SEG2CLICK: LEARNING PIXEL-LEVEL ACTION MASKS FOR GUI VISUAL GROUNDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual grounding is a central challenge for GUI agents, requiring an instruction to be mapped to a target within a crowded screenshot. Coordinate generation with single-point supervision reduces a spatially extended target to one position, despite the presence of many valid click locations. Region-based patch scoring alleviates this ambiguity, but its spatial decisions remain tied to visual-token footprints that may span the target, neighboring controls, and background. To address these limitations, we propose Seg2Click, which reformulates GUI grounding as executable-region prediction: identifying where a click can validly execute the given instruction. Using bounding-box supervision, Seg2Click predicts an executable-region mask and computes the final click point through a deterministic rule. This formulation captures the spatial extent of valid click locations and enables finer localization beyond the visual-token grid, making it well suited to small, densely arranged controls without requiring a verifier or zoom-in pass. Experiments across five backbone configurations from two model families demonstrate the effectiveness of Seg2Click. Seg2Click-9B achieves 92.06%, 94.10%, and 55.22% on ScreenSpot, ScreenSpot-v2, and ScreenSpot-Pro, respectively, while matched-backbone comparisons show improvements over both coordinate generation and patch-based grounding.

1 INTRODUCTION

GUI agents carry out user instructions across web, mobile, and desktop environments (Zhou et al., 2024; Rawles et al., 2025; Xie et al., 2024). A key capability is visual grounding: locating the target specified by an instruction within a screenshot (Cheng et al., 2024; Gou et al., 2025; Qin et al., 2025). This task is particularly challenging in dense professional interfaces, where small controls are often placed close together (Li et al., 2025). Recognizing the relevant toolbar or control group is insufficient: successful interaction requires placing the click within the specific element named by the instruction.

Existing approaches commonly address this localization problem through either direct coordinate generation or relevance prediction over visual tokens. Coordinate-generation methods represent screen positions as text and directly predict a click location (Chen et al., 2023; Cheng et al., 2024; Wu et al., 2025b). Under single-point supervision, these methods are trained to reproduce one annotated coordinate, even though many locations within the target may be equally valid. For example, different clicks inside a button can trigger the same action, whereas a nearby click just outside its boundary may activate another control. Single-point supervision does not explicitly capture this distinction, because it penalizes deviations from the annotated point regardless of whether they remain within the target. This mismatch motivates predicting the valid target region before selecting a specific click location. Visual-token-based methods move toward region prediction by assigning relevance scores to image patches. In particular, GUI-Actor supervises all visual patches that overlap the target, allowing multiple patches to receive positive supervision (Wu et al., 2025a). Although this approach relaxes the commitment to a single annotated point, its spatial decisions remain tied to the visual-token grid. A single token footprint may cover part of the target, neighboring controls, and background, yet receive only one relevance score. Interpolating these scores increases output resolution but does not introduce independently learned predictions within each footprint. Thus, moving from point supervision to region supervision leaves a granularity bottleneck unresolved, es-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

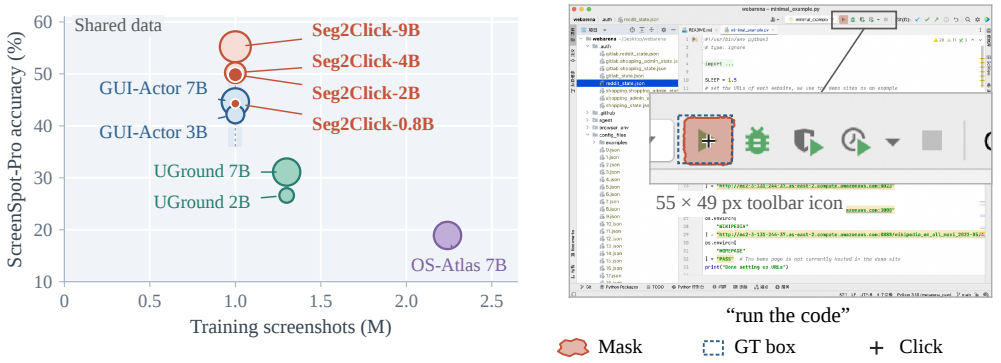

Figure 1: **Left:** ScreenSpot-Pro accuracy versus training screenshots; marker area denotes backbone size. Qwen3.5-based Seg2Click and GUI-Actor share the same grounding data, aligned at ∼1M screenshots. **Right:** Qwen3.5-4B predicts the mask and click for "run the code." The display-only inset shows the 55×49 pixel icon and its ground-truth box.

pecially for small, densely arranged controls. The remaining challenge is to preserve region-based supervision while learning spatial decisions finer than the visual tokens that represent the screenshot.

To address this challenge, we propose Seg2Click, which reformulates GUI grounding as executable-region prediction: predicting the region in which a click can validly execute the instruction, then computing the click point using a deterministic rule. We instantiate this formulation as box-supervised sub-token action-field prediction, learning distinct scores within each visual-token footprint (Figure 2). A query-conditioned pixel-shuffle decoder (Shi et al., 2016; Wang et al., 2025) combines aligned vision and language-model features to produce an executable-region mask at the processed screenshot resolution. Target bounding boxes provide supervision, and the centroid of the mask's largest connected component determines the final click. This formulation addresses the two limitations above: region supervision accommodates multiple valid click locations, while sub-token prediction enables finer spatial distinctions within token footprints. It requires no additional contour annotations, verifier, or zoom-in pass.

To assess whether these changes translate into more accurate grounding, we first compare Seg2Click with coordinate generation and patch scoring, then evaluate its applicability across backbones. Under matched backbone, training data, and optimization settings, Seg2Click-0.8B produces 24 more correct clicks than coordinate generation on ScreenSpot-Pro. On the same 0.8B backbone, it also outperforms reproduced GUI-Actor by 4.87 percentage points, corresponding to 77 additional correct clicks (Table 6). Beyond these comparisons, experiments cover five backbone configurations from two model families. Seg2Click achieves 47.00% on ScreenSpot-Pro with Qwen2.5-VL-3B and 55.22% with Qwen3.5-9B, supporting the effectiveness of executable-region prediction across model families and scales (Figure 1).

Our contributions are threefold:

- We reformulate GUI grounding as executable-region prediction, separating the learning of valid click regions from deterministic click selection.
- We introduce Seg2Click, which learns box-supervised sub-token action fields to resolve spatial distinctions within each visual-token region, without additional contour annotations, a verifier, or a zoom-in pass.
- We validate Seg2Click across five backbone configurations from two model families. It outperforms coordinate generation under matched training and reproduced patch scoring on the same backbone, with the largest gain over patch scoring on ScreenSpot-Pro. Our 9B model achieves the highest accuracy among the compared systems on all three benchmarks, while feature and decoder analyses support the design choices.

2 Related Work

GUI understanding and coordinate-based grounding. GUI agents map user instructions to interface elements through visual understanding and grounding. UIBert, Ferret-UI, and OmniParser advance UI representation and understanding (Bai et al., 2021; You et al., 2024; Lu et al., 2024), while Mind2Web, WebVoyager, Aguvis, and ShowUI support learning and evaluating visually grounded interaction (Deng et al., 2023; He et al., 2024; Xu et al., 2025b; Lin et al., 2025). A common grounding formulation directly generates target coordinates from screenshots and instructions. SeeClick, OS-Atlas, Aria-UI, and TinyClick improve this formulation through specialized models and training strategies (Cheng et al., 2024; Wu et al., 2025b; Yang et al., 2025; Pawlowski et al., 2025), while UGround and Jedi broaden grounding coverage through scalable data construction (Gou et al., 2025; Xie et al., 2025). Seg2Click addresses a complementary limitation: single-point supervision does not explicitly capture multiple valid click locations within a target. It instead predicts a box-defined action region before deriving the final coordinate.

Attention- and region-based grounding. Spatial relevance prediction offers an alternative to direct coordinate generation. TAG aggregates pretrained multimodal attention to localize GUI targets without fine-tuning (Xu et al., 2025a). GUI-Actor learns an attention-based action head with multi-patch supervision and an optional verifier for candidate selection (Wu et al., 2025a). Seg2Click builds on region-based grounding by refining prediction granularity: a single patch score cannot distinguish target and non-target locations within the same token footprint. Its spatial decoder instead learns distinct action scores within each footprint, and a fixed largest-component centroid rule converts the predicted mask into a click, without a verifier or zoom-in pass.

Language-conditioned dense prediction. Language-conditioned segmentation connects multimodal understanding with dense prediction. LISA decodes segmentation-token representations into masks, while PixelLM uses a segmentation codebook and pixel decoder for multi-target reasoning (Lai et al., 2024; Ren et al., 2024). MLLMSeg combines visual and language-model features with pixel-shuffle decoding (Wang et al., 2025). Seg2Click adapts these techniques to GUI grounding through dynamic-resolution grids and an action-placeholder query, predicting a box-defined action mask for deterministic click extraction. Unlike BoxInst, which uses boxes as weak supervision to recover object masks (Tian et al., 2021), Seg2Click directly learns the region defined by the benchmark's box-hit criterion, without requiring contour annotations.

3 Seg2Click

3.1 Action-Mask Formulation

GUI grounding maps a screenshot I and an instruction u to a click location. Under the box-hit evaluation protocol, multiple locations within the annotated target box b are equally valid. Seg2Click represents this spatial target as a dense **action mask**, separating region prediction from click selection. The annotated box defines the supervision region and serves as a proxy for the executable area in static grounding.

Let $I' \in \mathbb{R}^{H' \times W' \times 3}$ be the screenshot after the backbone's dynamic-resolution preprocessing. We predict

$$P_\theta(I', u) = \sigma(L_\theta(I', u)) \in [0,1]^{H' \times W'}, \qquad \hat{\mathbf{c}} = D(P_\theta(I', u)),$$

where L_θ is a dense logit field, σ is sigmoid, and D is a deterministic rule that returns a normalized click. Pixel-level prediction refers to the processed screenshot resolution. As shown in Figure 2, Seg2Click combines aligned visual features with an instruction-aware action query, predicts distinct scores within each visual-token footprint, and converts the resulting mask into a single click.

3.2 Query-Conditioned Visual Features

The backbone represents the processed screenshot on an $h \times w$ visual-token grid. At grid location i, we extract the post-merge vision feature $\mathbf{z}_i$ and the final language-model state $\mathbf{v}_i$ at the corresponding image-token position. Both lie in $\mathbb{R}^d$ and share the same spatial correspondence. We combine

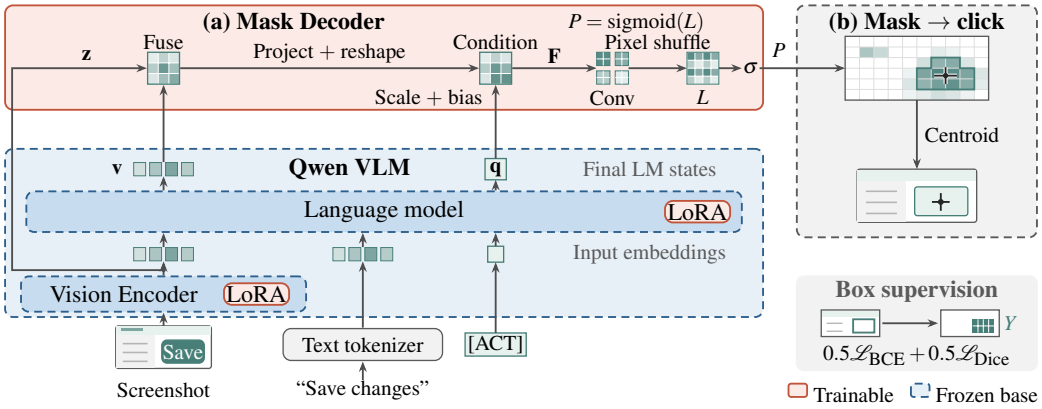

Figure 2: Overview of Seg2Click. (a) Aligned vision and language-model features are fused and conditioned on an action query to predict distinct scores within each visual-token footprint, supervised by masks derived from target boxes. (b) A fixed rule converts the predicted mask into a click at the centroid of its largest connected component. Orange denotes trainable modules; blue dashed outlines indicate frozen backbone weights.

these two representations through residual fusion:

$$\mathbf{f}_i = \mathbf{v}_i + \mathrm{MLP}([\mathbf{v}_i; \phi(\mathbf{z}_i)]) \,, \tag{1}$$

where ϕ projects the direct vision features and the MLP learns a correction to $\mathbf{v}_i$ from the concatenated streams.

To condition spatial prediction on the instruction, we append a fixed action placeholder after the screenshot and instruction and use its final hidden state as the action query $\mathbf{q}$. This position can incorporate both inputs and supplies their joint context to the decoder. After projecting and reshaping the fused features into $X \in \mathbb{R}^{C\times h\times w}$, we apply query-dependent channel modulation (Perez et al., 2018):

$$F_{:,r,c} = X_{:,r,c} \odot \big(1 + \boldsymbol{\gamma}(\mathbf{q})\big) + \boldsymbol{\beta}(\mathbf{q}), \tag{2}$$

where $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are learned scale and bias functions. Although shared across spatial positions, these modulation parameters act on position-dependent visual features, allowing the subsequent decoder to produce an instruction-specific mask.

3.3 Sub-Token Action-Mask Prediction

A single visual token can span both the target and surrounding controls. Assigning one score to that token cannot directly distinguish these locations. Seg2Click therefore predicts multiple scores within each token footprint. Let s be the grid stride in processed pixels, so that $H' = sh$ and $W' = sw$. Each grid cell produces s^2 logits, one for each pixel offset within its $s \times s$ footprint.

A convolutional decoder aggregates neighboring conditioned features and outputs these logits. Pixel shuffle (Shi et al., 2016; Wang et al., 2025) then rearranges them into the dense field:

$$U = g(F) \in \mathbb{R}^{s^2\times h\times w}, \qquad L = \mathrm{PS}_s(U) \in \mathbb{R}^{H'\times W'}, \tag{3}$$

where g consists of two spatial convolutional blocks followed by an output projection. Each output channel corresponds to a distinct within-cell offset. The learned decoder thus determines the sub-token scores, while pixel shuffle assigns them to their spatial locations. This differs from interpolating a token-level score map: the model learns separate outputs within each footprint, enabling finer localization while retaining the backbone's visual-token grid. Decoder configurations and the exact channel-to-pixel mapping are provided in Appendix A.

Table 1: Performance comparison on ScreenSpot-Pro, a benchmark of high-resolution professional interfaces. Dev, Creat., and Sci. denote Development, Creative, and Scientific, respectively.

			Application domain						Target type		
Method	Backbone VLM	Venue / Year	Dev	Creat.	CAD	Sci.	Office	OS	Text	Icon	Avg
Aria-UI	Aria-25.3B	ACL Findings 2025	8.4	14.7	6.1	18.1	16.1	2.6	17.1	2.0	11.3
CogAgent	CogVLM-17B	CVPR 2024	8.0	5.6	6.1	13.4	10.0	3.1	12.0	0.8	7.7
SeeClick	Qwen-VL-Chat	ACL 2024	0.3	0.6	1.9	2.0	0.9	1.5	1.8	0.0	1.1
ShowUI	Qwen2-VL-2B	CVPR 2025	9.4	5.3	1.9	10.6	13.5	6.6	10.8	2.6	7.7
UGround-V1	Qwen2-VL-2B	ICLR 2025	27.4	26.7	14.6	34.3	38.3	17.9	–	–	26.6
UI-TARS	Qwen2-VL-2B	arXiv 2025	26.4	27.6	14.6	39.8	42.6	14.3	39.6	8.4	27.7
OS-Atlas	Qwen2-VL-7B	ICLR 2025	17.7	17.9	10.3	24.4	27.4	16.8	28.1	4.0	18.9
Aguvis	Qwen2-VL-7B	ICML 2025	16.1	21.4	13.8	34.6	34.3	19.4	–	–	22.9
UGround-V1	Qwen2-VL-7B	ICLR 2025	28.1	31.7	14.6	39.0	49.6	24.5	–	–	31.1
UI-TARS	Qwen2-VL-7B	arXiv 2025	36.1	32.8	18.0	50.0	53.5	24.5	47.8	16.2	35.7
Qwen2.5-VL	Qwen2.5-VL-3B	arXiv 2025	21.4	25.8	18.4	29.5	40.9	20.4	–	–	25.9
Jedi	Qwen2.5-VL-3B	NeurIPS 2025	38.1	34.6	23.0	38.6	57.0	25.0	–	–	36.1
GUI-Actor	Qwen2.5-VL-3B	NeurIPS 2025	39.8	36.7	34.1	49.6	61.3	35.2	–	–	42.2
GUI-Actor + V.	Qwen2.5-VL-3B	NeurIPS 2025	43.8	37.8	43.3	48.4	63.5	42.9	–	–	45.9
Seg2Click	Qwen2.5-VL-3B	Ours	43.5	41.3	37.9	<u>53.1</u>	67.8	41.8	62.7	21.5	47.0
Qwen2.5-VL	Qwen2.5-VL-7B	arXiv 2025	29.1	24.9	13.8	31.1	45.7	22.4	–	–	27.6
Jedi	Qwen2.5-VL-7B	NeurIPS 2025	27.4	34.0	32.2	52.4	68.7	26.0	–	–	39.5
GUI-Actor	Qwen2.5-VL-7B	NeurIPS 2025	38.1	41.4	38.3	50.8	63.0	38.8	–	–	44.6
GUI-Actor + V.	Qwen2.5-VL-7B	NeurIPS 2025	43.1	41.9	<u>48.3</u>	47.2	65.7	43.4	–	–	47.7
Seg2Click	Qwen3.5-0.8B	Ours	40.1	41.3	35.6	46.9	64.3	40.3	61.5	16.4	44.3
Seg2Click	Qwen3.5-2B	Ours	**47.2**	45.7	37.9	**53.5**	69.6	49.0	<u>65.2</u>	25.0	49.8
Seg2Click	Qwen3.5-4B	Ours	<u>45.5</u>	<u>48.1</u>	39.5	51.6	<u>70.0</u>	<u>50.5</u>	<u>65.2</u>	<u>26.0</u>	<u>50.2</u>
Seg2Click	Qwen3.5-9B	Ours	44.1	**53.7**	**59.0**	48.0	**77.4**	**53.1**	**68.9**	**33.1**	**55.2**

3.4 Learning and Click Extraction.

Box-supervised learning. We rasterize the annotated target box into a binary mask $Y \in \{0,1\}^{H' \times W'}$ and train the prediction P with

$$\mathcal{L}_{\text{mask}} = \tfrac{1}{2}\mathcal{L}_{\text{BCE}}(P, Y) + \tfrac{1}{2}\mathcal{L}_{\text{Dice}}(P, Y). \tag{4}$$

Binary cross-entropy supervises pixel labels, while Dice loss measures overlap with the target region. This objective learns the spatial extent specified by existing box annotations without requiring contour labels. We optimize the fusion and decoder parameters together with LoRA adapters in the backbone (Hu et al., 2022), keeping the remaining parameters frozen. Appendix A specifies target rasterization, loss aggregation, and training settings.

Deterministic click extraction. At inference, we threshold P at 0.5, select its largest four-connected foreground component, and return the centroid of its pixel centers as the click. Selecting one component avoids averaging across disconnected predictions. The normalized coordinate is mapped back to the original screenshot by scaling with its width and height. We use this fixed rule for all backbones and benchmarks; Algorithm 1 in Appendix A gives the full inference procedure. Each screenshot–instruction pair requires one mask-producing prediction, without coordinate-token generation, a verifier, or a zoom-in pass.

4 Experiments

We evaluate Seg2Click along three complementary axes: grounding accuracy across general and professional interfaces, the effectiveness of action masks relative to coordinate generation and patch

Table 2: Performance comparison on ScreenSpot.

			Mobile		Desktop		Web		
Method	Backbone VLM	Venue / Year	Text	Icon	Text	Icon	Text	Icon	Avg
CogAgent	CogVLM-17B	CVPR 2024	67.0	24.0	74.2	20.0	70.4	28.6	47.4
Magma	Llama-3-8B	CVPR 2025	60.4	58.5	75.3	52.9	69.1	52.0	60.3
SeeClick	Qwen-VL-Chat	ACL 2024	78.0	52.0	72.2	30.0	55.7	32.5	53.4
UGround-V1	Qwen2-VL-2B	ICLR 2025	89.4	72.0	88.7	65.7	81.3	68.9	77.7
UI-TARS	Qwen2-VL-2B	arXiv 2025	93.0	75.5	90.7	68.6	84.3	74.8	82.3
GUI-Actor	Qwen2-VL-2B	NeurIPS 2025	93.0	79.9	88.1	78.6	90.9	84.0	86.5
GUI-Actor + V.	Qwen2-VL-2B	NeurIPS 2025	93.8	81.2	89.7	80.7	91.3	80.6	86.9
Qwen2-VL	Qwen2-VL-7B	arXiv 2024	75.5	60.7	76.3	54.3	35.2	25.7	55.3
Aguvis-G	Qwen2-VL-7B	ICML 2025	88.3	78.2	88.1	70.7	85.7	74.8	81.8
OS-Atlas	Qwen2-VL-7B	ICLR 2025	93.0	72.9	91.8	62.9	90.9	74.3	82.5
Aguvis	Qwen2-VL-7B	ICML 2025	95.6	77.7	93.8	67.1	88.3	75.2	84.4
UGround-V1	Qwen2-VL-7B	ICLR 2025	93.0	79.9	93.8	76.4	90.9	84.0	86.3
UI-TARS	Qwen2-VL-7B	arXiv 2025	94.5	85.2	95.9	85.7	90.0	83.5	89.5
GUI-Actor	Qwen2-VL-7B	NeurIPS 2025	94.9	82.1	91.8	80.0	91.3	85.4	88.3
GUI-Actor + V.	Qwen2-VL-7B	NeurIPS 2025	96.0	83.0	93.8	82.1	92.2	87.4	89.7
UI-TARS	Qwen2-VL-72B	arXiv 2025	94.9	82.5	89.7	88.6	88.7	85.0	88.4
Aguvis	Qwen2-VL-72B	ICML 2025	94.5	85.2	95.4	77.9	91.3	85.9	89.2
UGround-V1	Qwen2-VL-72B	ICLR 2025	94.1	83.4	94.9	85.7	90.4	**87.9**	89.4
Seg2Click	Qwen2.5-VL-3B	Ours	96.3	82.5	95.9	82.1	92.6	81.1	89.1
Seg2Click	Qwen3.5-0.8B	Ours	96.0	79.9	93.8	82.9	92.2	76.2	87.4
Seg2Click	Qwen3.5-2B	Ours	94.5	83.0	94.3	90.7	93.0	82.5	89.8
Seg2Click	Qwen3.5-4B	Ours	**96.7**	85.2	**97.4**	91.4	90.9	87.4	91.6
Seg2Click	Qwen3.5-9B	Ours	96.3	**88.2**	95.9	**92.9**	**93.5**	85.0	**92.1**

scoring, and the contribution of the feature fusion and spatial decoder. Experiments cover five backbone configurations from two model families, using a common mask-to-click procedure throughout.

4.1 Experimental Setup

Training. We instantiate Seg2Click with Qwen3.5-0.8B, 2B, 4B, and 9B, and Qwen2.5-VL-3B (Bai et al., 2025; Qwen Team, 2026a;b;c;d). All models use a 1,024-channel decoder and are trained for one epoch on the same grounding data as GUI-Actor, comprising approximately 1M screenshots from six public GUI datasets (Wu et al., 2025a). We use LoRA with rank 16, an effective batch size of 32 across eight GPUs, and fused AdamW with a learning rate of 10^{-4} and cosine decay. Appendix A provides the complete training configuration and trainable parameter counts.

Benchmarks and comparisons. ScreenSpot and ScreenSpot-v2 each contain 1,272 examples spanning mobile, desktop, and web interfaces; ScreenSpot-Pro contains 1,581 examples from professional applications(Cheng et al., 2024; Wu et al., 2025b; Li et al., 2025). We report Hit@1: a prediction is correct when its single click lies inside the annotated target box, including its boundary. Overall accuracy is computed over all examples. Tables 1-3 report the performance of Seg2Click compared to existing advanced methods on the datasets ScreenSpot-Pro, ScreenSpot, and ScreenSpot-v2. Table 6 complements these comparisons with a matched-training coordinate baseline and a GUI-Actor reproduction on Qwen3.5-0.8B. Every Seg2Click evaluation uses the same mask threshold and largest-component centroid rule, with one mask-producing prediction per input. Measured inference costs are reported in Appendix C.

Table 3: Performance comparison on ScreenSpot-v2. † denotes reported evaluations of official checkpoints.

			Mobile		Desktop		Web		
Method	Backbone VLM	Venue / Year	Text	Icon	Text	Icon	Text	Icon	Avg
OS-Atlas	InternVL2-4B	ICLR 2025	87.2	59.7	72.7	46.4	85.9	63.1	71.9
Magma	Llama-3-8B	CVPR 2025	62.8	53.4	80.0	57.9	67.5	47.3	61.5
SeeClick	Qwen-VL-Chat	ACL 2024	78.4	50.7	70.1	29.3	55.2	32.5	55.1
UI-TARS	Qwen2-VL-2B	arXiv 2025	95.2	79.1	90.7	68.6	87.2	78.3	84.7
GUI-Actor	Qwen2-VL-2B	NeurIPS 2025	95.0	82.2	92.2	81.8	92.9	82.7	88.6
GUI-Actor + V.	Qwen2-VL-2B	NeurIPS 2025	95.9	81.5	94.3	82.9	93.6	82.8	89.3
OS-Atlas	Qwen2-VL-7B	ICLR 2025	95.2	75.8	90.7	63.6	90.6	77.3	84.1
Aguvis†	Qwen2-VL-7B	ICML 2025	95.5	77.3	95.4	77.9	91.0	72.4	86.0
UGround-V1†	Qwen2-VL-7B	ICLR 2025	95.0	83.3	95.0	77.8	92.1	77.2	87.6
UI-TARS	Qwen2-VL-7B	arXiv 2025	96.9	89.1	95.4	85.0	93.6	85.2	91.6
GUI-Actor	Qwen2-VL-7B	NeurIPS 2025	96.5	84.3	91.7	84.1	93.9	82.3	89.5
GUI-Actor + V.	Qwen2-VL-7B	NeurIPS 2025	97.2	84.8	94.3	85.0	94.0	85.2	90.9
UI-TARS	Qwen2-VL-72B	arXiv 2025	94.8	86.3	91.2	87.9	91.5	**87.7**	90.3
Qwen2.5-VL	Qwen2.5-VL-3B	arXiv 2025	93.4	73.5	88.1	58.6	88.0	71.4	80.9
Jedi	Qwen2.5-VL-3B	NeurIPS 2025	96.6	81.5	96.9	78.6	88.5	83.7	88.6
GUI-Actor	Qwen2.5-VL-3B	NeurIPS 2025	97.6	83.4	96.9	83.6	94.0	85.7	91.0
GUI-Actor + V.	Qwen2.5-VL-3B	NeurIPS 2025	98.3	85.3	96.9	87.9	95.3	86.7	92.4
Seg2Click	Qwen2.5-VL-3B	Ours	98.8	85.3	96.7	84.1	95.7	82.3	91.3
Qwen2.5-VL	Qwen2.5-VL-7B	arXiv 2025	97.6	87.2	90.2	74.2	93.2	81.3	88.8
Jedi	Qwen2.5-VL-7B	NeurIPS 2025	96.9	87.2	95.9	87.9	94.4	84.2	91.7
GUI-Actor	Qwen2.5-VL-7B	NeurIPS 2025	97.6	88.2	96.9	85.7	93.2	86.7	92.1
GUI-Actor + V.	Qwen2.5-VL-7B	NeurIPS 2025	96.9	89.6	97.4	86.4	95.7	84.7	92.5
Seg2Click	Qwen3.5-0.8B	Ours	97.7	81.7	94.4	87.3	93.6	81.9	89.9
Seg2Click	Qwen3.5-2B	Ours	95.7	86.9	95.0	92.1	96.1	82.7	91.6
Seg2Click	Qwen3.5-4B	Ours	97.7	88.5	**97.8**	92.9	93.9	87.3	93.1
Seg2Click	Qwen3.5-9B	Ours	**99.2**	**90.1**	96.7	**95.2**	**96.4**	86.5	**94.1**

4.2 Benchmark Results and Analysis

Overall grounding performance. Seg2Click-9B achieves 92.06%, 94.10%, and 55.22% on ScreenSpot, ScreenSpot-v2, and ScreenSpot-Pro, respectively, leading the systems in Tables 1–3 by 2.36, 1.60, and 7.52 percentage points over the strongest published baselines. The largest margin occurs on ScreenSpot-Pro, highlighting its effectiveness in professional interfaces with small controls and dense layouts. All evaluations use the same decoder and deterministic click extraction, without a verifier or zoom-in pass.

Comparison on a shared backbone. With Qwen2.5-VL-3B, Seg2Click reaches 47.00% on ScreenSpot-Pro, exceeding published GUI-Actor with the same base checkpoint by 4.80 points and its verifier-assisted variant by 1.10 points. Gains over GUI-Actor without a verifier span all six application domains, ranging from 3.5 to 6.6 points. Together with the 0.8B output-representation study in Section 4.3.3, these results support the effectiveness of the action-mask formulation for professional GUI grounding.

Localizing desktop icons and professional controls. Seg2Click-9B achieves 92.9% and 95.2% on desktop icons in ScreenSpot and ScreenSpot-v2, surpassing the strongest published baselines by 4.3 and 7.3 points. Seg2Click-4B also leads the corresponding desktop-text categories at 97.4% and 97.8%. On ScreenSpot-Pro, the Qwen3.5 variants collectively lead all six application domains: 9B

leads Creative, CAD, Office, and OS, while 2B leads Development and Scientific. These results demonstrate broad applicability across target types and professional interfaces.

Scaling benefits icons and small targets. With the training recipe fixed, scaling Qwen3.5 from 0.8B to 9B improves ScreenSpot-Pro accuracy by 10.94 points, corresponding to 173 additional correct clicks. Gains are larger for icons than text (16.72 vs. 7.37 points). Accuracy increases by 13.90 points for targets occupying at most 0.01% of the screenshot and by 11.77 points for those occupying 0.01%–0.1%, compared with 4.62 points for larger targets. These results indicate that backbone scaling particularly benefits icons and small targets; Appendix B.1 provides the full breakdown.

4.3 Ablation Study

We use Qwen3.5-0.8B to examine feature sources, fusion strategies, and output representations. Feature and fusion comparisons retain the same mask objective and decoder width. The coordinate-generation comparison shares the initialization, training data, LoRA configuration, optimizer arguments, and one-epoch schedule with Seg2Click. Appendix B.2 additionally evaluates decoder capacity.

4.3.1 Complementary Vision and Language-Model Features

Table 4 shows that residual fusion achieves the highest accuracy among the evaluated feature configurations on all three benchmarks. It improves over final language-model image states by 0.55, 0.47, and 0.76 points on ScreenSpot, ScreenSpot-v2, and ScreenSpot-Pro. On ScreenSpot-Pro, the fused representation reaches 44.28%, compared with 43.52% for language-model states and 42.38% for post-merge vision states. Improving over either stream alone supports their complementarity: direct vision features supply visual evidence, while final language-model states provide contextualized image representations. Combining the aligned streams gives the spatial decoder access to both sources when predicting the instruction-specific action region.

Table 4: Feature sources at fixed Qwen3.5-0.8B and decoder width $C = 1024$. Combining LLM and post-merge vision states improves all three benchmarks, reaching 44.28% on ScreenSpot-Pro. Accuracy is in percent (↑). Best and second-best distinct values are bold and underlined, respectively; ties use the displayed precision.

Feature source	SS	SS-v2	SS-Pro
Pre-merge only	81.68	84.43	22.01
Post-merge only	86.32	<u>89.54</u>	42.38
LLM only, $C = 1024$	<u>86.87</u>	89.47	<u>43.52</u>
Residual concat–MLP	**87.42**	**89.94**	**44.28**

The comparison between vision features also highlights the value of the backbone's learned spatial aggregation. Post-merge features reach 42.38% on ScreenSpot-Pro, improving over pre-merge features by 20.37 points, or 322 correct clicks. This finding supports a two-stage design: use the backbone's aggregated representations to identify relevant visual content, then learn distinct spatial outputs within each token footprint. Seg2Click couples these informative features with dense decoding to recover the localization granularity needed for click selection.

4.3.2 Residual Fusion and Decoder Capacity

Table 5 compares residual concat–MLP fusion with a scalar-gated residual connection. Concat–MLP reaches 87.42% and 89.94% on ScreenSpot and ScreenSpot-v2, improving over scalar gating by 1.81 and 1.26 points. Both fusion strategies improve ScreenSpot-Pro over the unfused 43.52% baseline, reaching 44.28% with concat–MLP and 44.53% with scalar gating. Concat–MLP provides gains over the unfused representation on every benchmark and the strongest results on the two standard benchmarks, motivating its use in the final architecture. Its joint transformation of the two feature streams allows the visual correction to depend on their combined content, while the residual path preserves the language-model representation.

Table 5: Feature fusion at fixed Qwen3.5-0.8B and $C = 1024$. Residual concat–MLP improves over unfused LLM states on all three benchmarks. Accuracy is in percent (↑). Best and second-best distinct values are bold and underlined, respectively; ties use the displayed precision.

Fusion	SS	SS-v2	SS-Pro
None (LLM only)	<u>86.87</u>	<u>89.47</u>	43.52
Scalar-gated residual	85.61	88.68	**44.53**
Residual concat–MLP	**87.42**	**89.94**	<u>44.28</u>

Decoder capacity interacts with feature fusion. With language-model image states alone, increasing the width from 128 to 1,024 channels improves the three benchmarks by 0.39, 0.16, and 1.39 points. With concat–MLP fusion, $C = 256$ matches $C = 1024$ on ScreenSpot-Pro, while $C = 1024$ yields the highest ScreenSpot and ScreenSpot-v2 scores (Appendix B.2) and remains the default configuration.

4.3.3 Action Masks Improve Grounding Accuracy

Table 6 directly evaluates the output representation on Qwen3.5-0.8B. Under matched training settings, Seg2Click improves over coordinate generation by 0.94, 1.02, and 1.52 points across the three benchmarks, adding 12, 13, and 24 correct clicks. The desktop subsets contribute 13 additional correct predictions on each standard benchmark, corresponding to gains of 3.89 and 4.25 points. On ScreenSpot-Pro, text-target accuracy increases from 58.85% to 61.51%. These results support learning the region of valid click locations before selecting the final point: box-derived supervision supplies a spatial target that is directly aligned with the box-hit evaluation criterion.

Table 6: Seg2Click leads the compared output representations on all three benchmarks with Qwen3.5-0.8B. GUI-Actor (Wu et al., 2025a) is reproduced on this backbone. Appendix A reports parameter counts. Accuracy is in percent (↑). Best and second-best distinct values are bold and underlined, respectively; ties use the displayed precision.

Output representation	Backbone VLM	SS	SS-v2	SS-Pro
Direct coordinates	Qwen3.5-0.8B	86.48	88.92	<u>42.76</u>
GUI-Actor	Qwen3.5-0.8B	<u>87.26</u>	<u>89.47</u>	39.41
Seg2Click	Qwen3.5-0.8B	**87.42**	**89.94**	**44.28**

Relative to reproduced GUI-Actor on the same backbone, Seg2Click improves by 0.16, 0.47, and 4.87 points, with 77 additional correct predictions on ScreenSpot-Pro. Office, CAD, and Scientific applications contribute 20, 19, and 19 net additional hits, respectively, accounting for 58 of the 77-hit gain. This concentration across professional domains is consistent with the benefit of sub-token action prediction, which learns separate scores for locations within a visual-token footprint. Taken together, the coordinate and patch-scoring comparisons provide complementary evidence for Seg2Click's central design: represent valid click regions explicitly and learn their spatial structure at finer granularity than the visual-token grid.

5 Conclusion

Seg2Click formulates GUI grounding as box-supervised sub-token action-field prediction, learning fine-grained valid-click regions before deterministic point selection. Matched coordinate comparisons demonstrate accuracy gains across all three benchmarks, while the same-backbone patch-scoring comparison adds 77 correct ScreenSpot-Pro clicks. Evaluation across two model families reaches 55.22% on ScreenSpot-Pro at 9B and 47.00% at 3B. These results establish learned action masks as an effective output representation for precise GUI grounding.

Building on these grounding results, future work will explore confidence-aware click selection and integration into interactive GUI agents, informed by the qualitative analysis in Appendix D.

AI USE STATEMENT

Generative AI tools assisted with language polishing and presentation refinement during manuscript writing. The authors reviewed the final manuscript and take responsibility for its content.

ETHICS STATEMENT

This work evaluates grounding on existing GUI datasets. Screenshots can contain personal or proprietary content, and redistribution must respect source licenses and privacy constraints. Grounding supports accessibility and authorized automation. Responsible deployment requires user authorization and action-appropriate safeguards to prevent unintended interface actions.

REPRODUCIBILITY STATEMENT

Section 3 specifies the complete decoder and click rule, including mask-to-click decoding in Section 3.4. Section 4.1 and Appendices A and C document training settings, data counts, evaluation protocols, numerical precision, and inference settings.

REFERENCES

AgentSea. Wave-UI. Hugging Face dataset, 2024. URL https://huggingface.co/datasets/agentsea/wave-ui.

Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, and Blaise Agüera y Arcas. UIBert: Learning generic multimodal representations for UI understanding. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pp. 1705–1712. International Joint Conferences on Artificial Intelligence Organization, 2021. doi: 10.24963/ijcai.2021/235. URL https://www.ijcai.org/proceedings/2021/235.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025. URL https://arxiv.org/abs/2502.13923.

Yuxiang Chai, Siyuan Huang, Yazhe Niu, Han Xiao, Liang Liu, Guozhi Wang, Dingyu Zhang, Shuai Ren, and Hongsheng Li. AMEX: Android multi-annotation expo dataset for mobile GUI agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 2138–2156. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.110. URL https://aclanthology.org/2025.findings-acl.110/.

Keqin Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal LLM's referential dialogue magic. *arXiv preprint arXiv:2306.15195*, 2023. doi: 10.48550/arXiv.2306.15195. URL https://arxiv.org/abs/2306.15195.

Wentong Chen, Junbo Cui, Jinyi Hu, Yujia Qin, Junjie Fang, Yue Zhao, Chongyi Wang, Jun Liu, Guirong Chen, Yupeng Huo, Yuan Yao, Yankai Lin, Zhiyuan Liu, and Maosong Sun. GUICourse: From general vision language model to versatile GUI agent. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21936–21959. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.1065. URL https://aclanthology.org/2025.acl-long.1065/.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, 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)*, pp. 9313–9332. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.505. URL https://aclanthology.org/2024.acl-long.505/.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2Web: Towards a generalist agent for the web. In *Advances in*

Neural Information Processing Systems, volume 36, pp. 28091–28114, 2023. doi: 10.52202/075280-1220. URL https://papers.neurips.cc/paper_files/paper/2023/hash/5950bf290a1570ea401bf98882128160-Abstract-Datasets_and_Benchmarks.html.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for GUI agents. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/4ca0e369689dadb25a5345ba9755ad6f-Abstract-Conference.html.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. WebVoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6864–6890. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.acl-long.371. URL https://aclanthology.org/2024.acl-long.371/.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Xin Lai, Zhuotao Tian, Yukang Chen, Yanwei Li, Yuhui Yuan, Shu Liu, and Jiaya Jia. LISA: Reasoning segmentation via large language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9579–9589, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Lai_LISA_Reasoning_Segmentation_via_Large_Language_Model_CVPR_2024_paper.html.

Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. ScreenSpot-Pro: GUI grounding for professional high-resolution computer use. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 8778–8786. Association for Computing Machinery, 2025. doi: 10.1145/3746027.3755688. URL https://doi.org/10.1145/3746027.3755688.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on UI control agents. In *Advances in Neural Information Processing Systems*, volume 37, pp. 92130–92154. Curran Associates, Inc., 2024. doi: 10.52202/079017-2925. URL https://papers.nips.cc/paper_files/paper/2024/hash/a79f3ef3b445fd4659f44648f7ea8ffd-Abstract-Datasets_and_Benchmarks_Track.html.

Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Stan Weixian Lei, Lijuan Wang, and Mike Zheng Shou. ShowUI: One Vision-Language-Action Model for GUI Visual Agent. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19498–19508, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Lin_ShowUI_One_Vision-Language-Action_Model_for_GUI_Visual_Agent_CVPR_2025_paper.html.

Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. OmniParser for pure vision based GUI agent. *arXiv preprint arXiv:2408.00203*, 2024. doi: 10.48550/arXiv.2408.00203. URL https://arxiv.org/abs/2408.00203.

Pawel Pawlowski, Krystian Zawistowski, Wojciech Lapacz, Adam Wiacek, Marcin Skorupa, Sebastien Postansque, and Jakub Hoscilowicz. TinyClick: Single-turn agent for empowering GUI automation. In *Interspeech 2025*, pp. 3035–3039, 2025. doi: 10.21437/Interspeech.2025-176. URL https://www.isca-archive.org/interspeech_2025/pawlowski25_interspeech.html.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11671. URL https://ojs.aaai.org/index.php/AAAI/article/view/11671.

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

Qwen Team. Qwen3.5-0.8B. Official model card, 2026a. URL https://huggingface.co/Qwen/Qwen3.5-0.8B.

Qwen Team. Qwen3.5-4B. Official model card, 2026b. URL https://huggingface.co/Qwen/Qwen3.5-4B.

Qwen Team. Qwen3.5-9B. Official model card, 2026c. URL https://huggingface.co/Qwen/Qwen3.5-9B.

Qwen Team. Qwen3.5-2B. Official model card, 2026d. URL https://huggingface.co/Qwen/Qwen3.5-2B.

Chris Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. AndroidWorld: A dynamic benchmarking environment for autonomous agents. In *International Conference on Learning Representations*, pp. 406–441, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/01a83bc2f2732a58e6aa731e659e7101-Abstract-Conference.html.

Zhongwei Ren, Zhicheng Huang, Yunchao Wei, Yao Zhao, Dongmei Fu, Jiashi Feng, and Xiaojie Jin. PixelLM: Pixel reasoning with large multimodal model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26374–26383, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Ren_PixelLM_Pixel_Reasoning_with_Large_Multimodal_Model_CVPR_2024_paper.html.

Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. doi: 10.1109/CVPR.2016.207. URL https://arxiv.org/abs/1609.05158.

Zhi Tian, Chunhua Shen, Xinlong Wang, and Hao Chen. BoxInst: High-performance instance segmentation with box annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5443–5452, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Tian_BoxInst_High-Performance_Instance_Segmentation_With_Box_Annotations_CVPR_2021_paper.html.

Jingchao Wang, Zhijian Wu, Dingjiang Huang, Yefeng Zheng, and Hong Wang. Unlocking the potential of MLLMs in referring expression segmentation via a light-weight mask decoder. *arXiv preprint arXiv:2508.04107*, 2025. URL https://arxiv.org/abs/2508.04107.

Qianhui Wu, Kanzhi Cheng, Rui Yang, Chaoyun Zhang, Jianwei Yang, Huiqiang Jiang, Jian Mu, Baolin Peng, Bo Qiao, Reuben Tan, Si Qin, Lars Liden, Qingwei Lin, Huan Zhang, Tong Zhang, Jianbing Zhang, Dongmei Zhang, and Jianfeng Gao. GUI-Actor: Coordinate-free visual grounding for GUI agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025a. doi: 10.52202/085713-0508. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/16130af940e9dabb43c726119bd3b42e-Abstract-Conference.html.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. OS-ATLAS: A foundation action model for generalist GUI agents. In *International Conference on Learning Representations*, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/0faa4bc5f522076947a030273629d4fe-Abstract-Conference.html.

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. In *Advances in Neural Information Processing Systems*, volume 37, pp. 52040–52094, 2024. doi: 10.52202/079017-1650. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5d413e48f84dc61244b6be550f1cd8f5-Abstract-Datasets_and_Benchmarks_Track.html.

Tianbao Xie, Jiaqi Deng, Xiaochuan Li, Junlin Yang, Haoyuan Wu, Jixuan Chen, Wenjing Hu, Xinyuan Wang, Yuhui Xu, Zekun Wang, Yiheng Xu, Junli Wang, Doyen Sahoo, Tao Yu, and Caiming Xiong. Scaling computer-use grounding via user interface decomposition and synthesis. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0817. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/22c868099177ee278eb7baccec649f35-Abstract-Datasets_and_Benchmarks_Track.html.

Hai-Ming Xu, Qi Chen, Lei Wang, and Lingqiao Liu. Attention-driven GUI grounding: Leveraging pretrained multimodal large language models without fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 8851–8859, 2025a. doi: 10.1609/aaai.v39i8.32957. URL https://ojs.aaai.org/index.php/AAAI/article/view/32957.

Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. Aguvis: Unified Pure Vision Agents for Autonomous GUI Interaction. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 69772–69805. PMLR, 2025b. URL https://proceedings.mlr.press/v267/xu25ae.html.

Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. Aria-UI: Visual grounding for GUI instructions. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 22418–22433. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.1152. URL https://aclanthology.org/2025.findings-acl.1152/.

Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, and Zhe Gan. Ferret-UI: Grounded mobile UI understanding with multimodal LLMs. In *Computer Vision – ECCV 2024*, pp. 240–255. Springer, 2024. doi: 10.1007/978-3-031-73039-9_14. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/papers/08095.pdf.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, pp. 15585–15606, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/4410c0711e9154a7a2d26f9b3816d1ef-Abstract-Conference.html.

A Training and Inference Details

All five main Seg2Click models are initialized from official pretrained checkpoints and trained for one epoch using LoRA with rank 16, alpha 32, dropout 0.05, and all-linear targeting. The backbone operates in BF16, while the fusion and mask decoder operate in FP32. All models use a 1,024-channel spatial decoder. Qwen3.5 uses a patch size of 16 pixels and a pixel-shuffle factor of 32, whereas Qwen2.5-VL uses a patch size of 14 pixels and a pixel-shuffle factor of 28. Query conditioning, BCE/Dice loss weights, and centroid-based click extraction are identical across runs. The Qwen2.5-VL run additionally uses FlashAttention-2 in both the vision and language modules, together with gradient checkpointing in the language blocks. Each model is evaluated using its final checkpoint after one epoch. Table 7 summarizes the model configurations.

We use the same training data as GUI-Actor, comprising the six sources listed in Table 8. GUI-Env and GUI-Act originate from GUICourse (Chen et al., 2025), while AMEX and AndroidControl provide mobile-interface annotations and demonstrations (Chai et al., 2025; Li et al., 2024). For Wave-UI (AgentSea, 2024), we use the processed subset of 52,529 records distributed with GUI-Actor. According to GUI-Actor's data documentation, this released subset excludes samples overlapping the test set.[1] Wave-UI remains subject to the license terms of its constituent sources.[2] Across all five runs, we retain at most ten non-system messages per record and compute the training loss by first averaging over target queries within each record and then averaging over records, giving each record equal weight.

Table 7: Training configurations across backbones. Parameter columns report LoRA adapters, the fusion/mask head, and their total; all runs use eight GPUs, effective batch 32 and 39,226 updates.

Backbone	Hidden d	Batch/GPU	Accum.	LoRA params	Head params	Total params
Qwen3.5-0.8B	1024	4	1	13,345,792	27,286,528	40,632,320
Qwen3.5-2B	2048	2	2	23,340,032	43,028,480	66,368,512
Qwen2.5-VL-3B	2048	1	4	41,363,456	42,782,480	84,145,936
Qwen3.5-4B	2560	1	4	38,993,920	54,045,184	93,039,104
Qwen3.5-9B	4096	1	4	51,265,024	99,678,208	150,943,232

Table 8: The shared training mixture contains 1,255,208 records from six sources. Each training record can pair a screenshot with one or more queries.

Source	Records
UGround	756,451
GUI-Env	262,445
GUI-Act	42,545
AMEX	101,537
AndroidControl	39,701
Wave-UI	52,529
Total	1,255,208

Optimization and numerical precision. The per-device batches are 4/2/1/1/1 and accumulation factors are 1/2/4/4/4 for Qwen3.5-0.8B, Qwen3.5-2B, Qwen2.5-VL-3B, Qwen3.5-4B, and Qwen3.5-9B, respectively. We use zero weight decay, no warmup, a sequence limit of 24,576, and processor pixel bounds of 3,136 and 5,720,064. LoRA targets linear layers in both the vision and language blocks while base weights remain frozen. Decoder parameters are stored in FP32; the spatial mask head and its losses also execute in FP32, with a BF16 backbone. New action-token embeddings use mean initialization and remain frozen during LoRA training.

Trainable parameters and grounding performance. Table 9 compares adapter and head capacity alongside grounding accuracy for the 0.8B models. The coordinate comparison matches backbone initialization, training data, adapter rank, and optimization. The head count includes learned feature fusion where enabled.

[1] https://huggingface.co/datasets/cckevinn/GUI-Actor-Data

[2] https://huggingface.co/datasets/agentsea/wave-ui/blob/main/LICENSE.md

Table 9: Trainable parameters and grounding accuracy for Qwen3.5-0.8B output architectures. Head counts include fusion where enabled. LLM-only rows use no direct vision fusion. Accuracy is in percent (↑); best and second-best distinct available values are bold and underlined.

	Trainable parameters			Accuracy (%)		
Configuration	LoRA	Head	Total	SS	SS-v2	SS-Pro
Direct coordinates	13,345,792	0	13,345,792	86.48	88.92	42.76
GUI-Actor	13,345,792	8,398,848	21,744,640	<u>87.26</u>	89.47	39.41
LLM only, $C = 128$	13,345,792	825,472	14,171,264	86.48	89.31	42.13
LLM only, $C = 1024$	13,345,792	23,083,008	36,428,800	86.87	89.47	43.52
Seg2Click, fused $C = 256$	13,345,792	6,439,168	19,784,960	86.56	<u>89.86</u>	**44.28**
Seg2Click, fused $C = 512$	13,345,792	11,028,992	24,374,784	87.19	89.54	<u>44.09</u>
Seg2Click, fused $C = 1024$	13,345,792	27,286,528	40,632,320	**87.42**	**89.94**	**44.28**

The fused $C = 256$ model uses 9.0% fewer trainable parameters than GUI-Actor (19.78M versus 21.74M), with ScreenSpot-Pro accuracies of 44.28% versus 39.41%. Appendix B.2 analyzes the decoder-width trade-off. These results describe the accuracy–capacity trade-off of the evaluated architectures; their differing parameter budgets do not isolate output representation from added capacity.

Spatial decoder implementation. The two 3×3 convolutional blocks use stride one, padding one, group normalization with 32 groups, and GELU. For grid position (r, c) and within-cell offset (a, b), pixel shuffle in Eq. (3) uses

$$L_j[sr + a, sc + b] = \mathbf{U}_j[a\, s + b, r, c], \qquad 0 \le a, b < s. \tag{5}$$

Equation (5) assigns the s^2 channels to distinct processed-image pixels without interpolation.

Target rasterization and loss definitions. For normalized target box b_j, the cell-overlap rule is

$$\begin{aligned} R_{yx} &= \left[\frac{x}{W'}, \frac{x+1}{W'}\right) \times \left[\frac{y}{H'}, \frac{y+1}{H'}\right), \\ Y_j(y, x) &= \mathbf{1}[\mathrm{area}(R_{yx} \cap b_j) > 0]\,. \end{aligned} \tag{6}$$

Equation (6) defines the box-derived action target, which directly supervises the valid click region. Training boxes must have finite coordinates in $[0, 1]$ with $x_{\min} < x_{\max}$ and $y_{\min} < y_{\max}$; output grid dimensions must be positive. The positive-area overlap rule then yields at least one positive processed-pixel cell, including for subpixel boxes. Invalid boxes and anomalous empty rasterizations raise an error. This training-target check is separate from the inference-time empty-prediction fallback below. We define the losses used in Eq. (4) as

$$\begin{aligned} \mathcal{L}^{(j)}_{\mathrm{BCE}} &= -\frac{1}{H'W'} \sum_{y,x} \left[Y_j(y, x) \log P_j(y, x) + (1 - Y_j(y, x)) \log(1 - P_j(y, x))\right], \\ \mathcal{L}^{(j)}_{\mathrm{Dice}} &= 1 - \frac{2 \sum_{y,x} P_j(y, x) Y_j(y, x) + \epsilon}{\sum_{y,x} P_j(y, x) + \sum_{y,x} Y_j(y, x) + \epsilon}, \qquad \epsilon = 1. \end{aligned} \tag{7}$$

In Eq. (7), binary cross-entropy is implemented on logits for numerical stability.

Empty-mask fallback. When thresholding produces no foreground component, we compute the probability-weighted centroid

$$\hat{\mathbf{c}}_j = \frac{\sum_{y,x} P_j(y, x) \mathbf{c}_{yx}}{\sum_{y,x} P_j(y, x)}. \tag{8}$$

Here, $\mathbf{c}_{yx} = ((x + \frac{1}{2})/W', (y + \frac{1}{2})/H')$ denotes the normalized pixel center. Equation (8) weights every pixel center by its predicted probability; a zero probability sum returns $(0.5, 0.5)$. In the batch-size-one Qwen2.5-VL-3B evaluations in Appendix C, empty-mask fallback occurs in 0/1,272 ScreenSpot examples, 0/1,272 ScreenSpot-v2 examples, and 1/1,581 ScreenSpot-Pro examples. The single fallback uses the probability-weighted centroid.

Action prompt. The screenshot and instruction follow the grounding system message. The fixed assistant suffix in the repository's action template is `pyautogui.click(<|pointer_start|><|pointer_pad|><|pointer_end|>)`. The `<|pointer_pad|>` state is the action query. At inference there is one instruction per example and one click per prediction, even when the numerical batch contains two examples.

Coordinate-generation baseline. The matched training comparison appears in Section 4.3.3. The coordinate baseline preserves the source action coordinates supplied in the processed training records, already normalized to $[0, 1]^2$ against the original screenshot. It formats four-decimal `pyautogui.click(x=...,y=...)` targets using `ROUND_HALF_UP`, without recomputing points from box centers. It trains the adapters with cross-entropy averaged over supervised assistant tokens, including the backbone's native end-of-turn token `<|im_end|>`, and uses no spatial head. At inference, greedy decoding generates at most 64 tokens; a strict parser accepts finite in-range coordinates without clipping or repair. All 4,125 coordinate predictions are valid and terminate with the native end-of-turn token.

Algorithm 1 Seg2Click inference for one screenshot and instruction

Input: screenshot I of size $H \times W$, instruction u, trained backbone and head.
Output: one original-image click $(W\hat{x}, H\hat{y})$.

1 Preprocess $I' = \Pi(I)$; append the fixed action placeholder after u.
2 Run the backbone; extract aligned $\{\mathbf{z}_i, \mathbf{v}_i\}$ and action query $\mathbf{q}$.
3 Fuse features by Eq. (1); condition the grid by Eq. (2).
4 Apply Eq. (3) to obtain $L \in \mathbb{R}^{H' \times W'}$; set $P = \sigma(L)$.
5 Find four-connected components of $P \geq 0.5$ and select the largest; break ties by smallest row-major index.
6 If a component exists, use its unweighted centroid of normalized pixel centers; otherwise use Eq. (8), returning $(0.5, 0.5)$ for a zero sum.
7 Return $(W\hat{x}, H\hat{y})$.

Table 10: Notation links the backbone grid to the dense action output.

Symbol	Meaning
H, W; H', W'	Original and processed screenshot dimensions
$p, m, s = pm$	Patch size, merge factor, processed-pixel stride
$h, w, N = hw$	Post-merge grid dimensions and token count
d, C	Backbone hidden size and decoder width
$\mathbf{z}_i, \mathbf{v}_i, \mathbf{f}_i$	Direct vision, final image, and fused features
$\mathbf{q}_j$	Final action-query hidden state for query j
L_j, P_j, Y_j	Dense logits, probabilities, and box-derived target
$\mathcal{C}^*, \hat{\mathbf{c}}_j$	Selected component and normalized click

Table 10 collects the symbols used in the method. Algorithm 1 fixes every inference choice independently of benchmark or backbone.

B Additional Experimental Analysis

This appendix reports the backbone, target-size, and decoder-capacity analyses complementing the main comparisons in Section 4.2 and the ablations in Section 4.3.

B.1 Grounding Performance and Backbone Scaling

Target-type and target-area analysis. Table 11 reports the ScreenSpot-Pro breakdown for 977 text and 604 icon targets. Target area is $a = (x_{\max} - x_{\min})(y_{\max} - y_{\min})$, using box coordinates normalized by the original screenshot dimensions. The groups $a \leq 0.01\%$, $0.01\% < a \leq 0.1\%$, and $a > 0.1\%$ contain 259, 1,062, and 260 targets, respectively.

Scaling from 2B to 4B redistributes gains across target sizes: the intermediate-area group adds 21 correct predictions, while the smallest and largest groups lose 6 and 9, respectively, leaving a net gain

Table 11: Seg2Click accuracy on ScreenSpot-Pro (%, ↑) by target type and normalized target-box area across backbones. Best and second-best distinct values are bold and underlined, respectively; ties use the displayed precision.

	Target type			Normalized target area a		
Backbone VLM	Text	Icon	Overall	$a \leq 0.01\%$	$0.01\% < a \leq 0.1\%$	$a > 0.1\%$
Qwen3.5-0.8B	61.51	16.39	44.28	9.27	46.33	70.77
Qwen3.5-2B	<u>65.20</u>	25.00	49.84	<u>17.37</u>	51.51	**75.38**
Qwen2.5-VL-3B	62.74	21.52	47.00	11.20	49.81	71.15
Qwen3.5-4B	<u>65.20</u>	<u>25.99</u>	<u>50.22</u>	15.06	<u>53.48</u>	<u>71.92</u>
Qwen3.5-9B	**68.88**	**33.11**	**55.22**	**23.17**	**58.10**	**75.38**

of six. Text accuracy remains at 65.20%. Thus, a higher overall score does not imply improvement in every target group.

The intermediate-area group contributes 125 of the 173 additional correct predictions from 0.8B to 9B, versus 36 and 12 from the smallest and largest groups. As the most populous group, its contribution reflects both its frequency and improved accuracy.

B.2 Does additional decoder capacity improve grounding?

Table 12: Decoder-width ablation on Qwen3.5-0.8B with residual concat–MLP fusion. Accuracy is in percent (↑). Best and second-best distinct values are bold and underlined, respectively; ties use the displayed precision.

Decoder channels	SS	SS-v2	SS-Pro
256	86.56	<u>89.86</u>	**44.28**
512	<u>87.19</u>	89.54	<u>44.09</u>
1024	**87.42**	**89.94**	**44.28**

Table 12 compares decoder widths of 256, 512, and 1,024 channels with residual concat–MLP fusion on Qwen3.5-0.8B.

The fused $C = 256$ model matches $C = 1024$ at 44.28% on ScreenSpot-Pro and trails it by only one correct prediction on ScreenSpot-v2 (89.86% versus 89.94%), while using 51.3% fewer total trainable parameters (Table 9). The $C = 512$ model reduces total trainable parameters by 40.0%, but its ScreenSpot-v2 and ScreenSpot-Pro scores are below those of $C = 256$. In these runs, accuracy therefore does not improve uniformly with decoder width. The $C = 1024$ configuration gives the highest ScreenSpot and ScreenSpot-v2 accuracy, whereas $C = 256$ offers a smaller parameter budget while retaining the same ScreenSpot-Pro accuracy.

C Measured Inference Cost

The two spatial decoder convolutions require $18hwC^2$ multiply–accumulates per query, and the output projection requires $hwCs^2$; pixel shuffle only rearranges values. Thus spatial refinement operates on $h \times w$, before producing the $H' \times W'$ action field.

Table 13 reports inference cost for Qwen2.5-VL-3B on one NVIDIA H200 with batch size 1. Each run covers the full benchmark: 1,272 examples for ScreenSpot and ScreenSpot-v2, and 1,581 for ScreenSpot-Pro. Mask prediction and click extraction together average 42.49–63.59 ms per example and account for 8.25–11.62% of evaluation-loop wall time.

Wall time covers the evaluation loop, including image and result I/O, and excludes setup and final reporting. Decoder time includes feature fusion and mask prediction; postprocessing extracts connected components and the click centroid on the CPU. Unassigned I/O and loop work account for the difference between wall time and the sum of stages.

Table 13: Inference cost of Qwen2.5-VL-3B on one H200 with batch size 1. Decoder + Post. gives combined mask prediction and click extraction time, its share of wall time, and mean time per example. Peak memory is allocated GPU memory. Derived values use unrounded measurements.

Dataset	Stage time (s)					Decoder + Post.			Peak memory
	Wall	Prep.	Backbone	Decoder	Post.	Time (s)	Share (%)	ms/example	(GiB)
ScreenSpot	472.5	173.3	232.4	7.4	46.7	54.04	11.44	42.49	8.60
ScreenSpot-v2	471.5	173.1	231.3	7.3	47.5	54.80	11.62	43.08	8.48
ScreenSpot-Pro	1218.9	490.5	487.5	13.8	86.7	100.53	8.25	63.59	8.61

D Qualitative Analysis and Limitations

D.1 Strengths in Small-Target Grounding

Figure 3 shows how Seg2Click-4B localizes compact controls in crowded interfaces. The Unreal Engine filter control and Vivado percentage icon measure only 42×30 and 32×30 pixels. In both cases, the predicted component concentrates on the target and its centroid yields a valid click. The code-execution and font-color examples similarly show localization among neighboring toolbar controls. These cases illustrate the fine spatial predictions enabled by action masks, using one prediction on the full screenshot without a verifier or an additional zoom-in pass.

The spatial output also makes localization behavior inspectable: the mask shows where the model responds, while the click shows how the deterministic rule converts that response into an action. This distinction helps diagnose whether a miss arises from a poorly localized region or from converting that region into a click. Enlarged views are for display only.

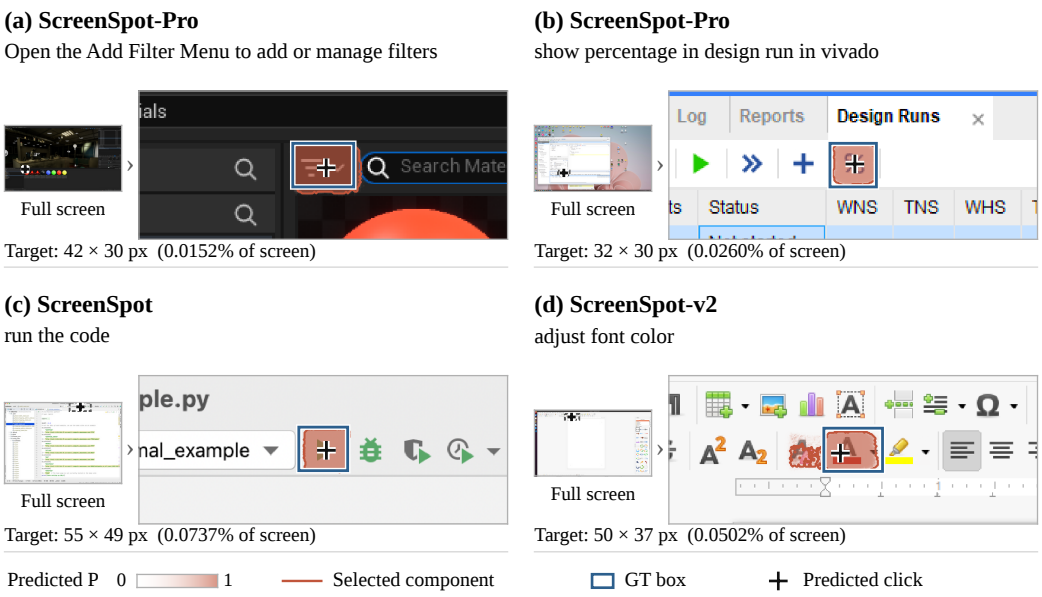

Figure 3: Seg2Click-4B localizes small icons and adjacent controls through learned action masks. Warm overlays show predicted probabilities; outlines mark selected components, blue boxes mark targets, and crosses mark clicks. Insets are display-only enlargements.

D.2 Failure Cases and Limitations

Figure 4 shows four failures in which the predicted region extends beyond the intended target, revealing limitations in boundary localization and mask-to-click decoding.

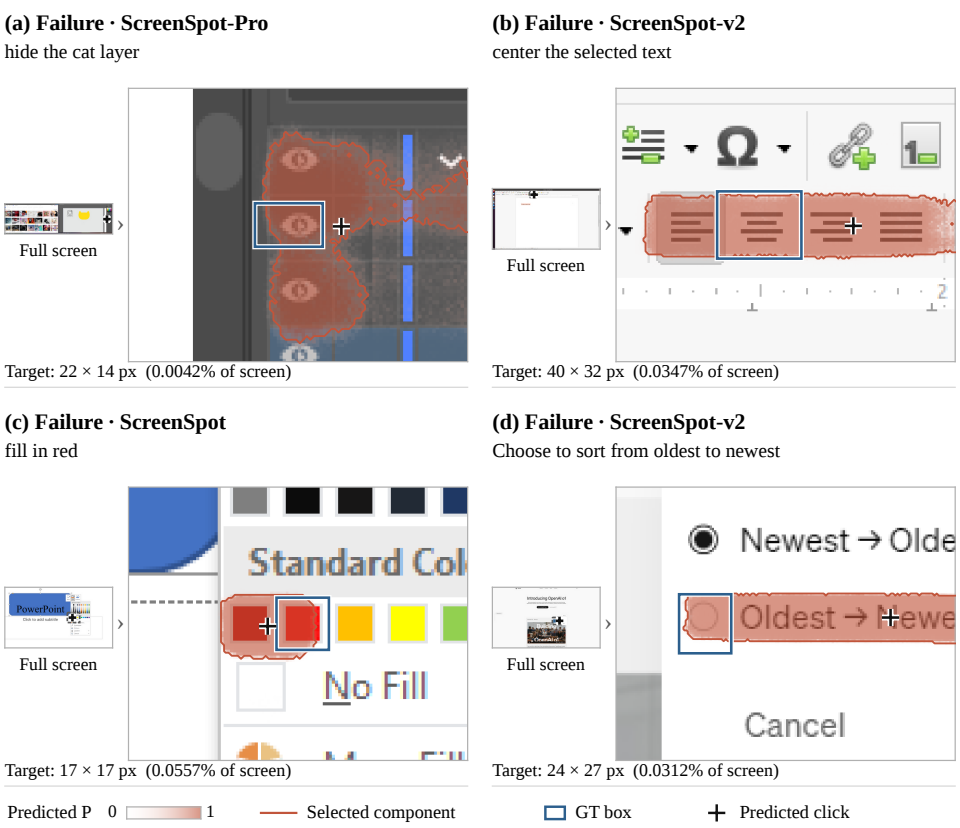

Figure 4: Failures with responses extending beyond the annotated target: (a) layer visibility, (b) paragraph alignment, (c) color selection, and (d) a radio control and its label. Warm mask overlays and component outlines show how the centroid can fall outside the blue target box. Failure labels follow the same inclusive box-hit criterion as the quantitative evaluation.

Separating adjacent controls. In Figure 4(a–c), the selected component spans neighboring layer controls, alignment buttons, or color swatches. In (d), it covers both the radio button and its text label. The response reaches the relevant control group but does not isolate the annotated element. Fine output resolution therefore provides the capacity to separate nearby controls, but accurate boundaries still depend on the learned features and supervision.

Sensitivity of centroid selection. The largest-component rule selects one connected region, and its unweighted centroid gives every foreground pixel equal influence. When the region extends onto an adjacent control or label, these extra pixels can shift the click outside the target box even when the mask overlaps the correct element. This exposes a limitation of fixed geometric decoding: it does not assess whether the selected point lies in the intended control. Confidence-aware click selection could address this issue, but its benefit requires evaluation.

Scope of the evidence. These examples illustrate behavior rather than the frequency of each failure mode. Bounding-box supervision and evaluation approximate the valid click region; for instance, the label click in Figure 4(d) fails the box-hit criterion, but a static screenshot cannot establish whether the interface would accept it. Evaluating Seg2Click within interactive agents remains necessary to measure task completion and recovery from grounding errors.