (%)	Grounding (%)	SR (%)	Type (%)	Grounding (%)	SR (%)
Qwen2.5-VL-3B	base	65.18	48.09	31.77	57.25	38.11	22.33
	som	65.05	50.32	33.50	57.12	40.48	24.64
	rand	**65.29**	47.67	31.03	**57.39**	37.72	21.90
	ours	65.22	**51.26**	**34.24**	57.30	**41.03**	**25.07**
Qwen2.5-VL-7B	base	70.82	54.79	38.18	64.39	46.51	29.69
	som	70.65	57.42	39.90	64.22	**49.33**	**31.86**
	rand	**70.95**	54.45	37.44	**64.55**	46.04	29.12
	ours	70.91	**57.85**	**40.15**	64.48	49.21	31.50
Qwen3-VL-8B	base	73.15	57.62	40.39	66.79	49.19	31.57
	som	**73.28**	60.19	42.21	**66.95**	51.87	33.53
	rand	73.01	57.28	39.66	66.64	48.76	31.14
	ours	73.09	**60.45**	**42.86**	66.88	**52.15**	**33.96**
Jedi-7B-1080p	base	81.93	70.69	55.67	76.41	63.22	45.96
	som	**82.07**	73.42	57.64	**76.55**	65.97	47.76
	rand	81.79	70.35	55.17	76.22	62.81	45.45
	ours	82.01	**73.68**	**57.88**	76.48	**66.31**	**48.12**
UI-Venus-Ground-7B	base	78.04	64.41	48.28	71.52	56.29	38.59
	som	77.89	67.28	50.25	71.38	58.79	40.25
	rand	**78.12**	64.13	47.78	**71.64**	56.05	38.37
	ours	77.96	**67.55**	**50.49**	71.45	**59.05**	**40.61**
InternVL2.5-4B	base	59.65	42.68	27.34	52.38	33.92	19.66
	som	59.49	45.51	29.56	52.21	36.42	**21.83**
	rand	**59.78**	42.36	26.85	**52.55**	33.58	19.45
	ours	59.72	**45.77**	**29.80**	52.48	36.19	21.61
InternVL2.5-7B	base	65.95	49.96	33.99	59.65	41.95	25.65
	som	**66.12**	52.79	36.21	59.81	44.68	27.82
	rand	65.81	49.62	33.50	59.49	41.62	25.43
	ours	66.09	**53.15**	**36.45**	**59.88**	**45.25**	**28.60**
Pixtral-12B	base	51.61	34.85	19.70	45.22	27.59	13.37
	som	51.48	37.68	22.17	45.08	30.35	15.48
	rand	**51.77**	34.51	19.46	**45.35**	27.25	13.15
	ours	51.74	**38.04**	**22.41**	45.29	**31.12**	**17.26**

multiple task types. As reported in Tab. 1, *DeCoPatch* exhibits robust generalization, particularly in the *FineManip* and *Layout* sub-tasks. For Jedi-7B-1080p [31], our approach yields the highest overall accuracy of 54.26%, which is 1.07 percentage points higher than the base model. A critical observation is that while the *rand* baseline often degrades performance (e.g., 26.95% for Qwen2.5-VL-3B [4] compared to 27.30% of base), our method consistently stays above the base performance, highlighting the necessity of targeted intervention in the latent space.

Performance on ScreenSpot-Pro. ScreenSpot-Pro [20] presents a significant challenge due to the need to manipulate scientific analysis software. Table 2 evaluates the robustness of our method in these specialized domains. Our

Table 4: Performance comparison on ScreenSpot-v2. Results are reported as accuracy (%). Higher is better. Bold numbers indicate the best result within each model group.

Model	Method	M-Text	M-Icon	D-Text	D-Icon	W-Text	W-Icon	Avg
Qwen2.5-VL-3B	base	92.76	74.88	87.63	57.86	88.03	70.44	80.74
	som	94.48	75.83	86.60	59.29	88.46	72.91	81.76
	rand	92.07	71.56	86.08	56.43	86.32	69.46	79.17
	ours	**94.48**	**76.78**	**90.21**	**63.57**	**90.17**	**74.88**	**83.57**
Qwen2.5-VL-7B	base	**98.28**	86.73	91.24	73.57	92.31	80.79	88.68
	som	97.24	**88.63**	90.21	73.57	**94.87**	81.77	89.23
	rand	96.90	85.78	89.18	72.14	91.88	79.80	87.50
	ours	97.93	88.15	**91.75**	**75.71**	93.59	**82.76**	**89.70**
Jedi-7B-1080p	base	96.55	87.68	95.36	**87.86**	95.30	83.74	91.67
	som	**97.24**	87.68	95.36	**87.86**	94.44	84.24	91.75
	rand	96.21	85.78	94.33	86.43	93.16	83.25	90.49
	ours	96.55	**88.15**	**96.39**	**87.86**	**94.87**	**84.73**	**91.98**
UI-Venus-Ground-7B	base	**99.31**	89.10	96.39	**91.43**	95.73	88.18	93.87
	som	98.97	89.57	**97.94**	**91.43**	95.73	88.18	94.10
	rand	98.62	88.63	95.88	89.29	95.30	**89.66**	93.47
	ours	**99.31**	**89.57**	97.42	**91.43**	**96.58**	**89.66**	**94.50**
Qwen3-VL-8B	base	93.79	**85.31**	92.78	81.43	90.60	79.31	87.97
	som	94.48	83.89	92.78	82.86	**92.31**	78.82	88.29
	rand	93.79	84.36	92.27	80.00	89.74	76.85	87.03
	ours	**95.17**	**85.31**	**93.81**	**83.57**	91.88	**80.79**	**89.15**
InternVL2.5-4B	base	78.97	70.14	82.99	63.57	85.04	72.91	76.57
	som	82.07	73.46	86.60	68.57	88.89	75.37	80.03
	rand	78.62	67.77	81.44	60.71	84.19	69.95	74.92
	ours	**85.52**	**78.20**	**89.18**	**71.43**	**91.88**	**79.31**	**83.49**
InternVL2.5-7B	base	80.34	72.51	85.57	69.29	87.61	76.35	79.32
	som	80.69	76.30	85.05	73.57	87.61	77.34	80.58
	rand	78.28	70.62	84.02	67.86	86.75	73.40	77.52
	ours	**85.52**	**79.62**	**91.24**	**74.29**	**93.16**	**81.28**	**84.91**
Pixtral-12B	base	43.79	37.91	31.96	41.43	29.49	36.45	36.95
	som	48.28	43.60	39.18	48.57	35.04	43.84	43.00
	rand	43.10	35.55	30.93	41.43	27.78	33.99	35.53
	ours	**52.07**	**48.34**	**41.24**	**54.29**	**37.18**	**45.32**	**46.23**

method demonstrates superior precision in these complex scenarios. For instance, Qwen3-VL-8B [3] with *DeCoPatch* achieves the average performance of 48.96%, outperforming its *SoM* counterpart. The consistent improvement in *Creative* and *Scientific* categories (T) suggests that latent-space intervention enhances the model's ability to interpret professional-grade GUI semantics without the need for external visual prompts that might obscure fine-grained details.

Performance on AndroidControl-Curated. Evaluation on mobile benchmark AndroidControl-Curated [19] further confirms the versatility of *DeCoPatch* (Tab. 3). In the *Hard* subset, which requires long-horizon reasoning and precise grounding, Qwen3-VL-8B (ours) [3] reaches a success rate of 33.96%, significantly higher than the base (31.57%) and rand (31.14%) configurations. The improvement in *Grounding* scores across the board (e.g., 60.45% and 57.62%

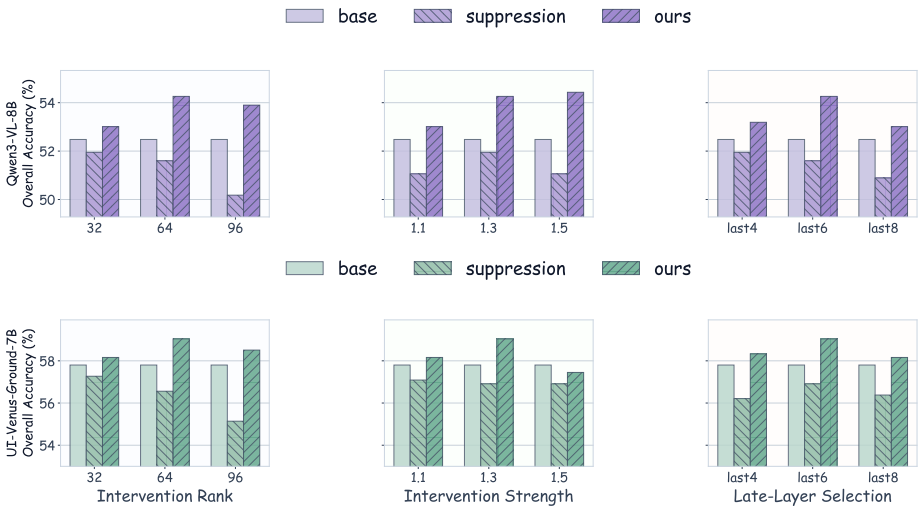

Fig. 4: Results of the hyperparameter analysis. Colors denote models, while hatching styles denote methods (base, ours, and suppression) under different rank of intervention k, intervention strength α, and late-layer selection.

for Qwen3-VL-8B [3] on the *Easy* subset) also indicates that reinforcing the latent grounding signals allows the model to better locate small, dense interactive elements typical of mobile operating systems.

Performance on ScreenSpot-v2. ScreenSpot-v2 [29] is a cross-platform GUI grounding benchmark across mobile, desktop, and web interfaces. As reported in Tab. 4, our method (*ours*) achieves a consistent performance gain across all models on this benchmark. Notably, for InternVL2.5-7B [6], *DeCoPatch* reaches an average accuracy of 84.91%, surpassing the *base* model and the *SoM*-based [32] approach by 5.59 and 4.33 percentage points, respectively. The performance gap between *ours* and *rand* strongly validates that our neuron selection successfully identifies the causal grounding circuits rather than relying on random noise injection. Even for top-tier models like UI-Venus-Ground-7B [15], our method further pushes the performance ceiling to 94.50%.

Across these four evaluation benchmarks, *DeCoPatch* yields notable performance gains across different model families. Notably, the random control results are typically close to the baseline, highlighting the importance of our designed key-neuron modulation strategy.

5.3 Analysis

Hyperparameter Analysis. We analyze the sensitivity of *DeCoPatch* to several key hyperparameters that control the strength and scope of the intervention. All experiments are conducted on the OSWorld-G [31] benchmark. We vary three

Table 5: Runtime overhead of *DeCoPatch*. We report the wall-clock latency per example and the relative change compared to base inference under the same batch size and precision settings.

Suite	Model	base (ms)	ours (ms)	relative change (%)
ScreenSpot-v2	Qwen3-VL-8B	6823.0	6831.2	0.12
ScreenSpot-v2	UI-Venus-Ground-7B	2698.4	2403.0	-10.95
ScreenSpot-Pro	Qwen3-VL-8B	7046.2	7041.1	-0.07
ScreenSpot-Pro	UI-Venus-Ground-7B	2735.6	2576.2	-5.83
AndroidControl	Qwen3-VL-8B	6584.3	6610.8	0.40
AndroidControl	UI-Venus-Ground-7B	3480.3	3211.3	-7.73
OSWorld-G	Qwen3-VL-8B	6841.8	6879.9	0.56
OSWorld-G	UI-Venus-Ground-7B	2525.6	2393.1	-5.25

hyperparameters: the intervention rank (k), the intervention strength (α), and the number of late decoder layers to which the intervention is applied. Specifically, we sweep $k \in \{32, 64, 96\}$, $\alpha \in \{1.1, 1.3, 1.5\}$, and the intervention layers $\in$ {last4,last6,last8}. For the suppression variant, we apply the same neuron mask but scale down the selected activations by division, i.e., $z'_{\ell,t,j} = z_{\ell,t,j}/\alpha$ for $j \in S_\ell$ (and $z'_{\ell,t,j} = z_{\ell,t,j}$ otherwise; default $\alpha = 1.3$), which reduces the activation magnitude of the selected neurons.

As shown in Fig. 4, *DeCoPatch* consistently improves over the base model for both Qwen3-VL-8B [3] and UI-Venus-Ground-7B [15] under different settings of intervention rank k, strength α, and late-layer selection. Performance typically peaks with a moderate intervention budget (e.g., k=64) and moderate scaling (e.g., α=1.3), while overly large k or stronger scaling can lead to saturation or instability. In contrast, applying suppression on the same neuron set consistently degrades accuracy, supporting our claim that these late-layer FFN dimensions form a grounding-relevant low-rank subspace.

Runtime Overhead Analysis. We measure the end-to-end inference latency in milliseconds per example under identical batch size and precision settings for both the base models and *DeCoPatch* (Tab. 5). The results show that the additional runtime overhead introduced by *DeCoPatch* is negligible. For Qwen3-VL-8B [3], the latency increase is within 1%, while for UI-Venus-Ground-7B [15] the change remains within 11%. These results indicate that *DeCoPatch* improves grounding performance with almost no additional computational cost.

6 Conclusion

In this work, we systematically investigate the pivotal role of visual markers in shaping the internal reasoning of VLMs within GUI environments. Our analysis reveals that the causal influence of these markers is highly concentrated within low-rank latent subspaces in the late decoder layers. Building on this insight, we propose *DeCoPatch*, a decode-time causal intervention that selectively

modulates the activations of key neurons, enabling precise reinforcement of the model's internal visual reasoning. Extensive experiments across four challenging benchmarks demonstrate that this method consistently improves GUI grounding performance with negligible computational overhead. Importantly, beyond offering an effective intervention technique, our work provides a deeper mechanistic understanding of how visual markers influence VLMs' internal representations, laying a critical foundation for more precise and controllable improvements in GUI agent optimization.

Acknowledgments and Disclosure of Funding

We thank the ECCV reviewers for their invaluable feedback. This work was supported in part by the National Natural Science Foundation of China (Grant No. 92467204 and 62472249), and the Shenzhen Science and Technology Program (Grant No. KJZD20240903102300001).

References

1. Agrawal, P., Antoniak, S., Hanna, E.B., Bout, B., Chaplot, D., Chudnovsky, J., Costa, D., De Monicault, B., Garg, S., Gervet, T., et al.: Pixtral 12b. arXiv preprint arXiv:2410.07073 (2024)
2. Bai, J., Bai, S., Yang, S., Wang, S., Tan, S., Wang, P., Lin, J., Zhou, C., Zhou, J.: Qwen-VL: A versatile vision-language model for understanding, localization, text reading, and beyond (2023), https://arxiv.org/abs/2308.12966
3. Bai, S., Cai, Y., Chen, R., Chen, K., Chen, X., Cheng, Z., Deng, L., Ding, W., Gao, C., Ge, C., et al.: Qwen3-vl technical report. arXiv preprint arXiv:2511.21631 (2025)
4. Bai, S., Chen, K., Liu, X., Wang, J., Ge, W., Song, S., Dang, K., Wang, P., Wang, S., Tang, J., Zhong, H., Zhu, Y., Yang, M., Li, Z., Wan, J., Wang, P., Ding, W., Fu, Z., Xu, Y., Ye, J., Zhang, X., Xie, T., Cheng, Z., Zhang, H., Yang, Z., Xu, H., Lin, J.: Qwen2.5-vl technical report (2025), https://arxiv.org/abs/2502.13923
5. Chen, L., Zhou, H., Cai, C., Zhang, J., Tong, P., Kong, Q., Zhang, X., Liu, C., Liu, Y., Wang, W., et al.: Ui-ins: Enhancing gui grounding with multi-perspective instruction-as-reasoning. arXiv preprint arXiv:2510.20286 (2025)
6. Chen, Z., Wang, W., Tian, H., Ye, S., Gao, Z., Cui, E., Tong, W., Hu, K., Luo, J., Ma, Z., Ma, J., Wang, J., Dong, X., Yan, H., Guo, H., He, C., Shi, B., Jin, Z., Xu, C., Wang, B., Wei, X., Li, W., Zhang, W., Zhang, B., Cai, P., Wen, L., Yan, X., Dou, M., Lu, L., Zhu, X., Lu, T., Lin, D., Qiao, Y., Dai, J., Wang, W.: How far are we to GPT-4V? closing the gap to commercial multimodal models with open-source suites (2024), https://arxiv.org/abs/2404.16821
7. Cheng, K., Sun, Q., Chu, Y., Xu, F., Yantao, L., Zhang, J., Wu, Z.: 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). pp. 9313–9332. Association for Computational Linguistics, Bangkok, Thailand (Aug 2024). https://doi.org/10.18653/v1/2024.acl-long.505, https://aclanthology.org/2024.acl-long.505/

8. Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., Wei, F.: Knowledge neurons in pretrained transformers. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 8493–8502. Association for Computational Linguistics, Dublin, Ireland (May 2022). https://doi.org/10.18653/v1/2022.acl-long.581, https://aclanthology.org/2022.acl-long.581/
9. Du, Y., Yan, Y., Tang, F., Lu, Z., Zong, C., Lu, W., Jiang, S., Shen, Y.: Test-time reinforcement learning for gui grounding via region consistency. arXiv preprint arXiv:2508.05615 (2025)
10. Fan, Y., Zhao, H., Zhang, R., Shen, Y., Wang, X.E., Wu, G.: Gui-bee: Align gui action grounding to novel environments via autonomous exploration. In: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing. pp. 33249–33266 (2025)
11. Fang, J., Jiang, H., Wang, K., Ma, Y., Shi, J., Wang, X., He, X., Chua, T.S.: Alphaedit: Null-space constrained model editing for language models. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=HvSytvg3Jh
12. Gao, C., Chen, H., Xiao, C., Chen, Z., Liu, Z., Sun, M.: H-neurons: On the existence, impact, and origin of hallucination-associated neurons in llms (2025), https://arxiv.org/abs/2512.01797
13. Gao, M., Bu, W., Miao, B., Wu, Y., Li, Y., Li, J., Tang, S., Wu, Q., Zhuang, Y., Wang, M.: Generalist virtual agents: A survey on autonomous agents across digital platforms. arXiv preprint arXiv:2411.10943 (2024)
14. Gou, B., Wang, R., Zheng, B., Xie, Y., Chang, C., Shu, Y., Sun, H., Su, Y.: Navigating the digital world as humans do: Universal visual grounding for GUI agents. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=kxnoqaisCT
15. Gu, Z., Zeng, Z., Xu, Z., Zhou, X., Shen, S., Liu, Y., Zhou, B., Meng, C., Xia, T., Chen, W., et al.: Ui-venus technical report: Building high-performance ui agents with rft. arXiv preprint arXiv:2508.10833 (2025)
16. Hong, W., Yu, W., Gu, X., Wang, G., Gan, G., Tang, H., Cheng, J., Qi, J., Ji, J., Pan, L., et al.: Glm-4.5 v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. arXiv preprint arXiv:2507.01006 (2025)
17. Hu, X., Xiong, T., Yi, B., Wei, Z., Xiao, R., Chen, Y., Ye, J., Tao, M., Zhou, X., Zhao, Z., et al.: Os agents: A survey on mllm-based agents for general computing devices use. arXiv preprint arXiv:2508.04482 (2025)
18. Kang, L., Wang, Z., Zhang, Y., Wu, D., Wang, J., Ma, M., Yan, H., Wang, Z.: Learning with challenges: Adaptive difficulty-aware data generation for mobile gui agent training. arXiv preprint arXiv:2601.22781 (2026)
19. Leung, H.F., Xi, X., Zuo, F.: Androidcontrol-curated: Revealing the true potential of gui agents through benchmark purification (2025), https://arxiv.org/abs/2510.18488
20. Li, K., Meng, Z., Lin, H., Luo, Z., Tian, Y., Ma, J., Huang, Z., Chua, T.S.: Screenspot-pro: Gui grounding for professional high-resolution computer use. In: Proceedings of the 33rd ACM International Conference on Multimedia. pp. 8778–8786 (2025)
21. Li, W., Shao, Y., Yang, J., Lu, Y., Zhong, L., Wang, Y., Duan, M.: How auxiliary reasoning unleashes gui grounding in vlms. arXiv preprint arXiv:2509.11548 (2025)
22. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning (2023), https://arxiv.org/abs/2304.08485

23. Liu, X., Zhang, X., Zhang, Z., Lu, Y.: Ui-e2i-synth: Advancing gui grounding with large-scale instruction synthesis. In: Findings of the Association for Computational Linguistics: ACL 2025. pp. 15668–15684 (2025)
24. Luo, T., Logeswaran, L., Johnson, J., Lee, H.: Visual test-time scaling for gui agent grounding. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19989–19998 (2025)
25. Qin, Y., Ye, Y., Fang, J., Wang, H., Liang, S., Tian, S., Zhang, J., Li, J., Li, Y., Huang, S., et al.: Ui-tars: Pioneering automated gui interaction with native agents. arXiv preprint arXiv:2501.12326 (2025)
26. Rawles, C., Clinckemaillie, S., Chang, Y., Waltz, J., Lau, G., Fair, M., Li, A., Bishop, W.E., Li, W., Campbell-Ajala, F., Toyama, D.K., Berry, R.J., Tyamagundlu, D., Lillicrap, T.P., Riva, O.: Androidworld: A dynamic benchmarking environment for autonomous agents. In: The Thirteenth International Conference on Learning Representations (2025), https://openreview.net/forum?id=il5yUQsrjC
27. Wang, X., Wen, K., Zhang, Z., Hou, L., Liu, Z., Li, J.: Finding skill neurons in pre-trained transformer-based language models. In: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing. pp. 11132–11152 (2022)
28. Wu, Q., Cheng, K., Yang, R., Zhang, C., Yang, J., Jiang, H., Mu, J., Peng, B., Qiao, B., Tan, R., et al.: Gui-actor: Coordinate-free visual grounding for gui agents. arXiv preprint arXiv:2506.03143 (2025)
29. Wu, Z., Wu, Z., Xu, F., Wang, Y., Sun, Q., Jia, C., Cheng, K., Ding, Z., Chen, L., Liang, P.P., Qiao, Y.: OS-ATLAS: A foundation action model for generalist GUI agents (2024), https://arxiv.org/abs/2410.23218
30. Xiao, L., Yang, X., Lan, X., Wang, Y., Xu, C.: Toward visual grounding: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence **48**(3), 2749–2771 (2026). https://doi.org/10.1109/TPAMI.2025.3630635
31. Xie, T., Deng, J., Li, X., Yang, J., Wu, H., Chen, J., Hu, W., Wang, X., Xu, Y., Wang, Z., Xu, Y., Wang, J., Sahoo, D., Yu, T., Xiong, C.: Scaling computer-use grounding via user interface decomposition and synthesis. In: The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track (2025), https://openreview.net/forum?id=FS0voKxELr
32. Yang, J., Zhang, H., Li, F., Zou, X., Li, C., Gao, J.: Set-of-mark prompting unleashes extraordinary visual grounding in GPT-4V (2023), https://arxiv.org/abs/2310.11441
33. Yang, J., Xiong, T., Qian, S., Nahrstedt, K., Wu, M.: Circuit tracing in vision-language models: Understanding the internal mechanisms of multimodal thinking. arXiv preprint arXiv:2602.20330 (2026)
34. Yang, Q., Bi, W., Shen, H., Guo, Y., Ma, Y.: Pixelweb: The first web gui dataset with pixel-wise labels. arXiv preprint arXiv:2504.16419 (2025)
35. Yang, Y., Li, D., Dai, Y., Yang, Y., Luo, Z., Zhao, Z., Hu, Z., Huang, J., Saha, A., Chen, Z., et al.: Gta1: Gui test-time scaling agent. arXiv preprint arXiv:2507.05791 (2025)
36. You, K., Zhang, H., Schoop, E., Weers, F., Swearngin, A., Nichols, J., Yang, Y., Gan, Z.: Ferret-ui: Grounded mobile ui understanding with multimodal llms. In: European Conference on Computer Vision. pp. 240–255. Springer (2024)
37. Yu, Z., Lee, Y.J.: How multimodal llms solve image tasks: A lens on visual grounding, task reasoning, and answer decoding. arXiv preprint arXiv:2508.20279 (2025)
38. Zhang, C., He, S., Qian, J., Li, B., Li, L., Qin, S., Kang, Y., Ma, M., Liu, G., Lin, Q., et al.: Large language model-brained gui agents: A survey. arXiv preprint arXiv:2411.18279 (2024)

39. Zhang, F., Nanda, N.: Towards best practices of activation patching in language models: Metrics and methods. In: International Conference on Learning Representations. vol. 2024, pp. 1651–1678 (2024)
40. Zhang, H., Li, H., Li, F., Ren, T., Zou, X., Liu, S., Huang, S., Gao, J., Leizhang, Li, C., et al.: Llava-grounding: Grounded visual chat with large multimodal models. In: European Conference on Computer Vision. pp. 19–35. Springer (2024)
41. Zhang, S., Fu, P., Zhang, R., Yang, J., Du, A., Xi, X., Wang, S., Huang, Y., Qin, B., Luo, Z., et al.: Hyperclick: Advancing reliable gui grounding via uncertainty calibration. arXiv preprint arXiv:2510.27266 (2025)
42. Zhang, Z., Li, X., Xu, Z., Peng, W., Zhou, Z., Shi, M., Huang, S.: Mpdrive: Improving spatial understanding with marker-based prompt learning for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12089–12099 (June 2025)
43. Zheng, B., Gou, B., Kil, J., Sun, H., Su, Y.: Gpt-4v (ision) is a generalist web agent, if grounded. In: Proceedings of the 41st International Conference on Machine Learning. pp. 61349–61385 (2024)
44. Zhu, K., Qin, Z., Yi, H., Jiang, Z., Lao, Q., Zhang, S., Li, K.: Guiding medical vision-language models with diverse visual prompts: Framework design and comprehensive exploration of prompt variations. In: Chiruzzo, L., Ritter, A., Wang, L. (eds.) Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers). pp. 11726–11739. Association for Computational Linguistics, Albuquerque, New Mexico (Apr 2025). https://doi.org/10.18653/v1/2025.naacl-long.587, https://aclanthology.org/2025.naacl-long.587/