V-GRPO: View-Guided Group Relative Policy Optimization for GUI Grounding

Anonymous ACL submission

Abstract

When applying Group Relative Policy Optimization (GRPO) for GUI Grounding, rollouts are sampled from a single screenshot view; groups often become either all failures on difficult instances or all successes on easy ones, yielding no useful relative advantage. We propose **V-GRPO**, a view-guided GRPO framework that constructs each comparison group from multiple target-preserving views of the same GUI instance. Each view is generated by a crop that keeps the target element visible and remaps its box exactly, so model rollouts are compared across semantically equivalent but geometrically different inputs. To stabilize short coordinate generation without turning reinforcement learning into unconditional imitation, V-GRPO further adds a model-verified cross-view anchor: an oracle answer optimized with an advantage-weighted loss, excluded from the group baseline and activated only when the model has produced a maximum-reward rollout. Across five GUI-grounding benchmarks and multiple Qwen backbones, V-GRPO consistently improves grounding accuracy. On ScreenSpot-Pro, it raises Qwen3-VL 4B/8B/30B-A3B from 55.5/52.7/53.7 to 63.4/65.8/67.0. Robustness analyses further show higher worst-view accuracy and lower prediction flip rates.

1 Introduction

GUI grounding enables autonomous agents to interact with digital interfaces by mapping a screenshot and a natural language instruction to a click coordinate (Wang et al., 2025a; Zhou et al., 2025a; Gu et al., 2025a; Wang et al., 2025c). Compared with general visual grounding, GUI grounding is especially sensitive to localization errors, as UI elements such as icons, input fields, and buttons are often small, densely arranged, and visually similar; a slight spatial mistake may activate the wrong element and disrupt the subsequent workflow (Tang et al., 2025b; Wang et al., 2025b).

Recent work has made significant progress in applying Group Relative Policy Optimization (GRPO) with verifiable rewards for GUI grounding, where click correctness can be evaluated by a rule-based *point-in-box* reward (Yang et al., 2025; Luo et al., 2025; Tang et al., 2025a).

However, as illustrated in Figure 1, directly applying GRPO to GUI grounding exposes two forms of reward degeneracy, both of which collapse the group-relative advantage and remove the learning signal (Guo et al., 2025; Yu et al., 2026; Lu et al., 2025). With sparse binary *point in box* rewards, repeated rollouts from a single view may all miss the target on difficult screenshots, forming a *fixed view all fail group*. On easier screenshots, all rollouts may hit the target, which also eliminates reward variance. Figure 3f shows that fewer than 5% of fixed-view training samples form informative groups, namely groups that are neither all-zero nor all-one. This issue is severe in GUI grounding because coordinates are tied to the screenshot geometry: a prediction that is correct in one view may shift after a target preserving crop, even when the target remains visible (see Table 5) (Zhang et al., 2025c). Thus, both all-fail and all-correct degeneracy reveal a shared bottleneck: vanilla GRPO constructs groups by repeatedly sampling from one fixed view, which often fails to expose informative differences among rollouts.

These observations suggest that the key design choice is not only how to reward a click, but also how to construct the comparison group. For GUI grounding, an informative group should preserve the same instruction and target semantics while varying the view geometry, so that the policy is compared across semantically equivalent but geometrically different inputs. Such view consistent grouping increases the chance of observing a successful rollout on difficult instances and reveals unstable predictions on easy ones. By keeping the instruction and target unchanged while varying

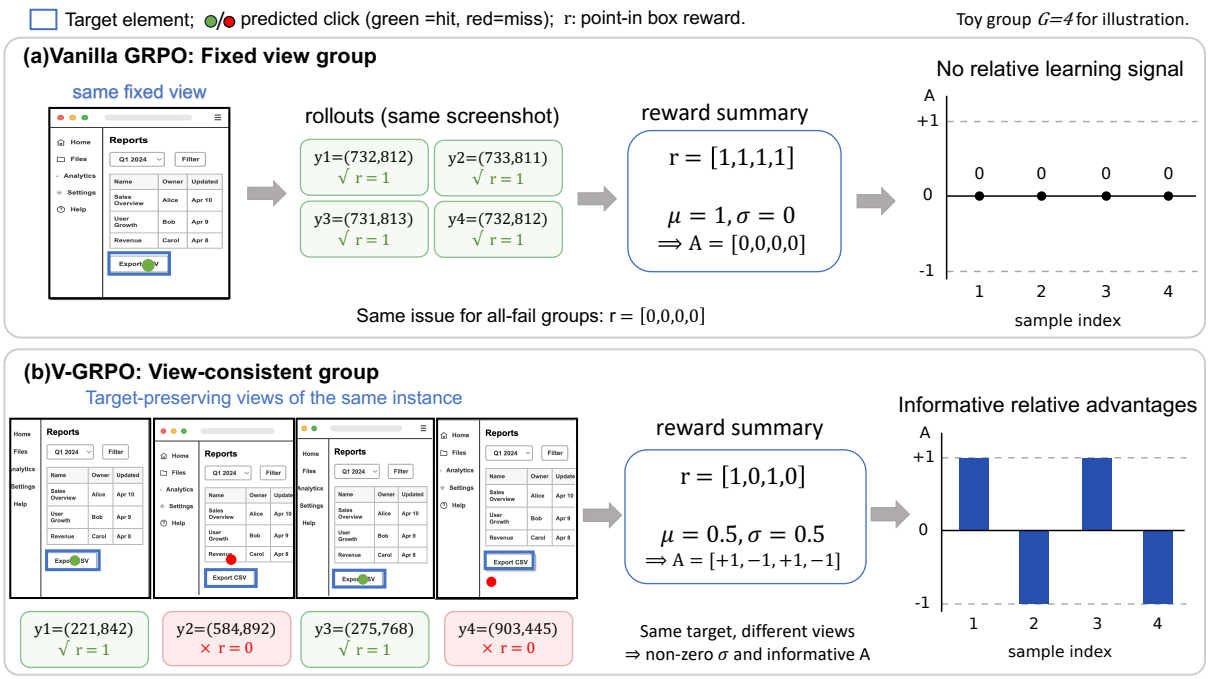


Figure 1: **Motivation of V-GRPO.** In vanilla GRPO, multiple rollouts from the same screenshot can produce homogeneous rewards, yielding zero relative advantage. V-GRPO constructs the group from target-preserving views of the same GUI instance. These views preserve the instruction and target semantics while changing the screenshot geometry. As a result, V-GRPO turns homogeneous fixed-view rewards into informative cross-view variation.

the view geometry, the policy is compared across inputs that require the same action but different coordinate predictions. Since coordinate generation is sensitive to format and geometry, we separate the model generated rollouts used for group statistics from the oracle answer, which is used only as a conditional stabilizing signal after the model has already succeeded.

In this work, we introduce V-GRPO, a GRPO-based training framework for GUI grounding that revisits group construction from this perspective. V-GRPO consists of two components: **View-Consistent Group Rollout** and **Model-Verified Cross-View Anchoring**. View-Consistent Group Rollout constructs each GRPO group from multiple target-preserving views of the same GUI instance, rather than from repeated completions on a single screenshot. As shown in Figure 3f, this construction increases the fraction of informative groups, namely groups that are neither all-zero nor all-one, from fewer than 5% under fixed-view GRPO to around 20%, restoring useful intra-group reward variance for both difficult and easy instances. Thus, V-GRPO mitigates all-fail degeneracy primarily through view-consistent group construction, not through unconditional oracle injection.

However, multi-view rollout alone can make coordinate generation unstable on hard or ambiguous views. To stabilize training, we further introduce a model-verified cross-view anchor that uses oracle answers only as a conditional stabilizing signal, while keeping the GRPO comparison statistics defined by model-generated rollouts.

Our contributions are summarized as follows:

- We propose view-consistent group rollout, which constructs GRPO groups from multiple target-preserving views of the same GUI grounding instance and computes model-only group statistics across these views.
- We introduce a model-verified cross-view anchor that provides an oracle coordinate only when the current policy has already produced a maximum-reward rollout, preventing oracle targets from changing the GRPO baseline or supervising all-fail groups.
- We provide comprehensive validation across model family and benchmarks. On ScreenSpot-Pro, V-GRPO improves Qwen3-VL series models from 55.5/52.7/53.7 to 63.4/65.8/67.0 at the 4B/8B/30B-A3B scales. On Qwen3.5 initialized backbones, V-GRPO improves ScreenSpot-Pro over standard GRPO by +2.0/+0.9/+1.2 points at the 4B/9B/35B-A3B scales, with the 35B-A3B model reaching 72.9.

2 Related Work

GUI Grounding Recent advances in GUI agents have been driven by specialized GUI grounding models that map natural-language instructions to precise screen coordinates (Tang et al., 2025b; Zhang et al., 2025a; Zheng et al., 2024; Wang et al., 2025b; Nguyen et al., 2025). Early works established the task through large-scale supervised fine-tuning (SFT) on GUI datasets, demonstrating effectiveness across mobile, web, and desktop interfaces (Cheng et al., 2024; Lin et al., 2024; Qin et al., 2025a; Yang et al., 2024; Gou et al., 2025; Wu et al., 2024; Lu et al., 2024; Xu et al., 2025). More recently, reinforcement learning with rule-based click rewards has emerged as a promising technique to further enhance grounding accuracy beyond SFT baselines (Yang et al., 2025; Tang et al., 2025a; Zhou et al., 2025b; Luo et al., 2025; Gu et al., 2025a; Team et al., 2026). However, existing RL approaches suffer from diminishing training effectiveness as optimization progresses.

Oracle Guidance in GRPO Training GRPO has been widely adopted for post-training large language models, consistently improving performance across diverse tasks (Yu et al., 2026; Guo et al., 2025; Shao et al., 2024). However, GRPO degenerates when group rollouts are uniformly correct or incorrect, collapsing the relative advantage to zero and eliminating the learning signal. To address this, recent works in math and reasoning try to introduce oracle guidance to ensure informative training updates. LUFFY (Yan et al., 2026) mixes off-policy traces from a stronger teacher into the group, using regularized importance sampling to balance imitation and on-policy exploration. BREAD (Zhang et al., 2026) adaptively inserts partial expert prefixes whenever on-policy rollouts fail, guaranteeing at least one successful trace per update.

3 Method

As illustrated in Figure 2, we introduce V-GRPO, a GRPO-based training framework for GUI grounding. The key idea is to construct the GRPO group from multiple target-preserving views of the same screenshot, rather than repeated rollouts on a single fixed rendering. To stabilize training, V-GRPO appends oracle answer as a cross-view anchor.

3.1 Problem Setup

Given a screenshot I, an instruction q, and a ground-truth target box $B = (x_1, y_1, x_2, y_2)$, GUI grounding asks a policy π_θ to output a coordinate string y. Following the Qwen-style grounding interface, the output string y is parsed into a click point $\hat{\mathbf{p}} = (\hat{x}, \hat{y})$ in the discrete 0–1000 image coordinate space. A prediction is considered correct if the parsed click point lies inside the target box.

We use a verifiable reward that matches this click-based interface. Let $\mathrm{ValidFmt}(y)$ indicate whether y can be parsed as a coordinate string of the form $[x, y]$. The reward is defined as

$$R(y, B) = \mathbb{I}[\mathrm{ValidFmt}(y)] \cdot \mathbb{I}[\hat{\mathbf{p}} \in B]\,. \tag{1}$$

The maximum reward is therefore $R_{\max} = 1$.

3.2 GRPO for GUI Grounding

Standard GRPO (Guo et al., 2025) samples G completions $\{y_i\}_{i=1}^{G}$ for the same prompt and computes normalized advantages from the group rewards:

$$\begin{aligned} \mu_G &= \frac{1}{G}\sum_{i=1}^{G} r_i, \sigma_G = \sqrt{\frac{1}{G}\sum_{i=1}^{G}(r_i - \mu_G)^2}, \\ A_i &= \frac{r_i - \mu_G}{\sigma_G + \epsilon}. \end{aligned} \tag{2}$$

Directly applying GRPO to GUI grounding has two limitations: rollouts from a fixed screenshot do not encourage robustness to view changes, and short coordinate outputs often collapse to near-identical strings, producing low reward variance. We therefore construct each group from target-preserving views of the same task, so that reward comparisons measure localization across renderings rather than repeated completions under one fixed screenshot.

3.3 View-Consistent Group Rollout

A valid view transformation must preserve the target geometry and allow exact label remapping. We use target-preserving crops that fully contain the target, with constrained crop scale and contextual margins to reduce semantic ambiguity. Each crop is therefore a view-consistent approximation of the original grounding task.

We view each crop as a sample from a conditional view distribution $\mathcal{T}(\cdot \mid I, B)$, so the group baseline is computed over target-preserving renderings of the same grounding instance rather than a single fixed prompt. With probability p_{crop}, a training instance is converted into K target-preserving cropped views. Otherwise, we use the original full-screen image as a single view and sample G completions from it, recovering the fixed-view GRPO group.

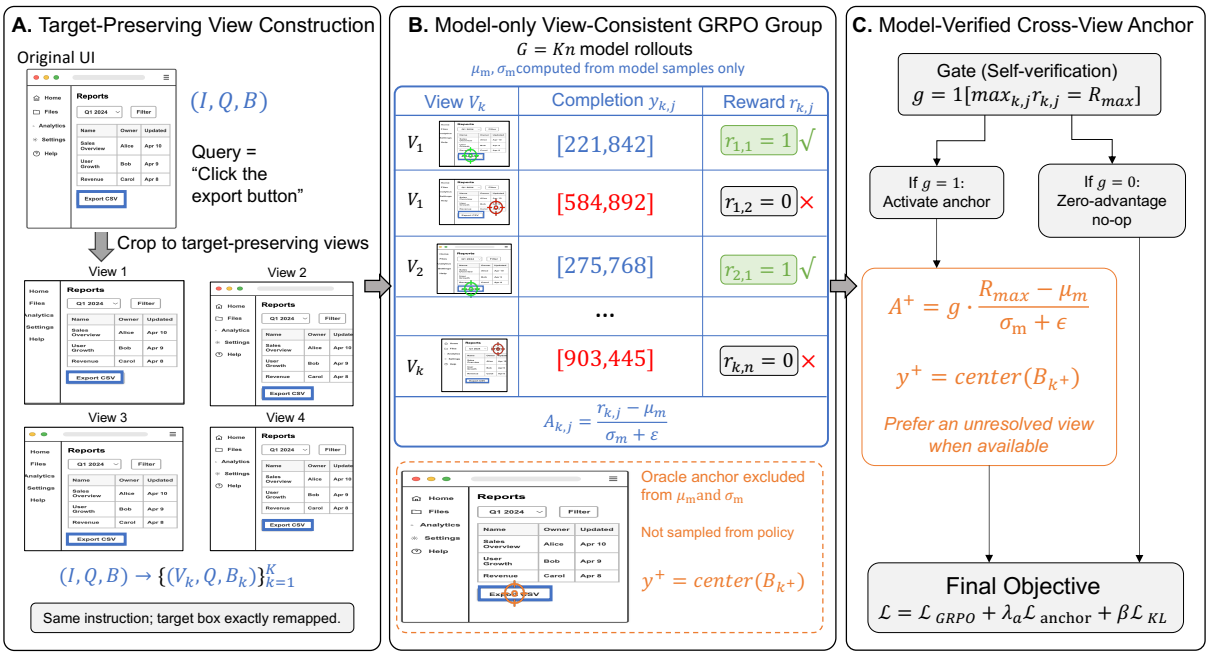


Figure 2: **Overview of V-GRPO.** V-GRPO constructs GRPO groups from target-preserving views of the same GUI grounding instance. Model rollouts define the group statistics, while an oracle-format center-point anchor is activated only for self-verified groups.

Target-preserving crop generation. Let the original image size be $W \times H$, and let $B^{\text{px}} = (x_1, y_1, x_2, y_2)$ denote the target box in pixel coordinates. For each cropped view k, the initial crop size is set to $(w_k, h_k) = (0.9W, 0.9H)$. If the resulting crop is smaller than the target box, we enlarge it so that the target can be fully contained.

A valid crop window $C_k = (l_k, t_k, w_k, h_k)$ must satisfy $B^{\text{px}} \subseteq C_k$. For each view, we independently sample the top-left corner uniformly from these feasible ranges:

$$\begin{aligned} l_k &\sim \mathcal{U}\left(\max(0, x_2 - w_k),\ \min(x_1, W - w_k)\right), \\ t_k &\sim \mathcal{U}\left(\max(0, y_2 - h_k),\ \min(y_1, H - h_k)\right), \end{aligned} \tag{3}$$

for $k = 1, \ldots, K$. This yields K cropped views without truncating the target UI element.

Coordinate remapping. For each valid crop, we remap the target box into the cropped coordinate frame and normalize it:

$$\begin{aligned} B_k = \big(&1000\tfrac{x_1 - l_k}{w_k}, 1000\tfrac{y_1 - t_k}{h_k}, \\ &1000\tfrac{x_2 - l_k}{w_k}, 1000\tfrac{y_2 - t_k}{h_k}\big). \end{aligned} \tag{4}$$

The cropped image is then resized for the VLM input, while the target box is obtained by exact geometric remapping.

Group construction. Given K target-preserving views $\{(V_k, B_k)\}_{k=1}^K$, we sample $n = G/K$ completions per view, assuming K divides the GRPO group size G:

$$\begin{aligned} y_{k,j} &\sim \pi_{\theta_{\text{old}}}(\cdot \mid V_k, q), \\ r_{k,j} &= R(y_{k,j}, B_k), \end{aligned} \qquad k = 1, \ldots, K,\ j = 1, \ldots, n. \tag{5}$$

The model-generated group contains $G = Kn$ rollouts. Its statistics are computed as

$$\begin{aligned} \mu_{\text{m}} &= \frac{1}{G} \sum_{k=1}^{K} \sum_{j=1}^{n} r_{k,j}, \\ \sigma_{\text{m}} &= \sqrt{\frac{1}{G} \sum_{k=1}^{K} \sum_{j=1}^{n} (r_{k,j} - \mu_{\text{m}})^2}, \end{aligned} \tag{6}$$

and the corresponding model-sample advantages are

$$A_{k,j} = \frac{r_{k,j} - \mu_{\text{m}}}{\sigma_{\text{m}} + \epsilon}. \tag{7}$$

When $n > 1$, the group retains within-view sampling stochasticity; when $K = G$, it maximizes cross-view diversity. In both cases, advantages compare predictions across views of the same semantic target.

3.4 Model-Verified Cross-View Anchoring

As shown by the training diagnostics in Figure 3, multi-view rollouts improve exploration over target-preserving views, but they can also make the reward signal less stable. In particular, GUI grounding outputs are short coordinate strings, and repeated negative advantages on invalid or poorly localized completions may reduce the probability of producing valid coordinates. To stabilize training

without reverting to unconditional supervised fine-tuning, we introduce a **model-verified** cross-view anchor. We use the term **model-verified** to indicate that the anchor is activated only when a model-generated rollout, rather than the oracle sequence itself, achieves the maximum verifier reward.

The anchor uses the ground-truth geometry only when the current policy has already produced evidence that the instance is solvable within the same view-consistent group. A maximum-reward rollout indicates that the policy can ground the target in at least one target-preserving view. In contrast, if no rollout reaches the maximum reward, the group provides no model-side evidence that the instance is currently solvable. In such cases, forcing an oracle coordinate would turn the update into unconditional supervised learning. We therefore activate the anchor only for self-verified groups.

3.5 Training Objective

Let $\mathcal{P} = \{(k, j) : r_{k,j} = R_{\max}\}$ denote the set of maximum-reward model rollouts, and let $\mathcal{N}$ be its complement within the group. When $\mathcal{N}$ is non-empty, we choose the anchor view uniformly from $\mathcal{N}$, so that the oracle anchor preferentially targets a view where the current policy has not yet produced a perfect prediction. If all model rollouts are already perfect, we choose the anchor view uniformly from the full group.

Given the selected view V_{k^+} and its remapped box $B_{k^+} = (x_1, y_1, x_2, y_2)$, we construct the oracle-format coordinate sequence as the box center, $y^+ = [\lfloor (x_1 + x_2)/2 \rfloor,\ \lfloor (y_1 + y_2)/2 \rfloor]$.

Although this sequence is geometrically valid by construction, it is not assigned a positive reward unconditionally. The anchor advantage is computed with the model-only group baseline:

$$A^+ = \mathbb{I}[\mathcal{P} \neq \emptyset]\, \frac{R_{\max} - \mu_{\mathrm{m}}}{\sigma_{\mathrm{m}} + \epsilon}. \tag{8}$$

The oracle sequence is excluded from μ_{m} and σ_{m}. This is important: including the oracle in the group statistics would allow a ground-truth sequence to shift the baseline, creating implicit supervision even for groups where the model never succeeds. By contrast, our model-only baseline ensures that the anchor contributes a nonzero supervised signal only when the group is self-verified by at least one maximum-reward rollout. When all model rollouts fail to reach $R_{\max}$, the anchor becomes a zero-advantage no-op. The model-generated rollouts may still contribute standard GRPO updates if their rewards are non-identical, but no oracle-anchor update is applied.

This gating prevents the anchor from becoming unconditional supervised fine-tuning on groups where the policy has not yet produced any evidence of successful grounding. It also reduces the influence of noisy annotations or ambiguous target views.

This design distinguishes V-GRPO from common mixed-policy or off-policy-supervised variants (Yan et al., 2026; Zhang et al., 2026). Ground-truth injection inserts oracle completions into the RL update or candidate group, which can alter the effective comparison set. Fixed-weight SFT mixing adds a supervised loss with a constant coefficient, regardless of whether the current policy has already discovered the target. In contrast, our anchor is model-verified, baseline-excluded, and conditional: it provides a supervised stabilizer only for groups where the current policy has already demonstrated successful grounding, while avoiding unconditional imitation on groups where grounding has not yet emerged.

For a model-generated completion $y_{k,j}$, we use the clipped GRPO objective

$$\begin{aligned} \ell_{\mathrm{clip}}(y_{k,j}, A_{k,j}) &= \sum_t \min\left(\rho_{k,j,t} A_{k,j}, \bar{\rho}_{k,j,t} A_{k,j}\right), \\ \bar{\rho}_{k,j,t} &= \mathrm{clip}\left(\rho_{k,j,t}, 1 - \epsilon_{\mathrm{clip}}, 1 + \epsilon_{\mathrm{clip}}\right), \end{aligned} \tag{9}$$

where

$$\rho_{k,j,t} = \frac{\pi_\theta(y_{k,j,t} \mid V_k, q, y_{k,j,<t})}{\pi_{\theta_{\mathrm{old}}}(y_{k,j,t} \mid V_k, q, y_{k,j,<t})}. \tag{10}$$

The oracle sequence is not an on-policy sample. We therefore optimize it as an advantage-weighted supervised anchor:

$$\ell_{\mathrm{anchor}}(y^+, A^+) = \mathrm{sg}(A^+) \sum_t \log \pi_\theta\left(y_t^+ \mid V_{k^+}, q, y_{<t}^+\right), \tag{11}$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient.

The final loss is

$$\begin{aligned} \mathcal{L}_{\text{V-GRPO}} = -\frac{1}{G+1} \Bigg[&\sum_{k=1}^{K} \sum_{j=1}^{n} \ell_{\mathrm{clip}}(y_{k,j}, A_{k,j}) \\ &+ \lambda_{\mathrm{a}} \ell_{\mathrm{anchor}}(y^+, A^+) \Bigg] + \beta \mathcal{L}_{\mathrm{KL}}, \end{aligned} \tag{12}$$

where λ_{a} controls the strength of the anchor loss and is set to 1 unless otherwise specified. In implementation, the oracle coordinate sequence is appended as an additional sequence and optimized through the same token-level training pipeline as the model completions.

Table 1: **Overall results of V-GRPO across five GUI-grounding benchmarks.** We highlight the best and second-best results within each model size category. * denotes our evaluated results.

Model	SSPro	SSV2	MMBench-L2	OSWorld-G-R	OSWorld-G	Avg.
≈4B						
Qwen3-VL-4B* (Bai et al., 2025)	55.5	88.5	85.3	67.9	58.2	71.1
Step-GUI-4B (Yan et al., 2025)	60.0	93.6	84.0	66.9	60.5	73.0
V-GRPO-4B	63.4	94.4	86.7	69.4	63.8	75.5
+ *MVP*	71.6	94.5	86.8	69.6	64.0	77.3
≈8B						
UI-TARS-1.5-7B (Qin et al., 2025a)	35.7	91.6	64.3	64.2	52.8	61.7
OpenCUA-7B (Wang et al., 2025c)	50.0	92.3	-	-	55.3	-
GTA1-7B (Yang et al., 2025)	50.1	92.4	-	67.7	60.1	-
UI-Venus-7B (Gu et al., 2025a)	50.8	94.1	79.9	61.7	54.6	68.2
GUI-Owl-7B (Ye et al., 2025)	54.9	92.8	80.5	-	55.9	-
Step-GUI-8B (Yan et al., 2025)	62.6	95.1	85.6	70.0	-	-
Qwen3-VL-8B* (Bai et al., 2025)	52.7	91.7	81.3	64.4	54.8	69.0
Holo2-8B (Company, 2025)	58.9	93.2	84.5	70.1	63.5	74.0
MAI-UI-8B (Zhou et al., 2025a)	65.8	95.2	88.8	68.6	60.1	75.7
V-GRPO-8B	65.8	95.5	86.8	70.8	62.4	76.3
+ *MVP*	72.0	95.6	87.3	70.9	63.1	77.8
≥30B						
Qwen3-VL-30B-A3B* (Bai et al., 2025)	53.7	94.7	83.7	69.3	66.5	73.6
Holo2-30B-A3B (Company, 2025)	66.1	94.9	86.8	76.1	65.2	77.8
OpenCUA-32B (Wang et al., 2025c)	55.3	93.4	-	70.2	59.6	-
GUI-Owl-32B (Ye et al., 2025)	58.0	93.1	83.0	-	58.0	-
GTA1-32B (Yang et al., 2025)	63.6	95.2	-	72.2	65.2	-
OpenCUA-72B (Wang et al., 2025c)	60.8	92.9	-	-	-	-
UI-Venus-72B (Gu et al., 2025a)	61.9	95.3	86.3	69.5	62.2	75.0
V-GRPO-30A3B	67.0	95.2	86.8	72.0	67.1	77.6
+ *MVP*	74.1	95.4	87.6	72.5	67.6	79.4

4 Experiments

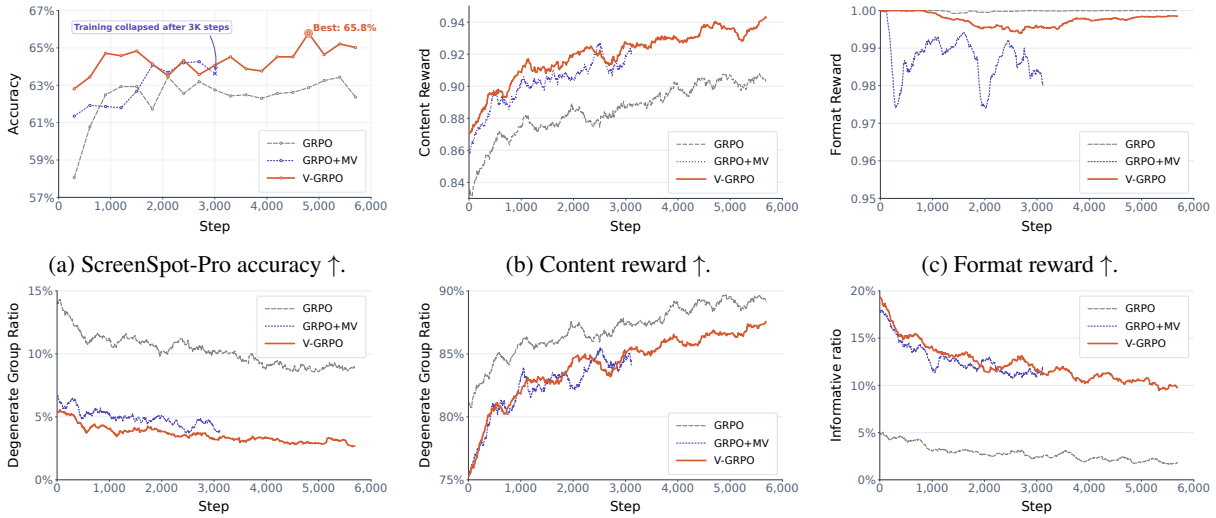
We evaluate V-GRPO from three perspectives. First, we test whether view-guided GRPO improves GUI grounding across model scales and benchmarks. Second, we examine whether the improvement is tied to a specific Qwen3-VL initialization. Third, we isolate the effects of view-consistent dynamic cropping, adaptive oracle anchoring, and crop-set robustness. Unless otherwise specified, V-GRPO uses $K{=}G{=}8$ target-preserving views and appends one oracle center-point completion per training block.

4.1 Experimental Setup

Benchmarks and metrics. We report results on five GUI-grounding benchmarks: ScreenSpot-Pro (Li et al., 2025), ScreenSpot-V2 (Wu et al., 2024), MMBench-GUI L2 (Xuehui Wang et al., 2025), OSWorld-G-R, and OSWorld-G (Xie et al., 2025). These benchmarks cover mobile, web, desktop, and high-resolution professional software interfaces. For all datasets, the model predicts a normalized coordinate in the 0–1000 frame, and a prediction is counted as correct when the point falls inside the target element. We use accuracy as the primary metric and report the average over the listed benchmarks when all required numbers are available in the corresponding table. All evaluations are conducted with deterministic decoding at temperature 0.

Models and baselines. Our main experiments instantiate V-GRPO on Qwen3-VL backbones at the 4B, 8B, and 30B-A3B scales. We compare against the original Qwen3-VL models and recent GUI grounding models within the same parameter-scale groups. All V-GRPO rows use standard single-view inference unless marked with + *MVP*. MVP (Zhang et al., 2025c) is an orthogonal inference-time multi-view aggregation method; we include it to test whether a model trained with view-consistent rollouts remains compatible with test-time view aggregation. For cross-backbone evaluation, we additionally train Qwen3.5-initialized models and compare V-GRPO against standard GRPO under the same backbone family.

Training dataset and cost. We train V-GRPO on roughly 120K GUI-grounding samples curated from open-source datasets, including SeeClick (Cheng et al., 2024), Widget Captioning (Li et al., 2020), ShowUI-web (Lin et al., 2024), UI-RefExp (Bai et al., 2021), OmniAct (Kapoor et al., 2024). Compared with standard GRPO, V-GRPO incurs a moderate training overhead: across different model sizes, training for the same number of steps increases wall-clock time by approximately 25% under the same training setup.



(a) ScreenSpot-Pro accuracy ↑. (b) Content reward ↑. (c) Format reward ↑.

(d) All-zero degenerate ratio ↓. (e) All-one degenerate ratio ↓. (f) Informative group ratio ↑.

Figure 3: **Training dynamics and reward diagnostics.** Degenerate group ratios: "All-zero" groups contain only zero-reward responses, while "All-one" groups contain only full-reward responses; both are uninformative for policy-gradient updates and lower is better. The informative group ratio, $1 - \mathrm{ratio}_{\mathrm{all-zero}} - \mathrm{ratio}_{\mathrm{all-one}}$, is the share of non-degenerate groups containing both successful and unsuccessful rollouts.

Implementation details. All V-GRPO runs use G=8 generations. For V-GRPO, we use K=8 target-preserving views, one completion per view, and one model-verified oracle anchor with λ_{a}=1. Dynamic cropping is applied to $p_{crop} = 80\%$ of training examples, while the remaining 20% use pass-through full-screen views. We train with DeepSpeed ZeRO-3, bfloat16 precision, frozen vision modules, a learning rate of 1×10^{-6}, KL coefficient β=0.04, and total batch size 128.

4.2 Main Results

Table 1 shows that V-GRPO consistently improves the Qwen3-VL backbone across scales. At 4B, 8B, and 30B-A3B, the average score increases from 71.1/69.0/73.6 to 75.5/76.3/77.6, respectively. The largest gains appear on ScreenSpot-Pro, where small targets and dense high-resolution interfaces make coordinate errors especially costly: V-GRPO improves Qwen3-VL by +7.9, +13.1, and +13.3 points at the three scales. By contrast, ScreenSpot-V2 is already near saturation for strong GUI models, so the gains there are smaller but still positive. This pattern supports the central motivation of V-GRPO: view-consistent rollouts and adaptive anchoring mainly help on difficult instances rather than only improving easy benchmarks.

V-GRPO also combines well with inference-time multi-view prediction. Adding MVP further raises the average scores to 77.3, 77.8, and 79.4 at the 4B, 8B, and 30B-A3B scales. The improvement is again most pronounced on ScreenSpot-Pro, where the + *MVP* rows reach 71.6, 72.0, and 74.1. Since MVP is applied only at inference time, these results indicate that the robustness learned by V-GRPO is complementary to test-time view aggregation.

Table 2: **Cross-backbone generalization with Qwen3.5-initialized models.**

Method	SSPro	SSV2	OS -G	OS-G-R
4B				
Qwen3.5-4B	60.3	90.4	54.4	66.8
GRPO-4B	62.2	**94.2**	59.9	69.2
V-GRPO-4B	**64.2**	93.8	**61.2**	**69.7**
Δ	**+2.0**	-0.4	**+1.3**	**+0.5**
9B				
Qwen3.5-9B	65.2	91.9	63.1	74.6
GRPO-9B	68.3	95.2	67.5	75.2
V-GRPO-9B	**69.2**	**95.8**	**68.1**	**75.5**
Δ	**+0.9**	**+0.6**	**+0.6**	**+0.3**
35B-A3B				
Qwen3.5-35B-A3B	68.6	93.8	65.8	72.5
GRPO-35B-A3B	71.7	95.7	70.4	74.3
V-GRPO-35B-A3B	**72.9**	**95.8**	**71.5**	**75.3**
Δ	**+1.2**	**+0.1**	**+1.1**	**+1.0**

Figure 3 summarizes the training dynamics and reward diagnostics. Compared with standard GRPO, multi-view rollout reduces all-zero degeneracy and improves content reward, but its format reward is noticeably unstable during training. This confirms that view diversification alone can introduce additional coordinate-format instability. By combining view-consistent rollout with the model-verified anchor, V-GRPO maintains a high and stable format reward while improving content reward and ScreenSpot-Pro accuracy.

4.3 Cross-Backbone Generalization

To test whether V-GRPO depends on Qwen3-VL initialization, we train Qwen3.5-initialized backbones at 4B, 9B, and 35B-A3B scales and compare

Table 3: **Ablation study on the components of V-GRPO.** The shaded row marks the full model with dynamic cropping and adaptive supervision. View group denotes view-consistent rollout groups; Anchor denotes model-verified adaptive anchor supervision.

Method	Components		Accuracy ↑	
	View group	Anchor	SSV2	SSPro
Qwen3-VL-8B	✗	✗	91.7	54.6
SFT	✗	✗	93.6	59.8
SFT + aug	✗	✗	94.6	60.5
GRPO	✗	✗	95.2	63.4
GRPO + crop	✓	✗	95.4	64.3
GRPO + anchor	✗	✓	95.3	64.8
V-GRPO	✓	✓	**95.5**	**65.8**

Table 4: **Ablation study on anchor supervision** (Qwen3-VL-8B). Adaptive gating avoids unverified oracle updates while preserving a model-only baseline. Gate denotes model-verified adaptive activation.

Anchor	Gate	SSV2	SSPro
None	–	95.4	64.3
Normalized	✗	93.8	57.8
Const. SFT	✗	94.8	63.9
Normalized	✓	**95.5**	**65.8**

them against standard GRPO. Table 2 shows that V-GRPO transfers beyond the Qwen3-VL family. It improves ScreenSpot-Pro at all three scales and improves OSWorld-G-R for each reported backbone. At 9B, V-GRPO still improves ScreenSpot-Pro, ScreenSpot-V2, and OSWorld-G-R.

4.4 Ablation Study

Table 3 isolates the two components of V-GRPO on the 8B model. Plain supervised crop augmentation has only a small effect on ScreenSpot-Pro, improving SFT from 59.8 to 60.5. This confirms that the benefit does not come from adding cropped images alone. Within RL, dynamic crop alone improves standard GRPO from 63.4 to 64.3, while adaptive supervision alone improves it to 64.8. Combining both components gives the best result, 65.8 on ScreenSpot-Pro and 95.5 on ScreenSpot-V2, showing that view construction and adaptive anchoring address complementary failure modes.

Supervision strategy. Table 4 compares different anchor-supervision strategies under the GRPO + multi-view rollout setting. Simply adding a normalized oracle anchor without gating severely hurts performance, dropping ScreenSpot-Pro from 64.3 to 57.8. This is because many early groups are all-zero: all model rollouts receive reward 0, so the model-only baseline has $\mu_{\mathrm{m}} = 0$ and $\sigma_{\mathrm{m}} = 0$.

Table 5: **Comparison under different view settings.** Worst denotes worst-view accuracy; VCR is view-consistency rate; Flip is prediction flip rate; Base is Qwen3-VL-8B.

Model	Accuracy ↑			VCR ↑	Flip ↓
	Orig.	Crop	Worst		
Base	81.82	81.25	71.46	75.76	17.28
GRPO	94.19	93.00	87.63	88.38	8.31
V-GRPO	**95.71**	**96.25**	**92.42**	**90.40**	**5.80**

If the ungated oracle anchor is still assigned reward 1, its normalized advantage becomes approximately $1/\epsilon$, producing an excessively large supervised update on unverified examples. Constant SFT avoids this advantage explosion, but it applies the oracle signal unconditionally and brings only marginal gain. In contrast, our gated anchor activates only when the group is self-verified by at least one maximum-reward model rollout. Thus, all-zero groups receive no oracle update, while verified groups still benefit from a positive stabilizing signal. This yields the best results, 95.5 on ScreenSpot-V2 and 65.8 on ScreenSpot-Pro.

4.5 Crop Perturbation Diagnostic

Finally, we conduct a training diagnostic to measure sensitivity to crop perturbations. We sample 250 training examples and, for each, evaluate the original view with eight randomly cropped target-preserving views, using the same crop procedure as training. We report original and crop-view accuracy, worst-view accuracy, view consistency rate (VCR), and prediction flip rate, where VCR measures prediction consistency for the same query across image inputs. For the exact computation of VCR and prediction flip rate, see Appendix A.2. Table 5 shows that V-GRPO improves both average accuracy and cross-view stability. Compared with standard GRPO, V-GRPO raises crop accuracy from 93.00% to 96.25%, worst-view accuracy from 87.63% to 92.42%, and VCR from 88.38% to 90.40%. It also reduces the prediction flip rate from 8.31% to 5.80%.

5 Conclusion

We presented V-GRPO, a view-guided GRPO framework that uses target-preserving cropped views, exact coordinate remapping, and an oracle anchor to better align RL with GUI grounding. Across benchmarks, scales, and backbones, V-GRPO improves accuracy and crop robustness, suggesting that view-consistent group construction is an effective training signal for GUI grounding.

Limitations

V-GRPO is designed for actionable GUI grounding tasks whose supervision can be verified by coordinate-format rewards. For datasets that mix actionable instructions with refusal-style examples, the anchor mechanism should be applied selectively rather than uniformly. In our implementation, oracle coordinate anchors are used only for actionable samples with valid target boxes, while refusal or non-actionable samples can be routed to a separate training objective or excluded from anchor activation. Equivalently, the anchor gate can be extended with task-type checks, so that refusal examples do not introduce non-coordinate anchor sequences into the cross-view group. This suggests that mixed actionable/refusal datasets are compatible with the framework, but require refusal-aware routing or reward design to avoid conflating coordinate grounding with response-style learning.

V-GRPO also introduces additional optimization sensitivity through view-consistent cropping. Although cross-view grouping improves reward diversity, overly aggressive cropping can increase variance in the RL update, especially when the crop probability or the number of cropped views is large. We therefore use a conservative augmentation schedule: we reduce p_{crop}, limit the number of crop views per group, and retain pass-through original views to stabilize training and reduce train–test mismatch. These choices make the point-in-box reward reliable throughout training while preserving the robustness benefits of cross-view comparison. Future work may further automate this schedule with adaptive crop probability, dynamic KL control, and format-aware early stopping.

References

Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, and 1 others. 2021. Uibert: Learning generic multimodal representations for ui understanding. *arXiv preprint arXiv:2107.13731*.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, and 45 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

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

H Company. 2025. Holo2 - open foundation models for navigation and computer use agents.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. 2025. Navigating the digital world as humans do: Universal visual grounding for GUI agents. In *The Thirteenth International Conference on Learning Representations*.

Zhangxuan Gu, Zhengwen Zeng, Zhenyu Xu, Xingran Zhou, Shuheng Shen, Yunfei Liu, Beitong Zhou, Changhua Meng, Tianyu Xia, Weizhi Chen, Yue Wen, Jingya Dou, Fei Tang, Jinzhen Lin, Yulin Liu, Zhenlin Guo, Yichen Gong, Heng Jia, Changlong Gao, and 5 others. 2025a. Ui-venus technical report: Building high-performance ui agents with rft. *Preprint*, arXiv:2508.10833.

Zhangxuan Gu, Zhengwen Zeng, Zhenyu Xu, Xingran Zhou, Shuheng Shen, Yunfei Liu, Beitong Zhou, Changhua Meng, Tianyu Xia, Weizhi Chen, and 1 others. 2025b. Ui-venus technical report: Building high-performance ui agents with rft. *arXiv preprint arXiv:2508.10833*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Raghav Kapoor, Yash Parag Butala, Melisa Russak, Jing Yu Koh, Kiran Kamble, Waseem Alshikh, and Ruslan Salakhutdinov. 2024. Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web. *Preprint*, arXiv:2402.17553.

Kaixin Li, Meng Ziyang, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. 2025. Screenspot-pro: GUI grounding for professional high-resolution computer use. In *Workshop on Reasoning and Planning for Large Language Models*.

Yang Li, Gang Li, Luheng He, Jingjie Zheng, Hong Li, and Zhiwei Guan. 2020. Widget captioning: Generating natural language description for mobile user interface elements. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 5495–5510.

Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Weixian Lei, Lijuan Wang, and Mike Zheng Shou. 2024. Showui: One vision-language-action model for gui visual agent. *Preprint*, arXiv:2411.17465.

Yuhang Liu, Zeyu Liu, Shuanghe Zhu, Pengxiang Li, Congkai Xie, Jiasheng Wang, Xueyu Hu, Xiaotian Han, Jianbo Yuan, Xinyao Wang, and 1 others. 2025.

Infigui-g1: Advancing gui grounding with adaptive exploration policy optimization. *arXiv preprint arXiv:2508.05731*.

Quanfeng Lu, Wenqi Shao, Zitao Liu, Fanqing Meng, Boxuan Li, Botong Chen, Siyuan Huang, Kaipeng Zhang, Yu Qiao, and Ping Luo. 2024. Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices. *arXiv preprint arXiv:2406.08451*.

Zhengxi Lu, Yuxiang Chai, Yaxuan Guo, Xi Yin, Liang Liu, Hao Wang, Guanjing Xiong, and Hongsheng Li. 2025. Ui-r1: Enhancing action prediction of gui agents by reinforcement learning. *arXiv preprint arXiv:2503.21620*.

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

Dang Nguyen, Jian Chen, Yu Wang, Gang Wu, Namyong Park, Zhengmian Hu, Hanjia Lyu, Junda Wu, Ryan Aponte, Yu Xia, Xintong Li, Jing Shi, Hongjie Chen, Viet Dac Lai, Zhouhang Xie, Sungchul Kim, Ruiyi Zhang, Tong Yu, Mehrab Tanjim, and 11 others. 2025. Gui agents: A survey. In *Proceedings of Annual Meeting of the Association for Computational Linguistics*.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, and 1 others. 2025a. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, and 1 others. 2025b. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. *URL https://arxiv.org/abs/2402.03300*, 2(3):5.

Fei Tang, Zhangxuan Gu, Zhengxi Lu, Xuyang Liu, Shuheng Shen, Changhua Meng, Wen Wang, Wenqi Zhang, Yongliang Shen, Weiming Lu, Jun Xiao, and Yueting Zhuang. 2025a. Gui-g^2: Gaussian reward modeling for gui grounding. *Preprint*, arXiv:2507.15846.

Fei Tang, Haolei Xu, Hang Zhang, Siqi Chen, Xingyu Wu, Yongliang Shen, Wenqi Zhang, Guiyang Hou, Zeqi Tan, Yuchen Yan, Kaitao Song, Jian Shao, Weiming Lu, Jun Xiao, and Yueting Zhuang. 2025b. A survey on (m)llm-based gui agents. *Preprint*, arXiv:2504.13865.

Venus Team, Changlong Gao, Zhangxuan Gu, Yulin Liu, Xinyu Qiu, Shuheng Shen, Yue Wen, Tianyu Xia, Zhenyu Xu, Zhengwen Zeng, and 1 others. 2026. Ui-venus-1.5 technical report. *arXiv preprint arXiv:2602.09082*.

Haoming Wang, Haoyang Zou, Huatong Song, Jiazhan Feng, Junjie Fang, Junting Lu, Longxiang Liu, Qinyu Luo, Shihao Liang, Shijue Huang, and 1 others. 2025a. Ui-tars-2 technical report: Advancing gui agent with multi-turn reinforcement learning. *arXiv preprint arXiv:2509.02544*.

Shuai Wang, Weiwen Liu, Jingxuan Chen, Yuqi Zhou, Weinan Gan, Xingshan Zeng, Yuhan Che, Shuai Yu, Xinlong Hao, Kun Shao, Bin Wang, Chuhan Wu, Yasheng Wang, Ruiming Tang, and Jianye Hao. 2025b. Gui agents with foundation models: A comprehensive survey. *Preprint*, arXiv:2411.04890.

Xinyuan Wang, Bowen Wang, Dunjie Lu, Junlin Yang, Tianbao Xie, Junli Wang, Jiaqi Deng, Xiaole Guo, Yiheng Xu, Chen Henry Wu, Zhennan Shen, Zhuokai Li, Ryan Li, Xiaochuan Li, Junda Chen, Boyuan Zheng, Peihang Li, Fangyu Lei, Ruisheng Cao, and 23 others. 2025c. Opencua: Open foundations for computer-use agents. *Preprint*, arXiv:2508.09123.

Qianhui Wu, Kanzhi Cheng, Rui Yang, Chaoyun Zhang, Jianwei Yang, Huiqiang Jiang, Jian Mu, Baolin Peng, Bo Qiao, Reuben Tan, and 1 others. 2025. Gui-actor: Coordinate-free visual grounding for gui agents. *arXiv preprint arXiv:2506.03143*.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and Yu Qiao. 2024. Os-atlas: A foundation action model for generalist gui agents. *Preprint*, arXiv:2410.23218.

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. 2025. Scaling computer-use grounding via user interface decomposition and synthesis. *Preprint*, arXiv:2505.13227.

Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. 2025. Aguvis: Unified pure vision agents for autonomous gui interaction. *Preprint*, arXiv:2412.04454.

JingJing Xie Xuehui Wang, Zhenyu Wu and 1 others. 2025. Mmbench-gui: Hierarchical multi-platform evaluation framework for gui agents. *arXiv preprint arXiv:2507.19478*.

Haolong Yan, Jia Wang, Xin Huang, Yeqing Shen, Ziyang Meng, Zhimin Fan, Kaijun Tan, Jin Gao, Lieyu Shi, Mi Yang, and 1 others. 2025. Step-gui technical report. *arXiv preprint arXiv:2512.15431*.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. 2026. Learning to reason under off-policy guidance. *Advances in Neural Information Processing Systems*, 38:117157–117186.

Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe Huang, Amrita Saha, Zeyuan Chen, Ran Xu, Liyuan Pan, Caiming Xiong, and Junnan Li. 2025. Gta1: Gui test-time scaling agent. *Preprint*, arXiv:2507.05791.

Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. 2024. Aria-ui: Visual grounding for gui instructions. *Preprint*, arXiv:2412.16256.

Jiabo Ye, Xi Zhang, Haiyang Xu, Haowei Liu, Junyang Wang, Zhaoqing Zhu, Ziwei Zheng, Feiyu Gao, Junjie Cao, Zhengxi Lu, and 1 others. 2025. Mobile-agent-v3: Fundamental agents for gui automation. *arXiv preprint arXiv:2508.15144*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2026. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244.

Xinbin Yuan, Jian Zhang, Kaixin Li, Zhuoxuan Cai, Lujian Yao, Jie Chen, Enguang Wang, Qibin Hou, Jinwei Chen, Peng-Tao Jiang, and 1 others. 2025. Enhancing visual grounding for gui agents via self-evolutionary reinforcement learning. *arXiv preprint arXiv:2505.12370*.

Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue Liu, Qingwei Lin, Saravan Rajmohan, Dongmei Zhang, and Qi Zhang. 2025a. Large language model-brained gui agents: A survey. *Preprint*, arXiv:2411.18279.

Miaosen Zhang, Ziqiang Xu, Jialiang Zhu, Qi Dai, Kai Qiu, Yifan Yang, Chong Luo, Tianyi Chen, Justin Wagle, Tim Franklin, and 1 others. 2025b. Phi-ground tech report: Advancing perception in gui grounding. *arXiv preprint arXiv:2507.23779*.

Xuechen Zhang, Zijian Huang, Yingcong Li, Chenshun Ni, Jiasi Chen, and Samet Oymak. 2026. Bread: Branched rollouts from expert anchors bridge sft & rl for reasoning. *Advances in Neural Information Processing Systems*, 38:96726–96752.

Yunzhu Zhang, Zeyu Pan, Zhengwen Zeng, Shuheng Shen, Changhua Meng, and Linchao Zhu. 2025c. Mvp: Multiple view prediction improves gui grounding. *Preprint*, arXiv:2512.08529.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024. Gpt-4v(ision) is a generalist web agent, if grounded. In *Forty-first International Conference on Machine Learning*.

Hanzhang Zhou, Xu Zhang, Panrong Tong, Jianan Zhang, Liangyu Chen, Quyu Kong, Chenglin Cai, Chen Liu, Yue Wang, Jingren Zhou, and Steven Hoi. 2025a. Mai-ui technical report: Real-world centric foundation gui agents. *Preprint*, arXiv:2512.22047.

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

A Appendix

A.1 Training Details

The key optimization and training hyperparameters for the V-GRPO experiments are summarized in Table 6.

Hyperparameter	Value
deepspeed	ZeRO-3
freeze_vision_modules	true
max_prompt_length	4096
num_generations	8
per_device_train_batch_size	8
gradient_accumulation_steps	4
bf16	true
torch_dtype	bfloat16
data_seed	42
gradient_checkpointing	true
attn_implementation	flash_attention_2
num_train_epochs	100
learning_rate	1e-6
β	0.04
logging_steps	1

Table 6: Hyperparameter settings used in the V-GRPO training experiments.

A.2 Implementation Details of View-Consistent Group Rollout

Our implementation follows the target-preserving crop construction described in the main method. For each training example, we first decide whether to use view augmentation by a deterministic index-seeded Bernoulli sampler: 80% of examples use dynamic crops, while 20% remain pass-through examples with repeated original views. This pass-through branch keeps standard full-screen layouts in the training distribution.

For augmented examples, let the original image size be $W \times H$, and let the normalized ground-truth box be $B = (x_1, y_1, x_2, y_2)$. We convert it to pixel coordinates as

$$B^{\text{px}} = (x_1 W, y_1 H, x_2 W, y_2 H).$$

For each cropped view k, the initial crop size is set to $(w_k, h_k) = (0.9W,\ 0.9H)$. If the crop width or height is smaller than the target box itself, that dimension is enlarged to the box size and clipped

by the image boundary. This corner-case handling guarantees that the target UI element is never truncated, even for large elements or narrow screenshots. Each crop window is then sampled from the feasible range of top-left locations that fully contain B^{px}, ensuring target preservation across all augmented views.

We then compute the feasible integer range for the crop's top-left corner so that the full target box is inside the crop:

$$\begin{aligned} l &\in [\max(0, x_2W - w), \min(W - w, x_1W)], \\ t &\in [\max(0, y_2H - h), \min(H - h, y_1H)]. \end{aligned} \tag{13}$$

The implementation samples eight independent crop windows from these feasible ranges, matching `num_generations=8`. For each crop, the target box is remapped exactly into the crop coordinate frame,

$$\begin{aligned} B' = (&\frac{x_1W - l}{w}, \frac{y_1H - t}{h}, \\ &\frac{x_2W - l}{w}, \frac{y_2H - t}{h}), \end{aligned} \tag{14}$$

clipped to $[0, 1]$, and finally converted to the model's 0–1000 coordinate system. Each cropped image is resized again with `smart_resize` before being passed to the VLM, but the supervision coordinates are computed from the pre-resize crop geometry, so the rollout group remains view-consistent under the crop-induced coordinate transform.

Metric definitions for Table 5. For each instance, let c_0 denote the correctness on the original view and c_m denote the correctness on crop view m, $m = 1, \ldots, 8$. We define the prediction flip rate as

$$\frac{1}{8} \sum_{m=1}^{8} \mathbb{I}[c_m \neq c_0],$$

averaged over instances.

VCR is the fraction of instances for which all nine views have the same correctness label:

$$\mathbb{I}[c_0 = c_1 = \cdots = c_8].$$

A.3 Additional Ablation and Analysis

Additional Ablation: Number of Oracle Anchors Table 7 shows that one oracle anchor is sufficient and preferable. Increasing the number of supervised completions from 1 to 2 or 4 reduces ScreenSpot-Pro accuracy from 65.8 to 65.0 and 64.0. This matches the role of the anchor in Section 3: it should provide a positive direction for difficult groups while keeping most sequences in the training block model-generated. Too many anchors shift the objective toward fixed supervised imitation and weaken the group-relative exploration signal.

Table 7: **Ablation study on the number of oracle anchors.** The shaded row indicates the default single-anchor setting.

Anchors	SSV2	SSPro
1	95.5	65.8
2	95.3	65.0
4	95.2	64.0

Table 8: **Ablation study on the oracle-anchor point choice** (Qwen3-VL-8B).

Oracle Anchor Point	SSV2	SSPro
Box center	95.5	65.8
Random point in GT box	95.2	64.8

Additional Ablation: Oracle-anchor point choice. Table 8 studies whether the oracle anchor is sensitive to using the ground-truth box center. We compare the default center-point anchor against an anchor sampled uniformly from inside the ground-truth bounding box, using the same Qwen3-VL-8B setting. The random-point variant reaches 95.2 on ScreenSpot-V2 and 64.8 on ScreenSpot-Pro, close to but below the default center-point anchor at 95.5 and 65.8. This suggests that the stabilizing effect does not rely exclusively on the exact box center being the most visually salient point, since any point inside the target box remains a valid click location. At the same time, the deterministic center point avoids injecting additional target-coordinate noise and gives the best ScreenSpot-Pro accuracy, so we use it as the default oracle anchor.

Table 9: **Ablation study on the number of target-preserving views.** The shaded row indicates the default setting used by V-GRPO.

Views	SSV2	SSPro
K=1	95.3	64.8
K=2	95.2	64.8
K=4	95.5	64.4
K=8	95.5	65.8

Additional Ablation: Number of views. Table 9 studies the number of target-preserving views. Us-

ing more views increases the diversity of geometric contexts compared with the single-view setting, but the effect is not strictly monotonic because some crops can remove useful context or introduce harder visual layouts. We use K=8 in the main experiments because it obtains the best ScreenSpot-Pro accuracy, 65.8, while matching the GRPO group size used in our method.

Table 10: **Ablation study on crop strategies.** Dynamic crop provides the strongest ScreenSpot-Pro accuracy.

Crop Strategy	SSV2	SSPro
No crop	95.3	64.8
Cross-offset crop	95.7	65.0
Dynamic crop	95.5	65.8

Additional Ablation: Crop strategy. Table 10 compares different crop constructions. Cross-offset cropping gives the highest ScreenSpot-V2 number, but dynamic cropping performs best on ScreenSpot-Pro. This is consistent with our goal: the dynamic target-preserving crop is most useful on challenging high-resolution interfaces where the model must remain correct under substantial view changes.

Table 11: **Ablation study on p_{crop}.**

p_{crop}	SSV2	SSPro
1.0	95.1	64.6
0.8	95.5	65.8
0.6	95.6	65.2

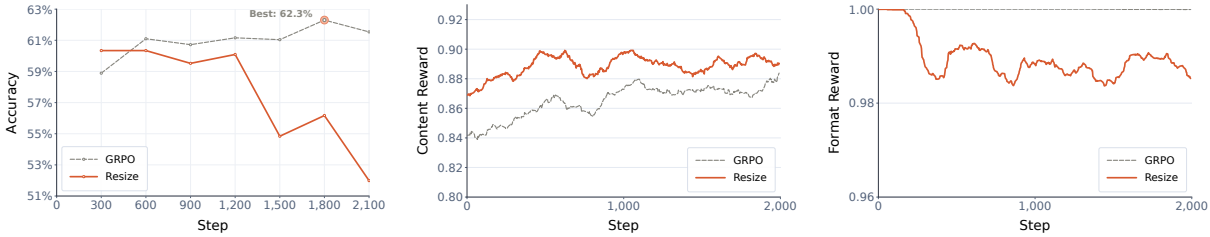
Additional Ablation: p_{crop}. Table 11 studies the probability of applying target-preserving crop augmentation during training. As described in Section 3, p_{crop} controls the mixture between view-consistent cropped groups and pass-through full-screen groups. A larger value exposes the policy to more crop-induced coordinate transformations, while a smaller value retains more original screenshots and reduces train–test mismatch. The results show that using crops for every example is not optimal: setting p_{crop}=1.0 lowers ScreenSpot-Pro accuracy to 64.6, suggesting that removing full-screen pass-through examples makes the RL update overly dominated by cropped views. Reducing the probability to p_{crop}=0.6 slightly improves ScreenSpot-V2 to 95.6 but weakens ScreenSpot-Pro to 65.2, indicating insufficient view diversity for the harder high-resolution benchmark. We therefore use p_{crop}=0.8 in the main experiments, which achieves the best ScreenSpot-Pro accuracy, 65.8, while maintaining strong ScreenSpot-V2 accuracy, 95.5.

Additional Ablation: Image processing strategy. We further study whether the benefit of view-consistent rollout comes from using multiple image inputs in general, or specifically from changing the target coordinates through view transformation. To this end, we conduct an additional training-dynamics experiment initialized from Qwen3-VL-4B. We compare the GRPO baseline with a simpler multi-image resize strategy, denoted as GRPO + multi-image resize. In this variant, each GRPO group is constructed from multiple resized versions of the same full screenshot. Unlike target-preserving cropping, resizing changes the visual scale but largely preserves the target element's relative coordinate in the normalized image space. Figure 4 reports the ScreenSpot-Pro accuracy, content reward, and format reward during training. ScreenSpot-Pro accuracy is evaluated every 300 steps starting from step 300.

Although multi-image resize slightly outperforms the GRPO baseline at the first evaluation point, its performance quickly becomes unstable and then degrades substantially. The baseline GRPO curve remains relatively stable, ending at 61.54, whereas GRPO + multi-image resize drops from 60.34 at step 300 to 51.99 at step 2100. The final gap reaches 9.55 points.

We hypothesize that this failure is caused by the lack of coordinate variation in the resized views. Resizing the full screenshot changes the visual scale and image tokenization, but it does not change the target element's relative location in the normalized coordinate system. Therefore, different resized inputs in the same GRPO group correspond to nearly the same correct coordinate output. In this case, the group mainly contains multiple scale variants with an unchanged answer, so the model is not forced to learn how the output coordinate should transform with the input view.

In contrast, target-preserving cropping changes the target element's relative position in the cropped coordinate frame. For the same instruction, different cropped views generally require different coordinate outputs after exact box remapping. This creates a stronger cross-view grounding signal: the model must localize the same semantic target under different visual contexts and produce coordinates that are consistent with the crop-induced coordinate transform. This explains why dynamic cropping provides useful view-consistent supervision, whereas naive multi-image resize gives largely redundant outputs and leads to unstable training.



(a) ScreenSpot-Pro accuracy ↑. (b) Content reward ↑. (c) Format reward ↑.

Figure 4: **Resize strategy training dynamics and reward diagnostics.**

Table 12: Performance comparison on the **ScreenSpot-Pro.** * denotes our evaluated results.

Model	CAD		Dev.		Creative		Scientific		Office		OS		Avg.
	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	Text	Icon	
≈4B													
Qwen3-VL-4B* (Bai et al., 2025)	53.3	18.8	78.6	31.7	70.7	22.4	75.7	30.0	83.1	39.6	76.6	33.7	55.5
GRPO-4B	65.5	25.0	79.9	40.0	71.2	28.7	83.3	40.0	90.4	47.2	74.8	39.3	61.5
V-GRPO-4B	64.0	28.1	81.1	45.2	72.7	33.6	81.9	40.0	89.8	54.7	76.6	40.5	63.4
+ *MVP*	79.2	51.6	87.2	53.4	74.2	48.3	85.4	50.0	92.1	73.6	78.5	61.8	71.8
≈8B													
UI-TARS-7B (Qin et al., 2025b)	20.8	9.4	58.4	12.4	50.0	9.1	63.9	31.8	63.3	20.8	30.8	16.9	35.7
Phi-Ground (Zhang et al., 2025b)	26.9	17.2	70.8	16.7	56.6	13.3	58.0	29.1	76.4	44.0	55.1	25.8	43.2
GUI-Actor-7B (Wu et al., 2025)	47.7	9.4	59.1	15.9	59.6	16.1	70.1	25.5	69.5	41.5	55.1	19.1	44.6
SE-GUI-7B (Yuan et al., 2025)	51.3	14.1	68.2	19.3	57.6	9.1	75.0	28.2	78.5	43.4	49.5	25.8	47.2
GUI-G^2-7B (Tang et al., 2025a)	55.8	12.5	68.8	17.2	57.1	15.4	77.1	24.5	74.0	32.7	57.9	21.3	47.5
OpenCUA-7B (Wang et al., 2025c)	-	-	-	-	-	-	-	-	-	-	-	-	50.0
GTA1-7B (Yang et al., 2025)	53.3	17.2	66.9	20.7	62.6	18.9	76.4	31.8	82.5	50.9	48.6	25.9	50.1
UI-Venus-7B (Gu et al., 2025a)	60.4	21.9	74.7	24.1	63.1	14.7	76.4	31.8	75.7	41.5	49.5	22.5	50.8
InfiGUI-G1-7B (Liu et al., 2025)	57.4	23.4	74.7	24.1	64.6	18.2	80.6	31.8	75.7	39.6	57.0	29.2	51.9
GUI-Owl-7B (Ye et al., 2025)	64.5	21.9	76.6	31.0	59.6	27.3	79.1	37.3	77.4	39.6	59.8	33.7	54.9
MAI-UI-8B (Zhou et al., 2025a)	72.6	35.9	83.8	52.4	76.3	33.6	79.9	37.3	88.7	60.4	76.6	49.4	65.8
Qwen3-VL-8B* (Bai et al., 2025)	56.9	10.9	75.3	22.8	68.2	16.1	78.5	32.7	80.8	39.6	71.0	20.2	52.7
GRPO-8B	73.6	31.2	82.5	39.3	77.8	28.0	81.9	40.0	88.7	49.1	73.8	40.4	63.4
V-GRPO-8B	74.6	34.4	85.3	45.9	73.7	29.4	81.9	40.9	91.5	58.5	76.6	42.7	65.8
+ *MVP*	81.7	45.3	89.1	58.2	76.3	42.7	86.1	56.9	92.1	71.7	80.4	53.9	72.0
≥30B													
Qwen3-VL-32B* (Bai et al., 2025)	60.4	28.1	69.5	22.1	75.8	25.2	84.7	25.5	85.9	43.4	62.6	15.7	54.9
OpenCUA-32B (Wang et al., 2025c)	-	-	-	-	-	-	-	-	-	-	-	-	55.3
GUI-Owl-32B (Ye et al., 2025)	62.4	28.1	84.4	39.3	65.2	18.2	82.6	39.1	81.4	39.6	70.1	36.0	58.0
GTA1-32B (Yang et al., 2025)	43.7	23.4	82.5	28.3	69.2	14.7	79.9	31.8	80.8	43.4	70.1	32.6	63.6
MAI-UI-32B (Zhou et al., 2025a)	70.1	45.3	86.4	40.7	82.8	37.8	91.7	46.4	90.4	71.7	78.5	34.8	67.9
UGround-v1-72B (Gou et al., 2025)	16.8	4.7	55.8	4.8	54.0	10.5	70.8	22.7	61.0	18.9	40.2	7.9	34.5
UI-Tars-72B (Qin et al., 2025b)	18.8	12.5	63.0	17.2	57.0	15.4	64.6	20.9	63.3	26.4	42.1	15.7	38.1
UI-Venus-72B (Gu et al., 2025b)	66.5	29.7	84.4	33.1	73.2	30.8	84.7	42.7	83.1	60.4	75.7	36.0	61.9
Qwen3-VL-30A3B* (Bai et al., 2025)	51.8	15.6	76.0	24.8	69.2	20.3	76.4	27.3	80.8	37.7	75.7	38.2	53.7
GRPO-30A3B	69.5	29.7	84.4	51.7	74.7	33.6	81.9	40.0	89.3	52.8	83.2	53.9	65.9
V-GRPO-30A3B	71.1	32.8	85.9	52.1	76.8	35.7	83.3	38.2	87.0	56.6	78.5	50.6	67.0
+ *MVP*	82.7	45.3	89.7	58.2	79.8	51.8	86.8	49.1	92.1	71.7	83.2	60.7	74.1

Detailed ScreenSpot-Pro results. Table 12 provides the category-level ScreenSpot-Pro breakdown behind the aggregate SSPro numbers in Table 1. Across the 4B, 8B, and 30A3B settings, V-GRPO improves the corresponding GRPO baseline by 1.3, 2.4, and 1.1 average points, respectively. Adding MVP further raises the averages to 71.8, 72.0, and 74.1, showing that the view-consistent training signal remains complementary to inference-time multi-view aggregation.