ONE PIXEL WIDER, A DIFFERENT CLICK: HOW IMAGE PREPROCESSING DESTABILIZES GUI GROUNDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Stable GUI grounding is essential for reliable interaction by computer-use agents. However, we find that changing a screenshot dimension by a single pixel can alter which UI element a model selects, with these changes occurring disproportionately at particular image sizes. Why can one pixel produce a different click? This paper identifies dynamic-resolution preprocessing, which maps raw screenshot dimensions to discrete visual token grids or tile layouts, as one source of this instability. We call the image sizes where these configurations change *preprocessing transition points*. Across eight models and three benchmarks, the average UI-element switching rate nearly doubles at these points relative to adjacent one-pixel changes. Controlled interventions confirm that these configuration changes contribute to the excess switching. Further analysis in two models shows that resizing and content alignment affect grounding beyond changes in token count alone. These findings motivate three investigations. First, because switching concentrates at transition points, we show how to "steal" hidden preprocessing settings from clicks. The inferred settings help predict unstable screenshot sizes, with predictions validated on unseen screenshots. Second, we use sensitivity to small movements of interface content to detect errors: clicks whose selected UI element changes under these movements are more likely to be incorrect. Combining this consistency check with model confidence improves error detection. Finally, guided by the effects of resizing and content alignment, we design a mitigation method that improves grounding stability. Together, these analyses and investigations offer insights into why one pixel can change a click and how to improve the stability of GUI grounding.

1 INTRODUCTION

GUI grounding enables computer-use agents to locate and click interface elements specified by user instructions (Cheng et al., 2024). Stable grounding is essential for reliable interaction: small changes in how a screenshot is presented should not redirect an agent to a different control when the instruction and task-relevant interface remain unchanged. Recent work has shown that adding borders to screenshots can move predicted clicks and change whether they are correct (Zhang et al., 2026). In this paper, we observe that a one-pixel change in screenshot dimensions may alter the selected UI element. Fig. 1(a) provides an illustrative example: adding a single pixel to the screenshot width may redirect the predicted click to a different button, even though the instruction, "Delete the receipt from X," and the task-relevant interface content remain unchanged. This raises the question: *why can one pixel produce a different click?* Here, a "different click" means only a switch between UI elements.

This sensitivity matters in practice because screenshot dimensions vary across devices or window sizes. Average accuracy, the standard benchmark metric, can conceal the resulting instability: correct-to-incorrect and incorrect-to-correct changes can offset each other (Fig. 1(b); App. G). Understanding when and why clicks change is therefore important for reliable GUI grounding.

To understand this sensitivity, we examine how screenshots are transformed before they reach the model and identify dynamic-resolution preprocessing as one source of this instability. Consider Fig. 1(c): increasing the screenshot width from 1071 to 1072 pixels changes the processed width from 1056 to 1088 pixels under nearest rounding to multiples of 32. At a fixed height of 768 pixels, this

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.

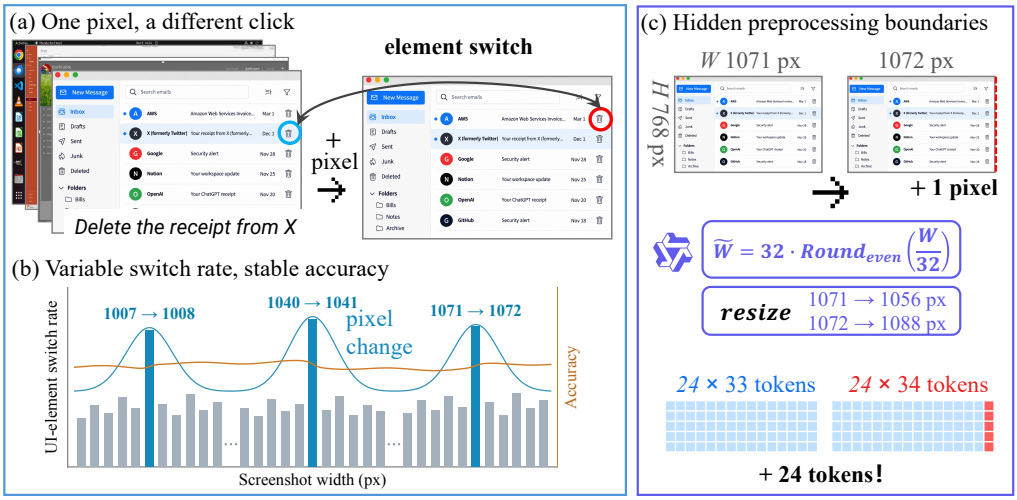


Figure 1: **One-pixel click instability and preprocessing transition points.** (a) An illustrative example shows how adding one pixel to the screenshot width can redirect the click to a different UI element under the same instruction. (b) A schematic shows peaks in UI-element switching (blue) alongside nearly unchanged average accuracy (orange, right axis). (c) A preprocessing example: nearest rounding to multiples of 32 maps widths of 1071 and 1072 pixels to 1056 and 1088 pixels, respectively. With height fixed at 768 pixels and pixel-budget scaling inactive, the grid has $768/32 = 24$ token rows. Adding one column therefore adds 24 visual tokens.

adds one column of 24 visual tokens, although the task-relevant interface content remains unchanged. Such changes follow directly from preprocessing rules, but their consequences for grounding behavior are less clear. More generally, dynamic-resolution preprocessing maps screenshot dimensions to discrete visual token grids or tile layouts. We call the image sizes where these configurations change *preprocessing transition points*. We investigate whether these transitions concentrate UI-element switching and whether the configuration changes themselves cause the excess instability.

To answer these questions, we first locate transition points from the preprocessing rules, without observing model outputs. We compare one-pixel size changes that cross these points with adjacent one-pixel changes that preserve the configuration, keeping the instruction, target, and GUI content fixed. Across three benchmarks, eight models, and four preprocessing families, the average UI-element switching rate **nearly doubles** at predicted transition points (Fig. 3). Preprocessing interventions with fixed model weights confirm that configuration changes contribute to this excess switching. We next examine how they alter the visual input beyond token count. In Qwen3-VL and UI-Venus-1.5, crossing a transition point rescales content by about 2%, and resizing by the same amount within a fixed grid produces comparable switching. Moving content within a fixed grid also changes element selection, and replacing features outside the target region can alter clicks. These results identify resizing, content alignment, and context as factors for improving grounding stability.

These findings motivate three further investigations. First, because switching concentrates at transition points, we infer hidden rules from click outputs alone. We design a one-pixel padding probe to infer token stride and rounding rules and predict unstable screenshot sizes. For four open-weight models, the inferred rules predict actual preprocessing transitions across all tested held-out size comparisons. For two closed-weight models, the inferred token stride matches the 32-pixel patch size in public API documentation, and the full inferred rule matches the documented rule for GPT-5.4 nano.

Second, we use sensitivity to content alignment to identify unreliable answers. A click is more likely to be incorrect when moving the interface content by a few pixels changes the selected UI element. Combining this consistency check with token confidence improves error-detection AUROC by 0.039–0.040 in Qwen3-VL and UI-Venus-1.5, with gains across all six evaluated models.

Finally, we introduce *Grid-Phase Canonicalization* (GridPC) to improve grounding stability. Guided by the effects of resizing and content alignment, GridPC reduces size-dependent resizing and uses padding to align content with the token grid. It requires no training and one forward pass, and reduces UI-element switching and correctness flips across scans of padding amounts and content positions.

2 Preprocessing Transitions and Evaluation Framework

We use the illustrative example in Fig. 2, with screenshot widths of 1070–1073 pixels, to explain preprocessing transitions and the matched one-pixel comparisons used in our evaluation.

Configurations and transitions. Let $G_\phi(H, W)$ denote the *preprocessing configuration* selected by rule ϕ for a screenshot of height H and width W: its visual token grid or tile layout. A *preprocessing transition point* occurs between W and $W + 1$ when $G_\phi(H, W) \neq G_\phi(H, W + 1)$, with height transition points defined analogously. Fig. 2 illustrates a width transition for Qwen3-VL, whose effective token stride is $Q = 32$ pixels (Bai et al., 2025a). With pixel-budget scaling inactive, nearest rounding with ties to even maps widths 1071 and 1072 to 1056 and 1088, respectively. The number of visual token columns changes from 33 to 34. We locate transition points using the full preprocessing rule for each model, including pixel-budget scaling and pre-resize. App. B.1 gives the derivations.

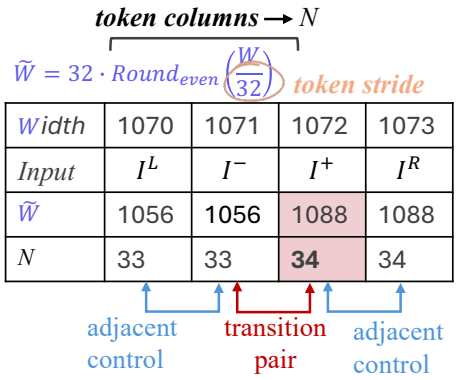


Figure 2: **Transition pair and adjacent controls.** W and $\widetilde{W}$ denote raw and processed widths; $N = \widetilde{W}/32$ is the number of token columns. Only (I^-, I^+) changes N; both adjacent controls preserve it. Every pair differs by one pixel. Pixel-budget scaling is inactive in this illustrative example.

Matched one-pixel comparisons. We vary padding along the right or bottom border. Each window contains four consecutive padding amounts, producing (I^L, I^-, I^+, I^R). The middle *transition pair*, (I^-, I^+), crosses a predicted transition point. The two *adjacent controls*, (I^L, I^-) and (I^+, I^R), preserve the configuration. Every comparison changes the canvas by one pixel; the instruction, target box, and original GUI content stay fixed. In Fig. 2, input widths are 1070, 1071, 1072, and 1073 pixels. Only the middle change, 1071→1072, crosses the transition point; the two outer changes preserve the configuration. We average both controls for comparison with the transition pair.

Metrics and aggregation. We use OmniParser V2 (Lu et al., 2025) to detect UI regions, then assign clicks to UI elements using the rules in App. A.2. We count a switch only when both clicks are assigned to different UI elements; pairs with an unassigned click or unparsed prediction remain in the denominator but do not count as switches. *Excess switching* is the transition-pair switch rate minus the mean switch rate of the two adjacent controls. Formal definitions, assignment details, and human validation are in App. A.2. *Relative click distance* is the Euclidean distance between predictions as a percentage of the original screenshot diagonal. A click is correct if it falls inside the target box; we use box centers for box outputs and count unparsed answers as incorrect. Unless stated otherwise, we average annotations within screenshots and weight screenshots equally; models receive equal weights. Confidence intervals use screenshot-cluster bootstrap resampling (App. A.3).

Data and models. The primary panel contains 2,010 targets on 1,450 screenshots: 600 ScreenSpot-Pro annotations (Li et al., 2025), 900 UI-Vision annotations (Nayak et al., 2025) (300 each from Basic, Functional, and Spatial), and 510 OSWorld-G annotations (Xie et al., 2026). We evaluate Qwen3-VL-8B (Bai et al., 2025a), Qwen2.5-VL-3B (Bai et al., 2025b), InternVL3.5-8B (Wang et al., 2025), UI-Venus-1.5-8B (Team et al., 2026), GTA1-7B (Yang et al., 2026), UI-TARS-1.5-7B (Qin et al., 2025), GUI-Actor-7B (Wu et al., 2026), and LocateAnything-3B (Wang et al., 2026b). They cover four preprocessing families, sharing basic patch or tiling rules and represented by Qwen3-VL, Qwen2.5-VL, LocateAnything, and InternVL3.5. We subsequently omit parameter counts unless needed to distinguish model variants. The primary study uses greedy decoding or GUI-Actor's top-ranked spatial prediction. App. A specifies sampling, checkpoints, and inference settings.

3 Dissecting Click Instability at Preprocessing Transitions

We test the structural explanation by locating excess switching, suppressing configuration changes on the same inputs, and moving transition points with model weights fixed.

3.1 Testing the Causal Contribution of Preprocessing Transitions

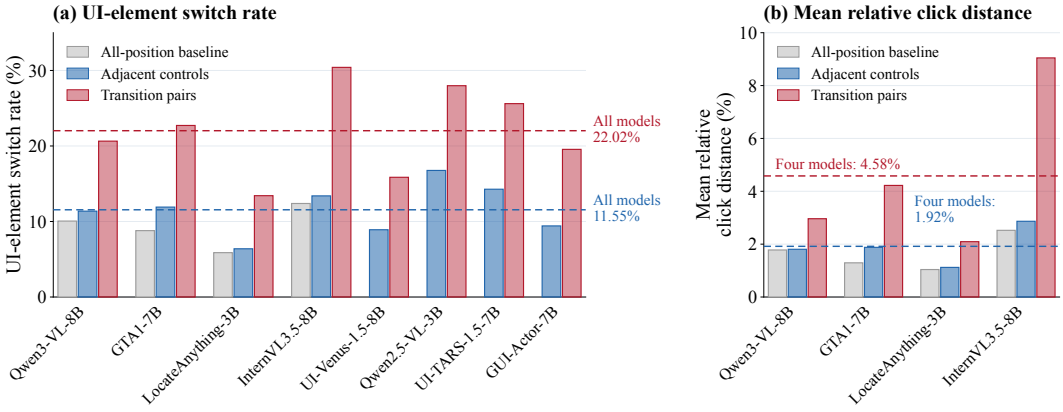


Figure 3: **UI-element switching and click distance under one-pixel size changes.** (a) UI-element switching across eight models. (b) Mean relative click distance for four models. Blue and red show adjacent controls and transition pairs on the full primary panel. Gray shows an all-position baseline from separate scans, averaging all 64 one-pixel steps per scan, including transitions. Dashed lines show equally weighted model means for the blue and red bars.

Instability concentrates at predicted transition points. We compare one-pixel changes that cross predicted transition points with adjacent one-pixel changes that preserve the configuration. Across eight models, average UI-element switching rises **from 11.55% to 22.02%**, nearly doubling (Fig. 3(a)). The paired increase is 10.47 percentage points (95% CI $[9.81, 11.14]$). Transition pairs show higher switching than adjacent controls in all eight models, despite both comparisons changing the screenshot by only one pixel. This gap indicates elevated sensitivity at preprocessing transition points. Fig. 3(b) shows higher relative click distance for the four displayed models. Across all eight models, mean relative click distance rises from 2.45% to 4.95%, a paired increase of 2.49% of the screenshot diagonal (95% CI $[2.29, 2.70]$). Transition pairs also exceed the all-position baseline in switching and click distance for all four scanned models (gray in Fig. 3). Selection probabilities change more at transitions (App. B.5).

Fixing preprocessing configurations reduces switching. We test whether configuration changes contribute to excess switching by assigning the same preprocessing configuration to both screenshots, with model weights fixed. Controlled resizing slightly changes the baseline relative to standard preprocessing (22.15% versus 22.02%). Within this controlled procedure, fixing the configuration reduces average switching **from 22.15% to 11.70%**, a reduction of 10.45 percentage points (95% CI $[9.79, 11.13]$), close to the 11.55% adjacent-control rate. Switching decreases across all eight models; the design and detailed results are in App. B.3.

Native $\widetilde{W} = 32\,\mathrm{Round}_{\mathrm{even}}\left(\frac{W}{32}\right)$		**Modified** $\widetilde{W} = 32\,\mathrm{Round}_{\mathrm{even}}\left(\frac{W-16}{32}\right)$	
Input W (px)	Output $\widetilde{W}$ (px)	Input W (px)	Output $\widetilde{W}$ (px)
$1071 \rightarrow 1072$	$1056 \rightarrow 1088$	$1071 \rightarrow 1072$	$1056 \rightarrow 1056$
$1087 \rightarrow 1088$	$1088 \rightarrow 1088$	$1087 \rightarrow 1088$	$1056 \rightarrow 1088$

Table 1: **From Native to Modified preprocessing.** Native has a transition point at 1071→1072 pixels; Modified subtracts 16 before rounding, shifting it to 1087→1088 pixels. Model weights are fixed. Qwen3-VL uses ties-to-even rounding at height 768 px, without pixel-budget scaling.

Model	Native transition point ↓	Shifted transition point ↑
Qwen3-VL	7.90 [5.90, 9.94]	8.25 [6.18, 10.33]
Qwen2.5-VL	11.17 [9.02, 13.39]	11.69 [9.52, 13.90]
LocateAnything	7.63 [5.91, 9.37]	8.87 [7.10, 10.65]
InternVL3.5	16.87 [14.52, 19.19]	16.32 [13.97, 18.69]
Average	**10.89** [9.84, 11.95]	**11.28** [10.19, 12.36]

Table 2: **Excess switching follows the shifted transition point.** Changing from Native to Modified decreases excess switching at the native transition point (↓) and increases it at the shifted point (↑). Values are paired changes in percentage points; brackets indicate 95% CIs.

Moving transition points moves the instability. To test whether excess switching follows the preprocessing rule, we move its transition points while keeping model weights fixed. We evaluate Qwen3-VL, Qwen2.5-VL, LocateAnything, and InternVL3.5, one representative from each preprocessing family, on the full primary panel. We compare *Native* preprocessing with a *Modified* rule that

selects $G_\phi(H, W-16)$ before processing the actual screenshot, moving each width transition point 16 pixels to the right; height transition points are shifted analogously (Table 1). We call the points under Native and Modified preprocessing the *native* and *shifted transition points*, respectively. Both rules are evaluated on the same inputs and adjacent controls around both sets of points. Averaged equally across the four models, changing from Native to Modified reduces excess switching at the native transition points by **10.89 percentage points** (95% CI $[9.84, 11.95]$) and increases it at the shifted transition points by **11.28 percentage points** (95% CI $[10.19, 12.36]$). Both effects have positive 95% confidence intervals in every model (Table 2).

Excess switching concentrates at preprocessing transitions, decreases when configurations are fixed, and moves when transition points move. These experiments identify preprocessing transitions as a cause of grounding instability. Sec. 3.2 identifies which change produces this effect.

3.2 Why Do Preprocessing Transitions Change Clicks?

The preceding interventions establish a causal contribution of preprocessing transitions, but do not separate the changes they introduce. Crossing a transition point can (1) resize the screenshot and (2) enlarge the token grid. We investigate these effects in Qwen3-VL and UI-Venus-1.5, which use token-grid preprocessing, then examine content alignment and context.

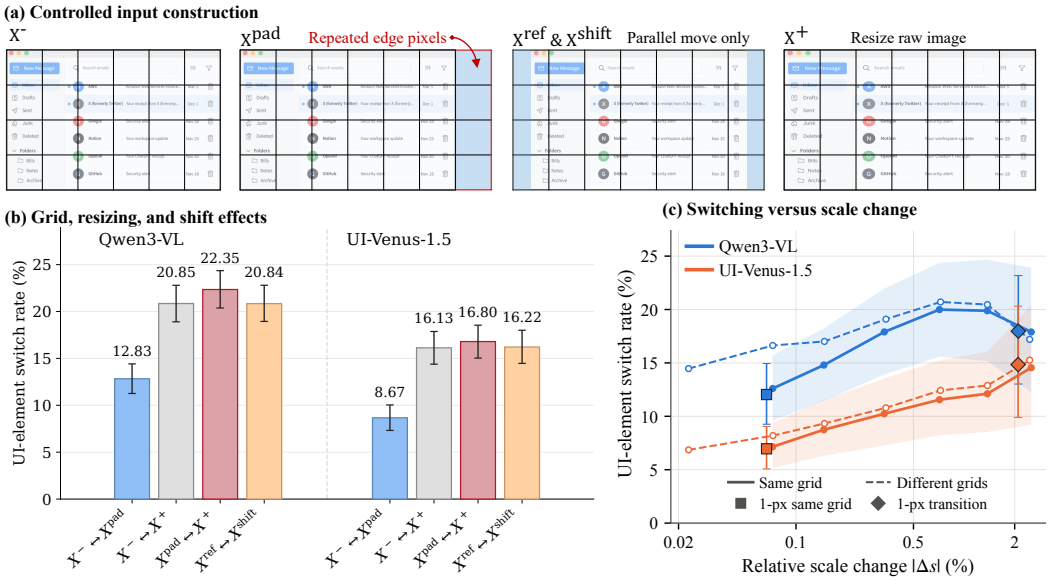


Figure 4: **Grid, resizing, and scale effects.** (a) Inputs from the same screenshot, with repeated-edge padding. X^{ref} and X^{shift} differ only in content position, with grid and content size fixed. (b) Switching for the labeled comparisons. (c) Switching versus relative scale change $|\Delta s|$, the relative change in content scale along the padded axis. Solid/dashed lines show same/different grids; squares/diamonds show one-pixel same-grid/transition pairs. Shading and error bars show 95% CIs.

Probing grid and resizing effects. Fig. 4(a) illustrates three processed inputs constructed from the same raw screenshot: X^- resizes it to the smaller grid, X^+ resizes it to the larger grid, and X^{pad} extends X^- to the larger grid by repeating edge pixels. Comparing X^- with X^{pad} adds tokens while preserving content pixels, comparing X^{pad} with X^+ changes the content scale at a fixed grid, and X^- versus X^+ changes both. All inputs bypass further resizing, and clicks are mapped back to raw coordinates (App. C.1). For Qwen3-VL, **changing scale at a fixed grid switches 22.35% of clicks** $[20.37, 24.36]$, comparable to 20.85% for the full comparison. Grid expansion with existing pixels preserved produces less switching (12.83%, 95% CI $[11.26, 14.42]$). UI-Venus-1.5 shows the same pattern (Fig. 4(b)). Resizing therefore matters more than the added tokens.

Transitions act through the jump in scale. Adjacent controls also rescale content, but only slightly: a one-pixel size change alters content scale by about 0.07% within a configuration (squares in Fig. 4(c)), versus 2.1% on average at a transition, where the processed size jumps by one stride.

Using a separate 96-screenshot padding-scan panel, we test whether this larger scale change explains the excess switching. Even within a fixed grid, switching rises with relative scale change $|\Delta s|$ up to about 1% (Fig. 4(c)). We compare one-pixel transition pairs with same-grid pairs matched for $|\Delta s|$. Switching rates for Qwen3-VL are 17.97% and 18.08%, respectively; for UI-Venus-1.5, they are 14.84% and 13.00%. Thus, comparable scale changes within a fixed grid produce switching rates close to those at transitions, supporting the jump in resize scale as a major contributor to excess switching. App. C.3 provides confidence intervals and detailed comparisons.

Resizing at a fixed token grid produces switching comparable to the full transition. The jump in resize scale accounts for much of the observed excess switching; we next examine the accompanying changes in content size and alignment.

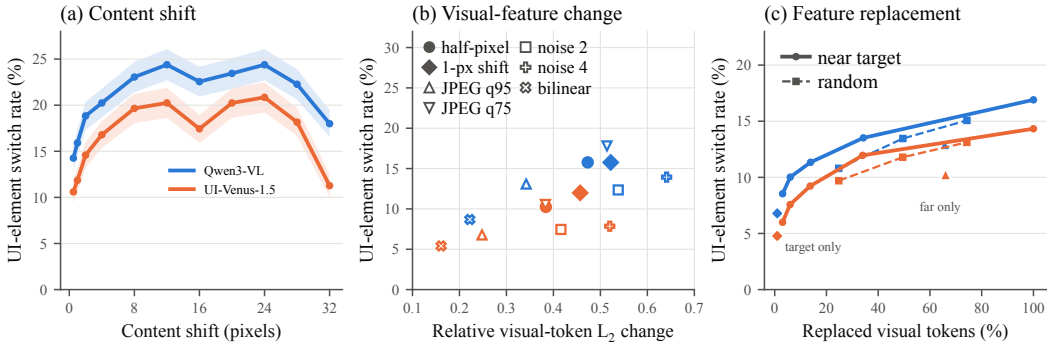


Figure 5: **Alignment, features, and context.** (a) Fixed-grid content shifts; shading shows 95% CIs. (b) Switching versus visual-token change: geometric (filled) and non-geometric (open) perturbations. (c) Reference features replaced with those from a target-anchored magnified input: outward from the target (solid) or at random context positions (dashed). Diamonds: target only; triangles: over 16 tokens from the target. Colors denote models as in (a).

Content alignment affects grounding. Resizing changes both content size and alignment with token-grid boundaries. Shifting content by the target displacement induced by resizing, without changing its size, switches 20.84% of Qwen3-VL clicks and 16.22% of UI-Venus-1.5 clicks, close to the full transition (X^{ref} versus X^{shift} in Fig. 4(b)). Magnification with the target center fixed still switches 17.29% and 13.85%, respectively (App. C.2). Both displacement and magnification can therefore change clicks. To examine alignment more generally, we shift content within a fixed grid using all 2,010 target annotations. These shifts change the pixels grouped within patches (Ercan, 2026). We call the position of content within one token stride its *grid phase*; a full-stride shift restores the alignment of interior content. Fig. 5(a) compares each shifted input with the unshifted reference, mapping clicks back to original coordinates. Shifts of 8–24 pixels switch 22.6–24.4% of Qwen3-VL clicks and 17.4–20.9% of UI-Venus-1.5 clicks. At 32 pixels, one effective stride in both models, switching falls relative to a 24-pixel shift by 6.39 percentage points for Qwen3-VL (95% CI $[4.96, 7.79]$) and 9.58 points for UI-Venus-1.5 ($[8.14, 11.00]$). **A larger displacement can therefore cause less switching when it restores grid alignment.**

Feature change and context. We next ask whether switching is explained by the magnitude of visual-feature change across geometric and non-geometric perturbations. Across content shifts, JPEG compression, Gaussian noise, and interpolation changes, switching generally increases with relative L_2 change in visual tokens, but similar feature distances can yield different switching rates (Fig. 5(b)). We therefore examine where feature changes affect the prediction. Using activation patching (Zhang & Nanda, 2024), we replace selected visual features of a reference input with those from a target-anchored magnified input on the same grid. Replacing only target-region features, with context unchanged, switches 6.79% of Qwen3-VL clicks and 4.78% of UI-Venus-1.5 clicks. **Replacing only features more than 16 tokens from the target still switches 12.93% and 10.25%, respectively** (Fig. 5(c)), showing that distant context can alter clicks while target-region features remain unchanged. Apps. C.4–C.5 give implementation checks and tile-based results.

Grounding depends on how content aligns with the token grid and on visual features outside the target region. Preserving target-region features alone does not ensure a stable click.

4 USING PREPROCESSING SENSITIVITY TO IMPROVE GUI GROUNDING

The preceding findings suggest three practical uses. Because switching concentrates at transition points, we infer hidden rules from click outputs alone (Sec. 4.1). Sensitivity to small movements of interface content suggests a signal for identifying unreliable answers (Sec. 4.2). The effects of resizing and content alignment motivate GridPC to stabilize both during preprocessing (Sec. 4.3).

4.1 PROBING HIDDEN PREPROCESSING SETTINGS FROM CLICK OUTPUTS

When preprocessing rules are unavailable, inferring them from clicks could help predict which screenshot sizes are likely to produce unstable answers.

Target: predict the rule family. We infer hidden preprocessing settings from clicks alone, using the rule

$$\widetilde{W} = Q \cdot \mathrm{Round}^{\text{rounding}}_{\text{tie-breaking}}\left(\frac{W}{Q}\right), \tag{1}$$

where Q is the candidate token stride in pixels, rounding is nearest, up, or down, and tie-breaking resolves nearest-rounding midpoints (up or to even). The same rule applies to height H.

Probe data. We add 0–64 pixels of repeated-edge padding to the right or bottom border in one-pixel steps, recording clicks while keeping the instruction and original content fixed. We use 300 shared screenshots, with each benchmark contributing 50 screenshots for inferring settings and 50 for held-out validation. Fig. 6 shows the three steps: vary padding, collect clicks, and select the candidate rule whose predicted transitions align with large click movements (Rule B rather than Rule A).

Inferring settings. Following Eq. 1, we search $Q \in \{8, \ldots, 64\}$ and rounding $\in \{\text{nearest}, \text{up}, \text{down}\}$. For nearest rounding, tie-breaking $\in \{\text{up}, \text{even}\}$; for up or down rounding, tie-breaking is not applicable (N/A). Each rule is a combination Rule $= (Q, \text{rounding}, \text{tie-breaking})$, giving $57 \times 4 = 228$ candidate rules. Each rule divides adjacent one-pixel steps into those crossing predicted transition points and those at other positions. For each valid one-pixel comparison i, $z_i \in \{0, 1\}$ indicates whether the click moves by more than 1% of the original screenshot diagonal (a large-movement event). This requires only click coordinates, without UI-element assignment. On the discovery set, we score how well each rule explains these events using a two-rate Bernoulli likelihood, with one rate for each group. Among candidate rules with a higher estimated event rate at predicted transition points, we select

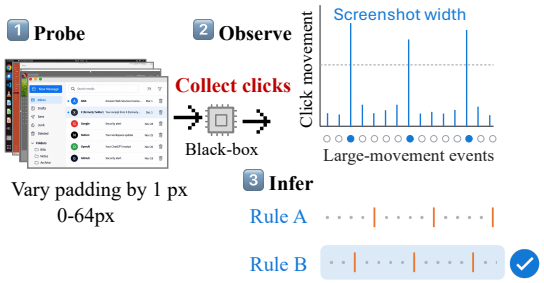


Figure 6: **A one-pixel padding probe for hidden preprocessing settings.** We vary screenshot padding, record click movements, and select the candidate rule that best explains large-movement events. Curves and candidate rules are schematic.

$$\widehat{\text{Rule}} = \arg\max_{\text{Rule}} \log P(\{z_i\} \mid \text{Rule}). \tag{2}$$

On held-out screenshots, we fix the rule and test whether large movements are more frequent at predicted transitions. Rule stability is the percentage of 10,000 bootstrap resamples of discovery screenshots that recover the selected rule. We resample within each benchmark and repeat the rule search; this measures selection consistency, not prediction accuracy. App. D gives the scoring formula and statistical procedures.

Results. We evaluate Qwen3-VL, GTA1, Qwen2.5-VL-7B, UI-Venus-1.5, and two closed-weight models, GPT-5.4 nano (OpenAI, 2026b) and GPT-5.6 Luna (OpenAI, 2026a). The inferred rules predict higher large-movement rates at transition points on held-out screenshots for all four open-weight models and GPT-5.4 nano, with confidence intervals above zero (Table 3). For all four open models, they also correctly predict actual grid changes across all 9,600 held-out size comparisons per model. Compared with public API documentation, the full inferred rule matches for GPT-5.4 nano, and the token stride matches for GPT-5.6 Luna. Because the probe needs only click coordinates, it also applies to API-only models. The inferred rule can flag unstable screenshot sizes, supply the token stride that GridPC requires (Sec. 4.3), and track preprocessing changes after model updates.

Table 3: **Settings inferred by the one-pixel padding probe.** Increase compares large-movement rates at predicted transition points and other positions on 150 held-out screenshots, with 95% bootstrap CIs. Rows above and below the divider are open- and closed-weight models. Green marks settings that match actual grid changes (✓) or API documentation (Doc.); blue marks a mismatch.

Model	Token stride Q (px)	Rounding	Tie-breaking	Rule stability (%)	Increase (pp) [95% CI]	Verified
Qwen3-VL	32	Nearest	Ties to even	88.25	10.34 [5.68, 14.99]	✓
GTA1	28	Nearest	Ties to even	99.94	10.76 [6.19, 15.39]	✓
Qwen2.5-VL-7B	28	Nearest	Ties to even	99.85	17.84 [12.40, 23.26]	✓
UI-Venus-1.5	32	Nearest	Ties to even	85.93	9.90 [5.74, 14.37]	✓
GPT-5.4 nano	32	Up (ceiling)	N/A	67.60	15.86 [10.85, 20.92]	Doc.
GPT-5.6 Luna	32	Down (floor)	N/A	40.97	4.64 [−0.04, 9.40]	Doc.

4.2 Flagging Unreliable Clicks with Shift Consistency

Consistency probe. Prior VQA work finds that stability under benign perturbations predicts correctness (Rosenfeld et al., 2025). We shift content right by a quarter, half, and three quarters of an effective token stride, repeating left-edge pixels while keeping the grid or tile layout fixed (App. E). The inconsistency score counts probes selecting a different UI element from the unshifted answer or no element; zero indicates consistency. This requires three extra greedy forward passes and no training. We combine this score with *token confidence*, the geometric mean probability of generated answer tokens (Tang et al., 2026), by averaging their risk ranks without fitting parameters. Higher inconsistency and lower confidence indicate greater risk.

Evaluation. We evaluate six models on all 2,010 target annotations: Qwen3-VL, UI-Venus-1.5, Qwen2.5-VL-3B, GTA1, UI-TARS-1.5, and InternVL3.5. We pool annotations to measure error detection with the area under the ROC curve (AUROC; higher is better) and accuracy on the lowest-risk half of answers. Intervals are 95% screenshot-cluster bootstrap intervals.

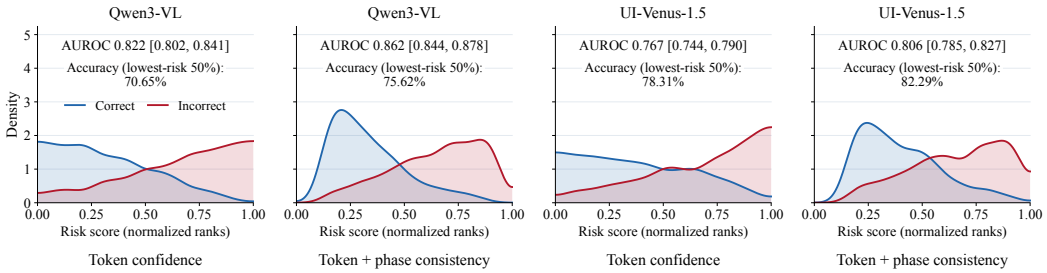


Figure 7: **Consistency under content shifts improves error detection.** Better error detection places correct answers (blue) at lower risk and incorrect answers (red) at higher risk. Curves are smoothed and normalized by class; AUROC and 95% CIs use unsmoothed scores. Accuracy uses the lowest-risk 50% of answers. Scores are normalized ranks, not error probabilities.

Results. Fig. 7 shows that combining shift consistency with token confidence improves error detection and the accuracy of the lowest-risk half of answers for Qwen3-VL and UI-Venus-1.5. Error detection also improves in four additional models (Table 12 in App. E). Across all six models, consistent answers are correct 57.5% to 78.1% of the time, compared with 10.9% to 31.4% for inconsistent answers. Combining the signals improves AUROC over token confidence alone by 0.039–0.147, with positive 95% CIs in every model. **High token confidence does not guarantee stability or correctness.** Among answers with token confidence at or above the median, inconsistent answers are correct in 13.9% to 50.3% of cases, compared with 62.4% to 84.8% for consistent answers. Single-probe and parser-free variants also improve error detection (App. E). **Shift consistency therefore complements token confidence for flagging likely errors and selecting more reliable answers.**

4.3 Mitigating Instability with Grid-Phase Canonicalization

Method. GridPC extends pad-to-grid with alignment correction, requiring one forward pass and no training. It chooses a scale $s = Q/P$, where Q is the effective token stride and $P \geq Q$ is the smallest

integer that permits a legal canvas with one extra stride along each axis. It preserves raw pixels when $P = Q$. It estimates horizontal and vertical alignment from image edges, ignoring repeated rows and columns to prevent padding from biasing the alignment estimate. Left and top padding compensate for the estimated phase, followed by padding to a legal token grid (App. F). Pad-to-grid uses the same scale without alignment correction.

Evaluation. We scan 96 screenshots, 32 per benchmark with one target each, using 0–64 raw pixels of right or bottom padding and 0–32 raw pixels of left or top padding to shift the content. Table 4 averages switching and correctness flips over all one-pixel steps, including native transitions and adjacent sizes.

Table 4: **GridPC improves grounding stability.** Switch, Flip, and Acc. are UI-element switching, correctness flips, and accuracy, reported as screenshot-weighted percentages. Stability uses 96 screenshots. Accuracy uses all 2,010 targets on 1,450 original screenshots.

		Padding		Content shift		
Model	Method	Switch ↓	Flip ↓	Switch ↓	Flip ↓	Acc. ↑
Qwen3-VL	Native	12.22	4.44	11.17	4.51	45.62
	Pad-to-grid	0.37	**0.09**	13.59	4.51	46.25
	GridPC	**0.33**	0.13	**0.59**	**0.33**	**46.59**
UI-Venus-1.5	Native	7.20	3.66	8.95	4.57	59.39
	Pad-to-grid	**0.13**	0.09	11.43	5.55	59.73
	GridPC	0.24	**0.08**	**0.59**	**0.23**	**60.95**

Results. Pad-to-grid reduces switching under padding but increases it under content shifts.GridPC reduces switching under both perturbations to **at most 0.59%** in both models, from 7.20–12.22% under native preprocessing, and also reduces correctness flips. Accuracy is maintained, with gains of 0.96 percentage points for Qwen3-VL and 1.56 for UI-Venus-1.5. App. F.1 reports evaluation on InternVL3.5. GridPC offers an initial mitigation guided by our findings, reducing instability under padding and content shifts at a fixed rendering scale. Addressing other sources of instability remains future work.

5 Related Work

Resolution, patch grids, and preprocessing. NaViT supports variable image sizes and aspect ratios (Dehghani et al., 2023), but vision models remain sensitive to small shifts (Shifman & Weiss, 2024) and resolution changes (Li et al., 2026). Mini-Monkey addresses semantic fragmentation from image partitioning (Huang et al., 2025). Shift-equivariant ViTs modify architectural components (Rojas-Gomez et al., 2024), whereas Phase Marginalization aggregates dense predictions over patch-grid phases (Ercan, 2026). We focus on content-preserving GUI padding and UI-element switches, and also relocate transition points. Prior work recovers preprocessing in black-box attacks (Sitawarin et al., 2022); Steal the Patch Size uses synthetic grids, padding, and consistency tests to infer tokenizer and resizing settings (Hu et al., 2026). Our probe instead infers token stride and rounding from clicks on natural GUI screenshots and validates them on held-out inputs.

GUI-grounding robustness and reliability. GUI-Perturbed evaluates visual and instruction perturbations (Wang et al., 2026c). MVP combines predictions from multiple views using spatial clustering (Zhang et al., 2026), while VISTA trains with target-preserving views (Qiu et al., 2026). GUI-RC aggregates sampled locations through region consistency (Du et al., 2026); SafeGround uses sampling-based spatial uncertainty (Wang et al., 2026a), and UI-Zoomer combines spatial agreement and token confidence for adaptive zooming (Tang et al., 2026). Our shift-consistency probe flags errors using content shifts within a fixed configuration, rather than aggregating sampled answers, and GridPC modifies preprocessing in one forward pass without changing the model.

6 Conclusion

This paper investigates why one pixel can change a click. Controlled interventions identify preprocessing transitions as a causal source of instability. Mechanism analysis reveals sensitivity to resizing, alignment, and context. These findings guide three investigations: inferring hidden settings from clicks, detecting errors through consistency under content shifts, and mitigating instability with GridPC. Consistency complements token confidence, while GridPC improves stability without training. Together, these findings link the mechanisms of unstable clicks to practical improvements in grounding reliability, providing insights into the design of more stable computer-use agents.

AI USE STATEMENT

Generative AI tools assisted literature discovery, coding, data analysis, and language editing. The authors independently reviewed and validated all AI-assisted outputs and take responsibility for the paper's claims, results, and presentation.

REPRODUCIBILITY STATEMENT

Sec. 2 and App. A specify the data, models, metrics, and inference settings. Sec. 3 and Apps. B–C describe concentration, configuration removal, transition-point relocation, and mechanism analyses, including controlled inputs, scale-change dose response, and activation patching. Sec. 4.1 and App. D specify output-based rule inference and held-out validation. Sec. 4.2 and App. E specify the shift-consistency probe. Sec. 4.3 and App. F describe GridPC and its evaluation under padding and content shifts. App. G reports correctness flips across preprocessing transitions.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025a.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9313–9332, 2024.

Mostafa Dehghani, Basil Mustafa, Josip Djolonga, Jonathan Heek, Matthias Minderer, Mathilde Caron, Andreas Steiner, Joan Puigcerver, Robert Geirhos, Ibrahim M Alabdulmohsin, et al. Patch n'pack: Navit, a vision transformer for any aspect ratio and resolution. *Advances in Neural Information Processing Systems*, 36:2252–2274, 2023.

Yong Du, Yuchen Yan, Fei Tang, Zhengxi Lu, Chang Zong, Weiming Lu, Shengpei Jiang, and Yongliang Shen. Test-time reinforcement learning for gui grounding via region consistency. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30593–30601, 2026.

Oğuzhan Ercan. Phase marginalization for patch-grid instability in vision transformers. *arXiv preprint arXiv:2606.08132*, 2026.

Kai Hu, Akash Bharadwaj, Weichen Yu, and Matt Fredrikson. Steal the patch size: Adversarially manipulate vision-language models. *arXiv preprint arXiv:2607.00174*, 2026.

Mingxin Huang, Yuliang Liu, Dingkang Liang, Lianwen Jin, and Xiang Bai. Mini-monkey: Alleviating the semantic sawtooth effect for lightweight mllms via complementary image pyramid. In *International Conference on Learning Representations*, volume 2025, pp. 85804–85823, 2025.

Chenxu Li, Zhicai Wang, Yuan Sheng, Xingyu Zhu, Yanbin Hao, and Xiang Wang. Res-bench: Benchmarking the robustness of multimodal large language models to dynamic resolution input. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 31545–31553, 2026.

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, 2025.

Y Lu, T Dhome-Casanova, J Yang, and A Awadallah. Omniparser v2: Turning any llm into a computer use agent. *Microsoft Research. Retrieved March*, 18:2025, 2025.

Shravan Nayak, Xiangru Jian, Kevin Qinghong Lin, Juan A Rodriguez, Montek Kalsi, Rabiul Awal, Nicolas Chapados, M Tamer Özsu, Aishwarya Agrawal, David Vazquez, et al. Ui-vision: A desktop-centric gui benchmark for visual perception and interaction. *arXiv preprint arXiv:2503.15661*, 2025.

OpenAI. GPT-5.6 Luna. OpenAI API model documentation, 2026a. URL `https://developers.openai.com/api/docs/models/gpt-5.6-luna`. Accessed September 22, 2026.

OpenAI. GPT-5.4 nano. OpenAI API model documentation, 2026b. URL `https://developers.openai.com/api/docs/models/gpt-5.4-nano`. Model snapshot `gpt-5.4-nano-2026-03-17`; accessed September 22, 2026.

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.

Xinyu Qiu, Yunzhu Zhang, Heng Jia, Shuheng Shen, Changhua Meng, and Linchao Zhu. Vista: View-consistent self-verified training for gui grounding. *arXiv preprint arXiv:2606.14579*, 2026.

Renan A Rojas-Gomez, Teck-Yian Lim, Minh N Do, and Raymond A Yeh. Making vision transformers truly shift-equivariant. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5568–5577, 2024.

Amir Rosenfeld, Neta Glazer, and Ethan Fetaya. Questioning the stability of visual question answering. *arXiv preprint arXiv:2511.11206*, 2025.

Ofir Shifman and Yair Weiss. Lost in translation: modern neural networks still struggle with small realistic image transformations. In *European Conference on Computer Vision*, pp. 231–247. Springer, 2024.

Chawin Sitawarin, Florian Tramèr, and Nicholas Carlini. Preprocessors matter! realistic decision-based attacks on machine learning systems. *arXiv preprint arXiv:2210.03297*, 2022.

Fei Tang, Bofan Chen, Zhengxi Lu, Tongbo Chen, Songqin Nong, Tao Jiang, Wenhao Xu, Weiming Lu, Jun Xiao, Yueting Zhuang, et al. Ui-zoomer: Uncertainty-driven adaptive zoom-in for gui grounding. *arXiv preprint arXiv:2604.14113*, 2026.

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

Qingni Wang, Yue Fan, and Xin Eric Wang. Safeground: Know when to trust gui grounding models via uncertainty calibration. *arXiv preprint arXiv:2602.02419*, 2026a.

Shihao Wang, Shilong Liu, Yuanguo Kuang, Xinyu Wei, Yangzhou Liu, Zhiqi Li, Yunze Man, Guo Chen, Andrew Tao, Guilin Liu, et al. Locateanything: Fast and high-quality vision-language grounding with parallel box decoding. In *European Conference on Computer Vision*, pp. 336–357. Springer, 2026b.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

Yangyue Wang, Harshvardhan Sikka, Yash Mathur, Tony Zhou, Jinu Nyachhyon, and Pranav Guruprasad. Gui-perturbed: Domain randomization reveals systematic brittleness in gui grounding models. *arXiv preprint arXiv:2604.14262*, 2026c.

Qianhui Wu, Kanzhi Cheng, Rui Yang, Chaoyun Zhang, Jianwei Yang, Huiqiang Jiang, Jian Mu, Baolin Peng, Bo Qiao, Reuben Tan, et al. Gui-actor: Coordinate-free visual grounding for gui agents. *Advances in Neural Information Processing Systems*, 38:15101–15128, 2026.

Tianbao Xie, Jiaqi Deng, Xiaochuan Li, Junlin Yang, Haoyuan Wu, Jixuan Chen, Wenjing Hu, Xinyuan Wang, Yuhui Xu, Zekun Wang, et al. Scaling computer-use grounding via user interface decomposition and synthesis. *Advances in Neural Information Processing Systems*, 38, 2026.

Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe Huang, Amrita Saha, Zeyuan Chen, et al. Gta1: Gui test-time scaling agent. In *International Conference on Learning Representations*, volume 2026, pp. 48295–48323, 2026.

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *International Conference on Learning Representations*, volume 2024, pp. 1651–1678, 2024.

Yunzhu Zhang, Zeyu Pan, Zhengwen Zeng, Shuheng Shen, Changhua Meng, and Linchao Zhu. Mvp: Multiple view prediction improves gui grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27482–27492, 2026.

A Evaluation Protocol

A.1 Data, models, and inference

The primary panel contains 2,010 target annotations on 1,450 screenshots. We sample 600 ScreenSpot-Pro annotations and 300 from each UI-Vision subset (Basic, Functional, and Spatial), and use all 510 OSWorld-G annotations with a target. Original instructions are retained. Each annotation receives one right- or bottom-padding axis, balanced within these five strata. There are 2,010 complete transition windows per model, except 2,005 for UI-Venus-1.5 and 2,009 for InternVL3.5.

Table 5 gives preprocessing settings and output interfaces. We use the Instruct checkpoints of Qwen3-VL-8B, Qwen2.5-VL-3B, and InternVL3.5-8B, and the Qwen2.5-VL variant of GUI-Actor-7B. Qwen3-VL uses a 3,136–8,847,360 pixel budget with a 16-pixel patch size, followed by no further resizing.

Inference uses BF16 and greedy decoding, or the top-ranked spatial prediction for GUI-Actor. Output limits are 32 tokens for Qwen3-VL and Qwen2.5-VL, 48 for GTA1, 64 for InternVL3.5, 256 for UI-Venus-1.5 and LocateAnything, and 500 for UI-TARS-1.5. LocateAnything uses its hybrid box decoder with temperature zero, top-p one, repetition penalty 1.1, and six future-token positions. Settings are fixed across paired conditions. Points and box centers are mapped to original screenshot coordinates using the model's output convention and the inverse image transform.

A.2 UI-element assignment and human validation

OmniParser V2 (Lu et al., 2025) detects regions on each original screenshot. A click is assigned to the smallest containing interactive container, or the smallest containing region if no interactive container exists. The icon, label, and background of one control share an element. Clicks outside all detected regions remain unassigned. For assigned elements e_a, e_b, with $\varnothing$ denoting an unassigned click or unparsed output, switching is $s(a, b) = \mathbf{1}[e_a \neq e_b,\ e_a \neq \varnothing,\ e_b \neq \varnothing]$. Every tested pair remains in the denominator. Assignment coverage is the fraction with two assigned elements. Unparsed outputs count as incorrect. For a matched window with predictions a^L, a^-, a^+, a^R, excess switching is

$$E_\phi = s(a^-, a^+) - \tfrac{1}{2}\{s(a^L, a^-) + s(a^+, a^R)\}. \tag{3}$$

One expert labels 900 pairs on 300 Qwen3-VL screenshots, 100 per benchmark and balanced by axis. Each screenshot supplies one transition pair and two controls. Pair order and within-pair prediction order are randomized. The annotator sees the instruction and predictions but not padding amounts, grid dimensions, or pair type. Labels distinguish the same element, different elements, background predictions, and ambiguity. All pairs are resolved. The strict endpoint counts two different UI elements. Automatic labels agree with 94.67% of judgments, with switch precision 83.19%, recall 77.95%, and F1 80.49%. Table 6 shows similar excess switching under both assignments. Of 65 human-labeled transition switches, 64 select elements with different functions. This audit does not estimate inter-annotator variation.

A.3 Aggregation and uncertainty

Unless stated otherwise, we average targets and tested pairs within each screenshot, then weight screenshots equally. Across-model means weight models equally. Paired 95% confidence intervals

Table 5: **Preprocessing and output interfaces.** Effective token stride is patch size times spatial merge factor, in processed-image pixels. InternVL3.5 uses tiles. Box outputs are scored at their centers.

Model	Preprocessing family	Token stride or tile size	Image-size processing	Output
Qwen3-VL-8B	Qwen3-VL	$16\times2 = 32$	Round each dimension to the nearest multiple. Use 3,136–8,847,360 processed pixels.	normalized point
UI-Venus-1.5-8B	Qwen3-VL	$16\times2 = 32$	Round each dimension to the nearest multiple. Use 65,536–16,777,216 processed pixels.	normalized point
Qwen2.5-VL-3B	Qwen2.5-VL	$14\times2 = 28$	Round each dimension to the nearest multiple. Use 3,136–12,845,056 processed pixels.	absolute box
GTA1-7B	Qwen2.5-VL	$14\times2 = 28$	Round each dimension to the nearest multiple. Use 3,136–12,845,056 processed pixels.	absolute point
UI-TARS-1.5-7B	Qwen2.5-VL	$14\times2 = 28$	Round each dimension to the nearest multiple. Use 3,136–12,845,056 processed pixels.	absolute point
GUI-Actor-7B	Qwen2.5-VL	$14\times2 = 28$	Pre-resize to about 5.76M pixels. Then round each dimension to the nearest multiple under a 5,720,064-pixel limit.	action attention map
LocateAnything-3B	LocateAnything	$14\times2 = 28$	Scale using a 25,600 pre-merge patch budget. Then round each dimension upward.	box or point
InternVL3.5-8B	InternVL3.5	448-pixel tiles	Select up to 12 tiles from the aspect ratio. Add a thumbnail when using multiple tiles.	normalized point

Table 6: **Blinded human validation of UI-element switching for Qwen3-VL-8B.** One expert labels 900 prediction pairs from 300 screenshots. Each screenshot provides one transition pair and two adjacent controls. Excess is the transition rate minus the mean control rate. Values are percentages or percentage points.

Assignment	Control −	Transition	Control +	Excess [95% CI]
Human UI element	11.00	21.67	9.67	11.33 [7.50,15.33]
Automatic UI element	10.00	20.67	9.00	11.17 [7.33,15.17]

use the 2.5th and 97.5th percentiles of 10,000 screenshot-cluster bootstrap replicates. Across-model contrasts jointly resample screenshot identities to preserve dependence. Paired differences are computed before rounding the reported rates.

Relative click distance is Euclidean click displacement divided by the original screenshot diagonal, multiplied by 100. Primary-panel distance comparisons require parsed clicks at all four window inputs, leaving 2,003, 2,005, 1,868, 2,009, 2,010, 2,010, 1,924, and 1,514 windows for the models in Table 5, in order. The two control distances are averaged within each window before aggregation.

B Preprocessing Transitions and Causal Interventions

B.1 Transition rules

With pixel-budget scaling inactive, Qwen3-VL rounds a raw dimension d to $r_Q(d) = Q\,\mathrm{round}_{\mathrm{even}}(d/Q)$, where $Q = 32$. At $(k+\frac{1}{2})Q$, ties-to-even selects kQ for even k and $(k+1)Q$ for odd k. Thus the transition pair is $(b_k - 1, b_k)$, where

$$b_k = kQ + Q/2 + \mathbf{1}\{k \text{ is even}\}, \qquad r_Q(b_k - 1) = kQ, \quad r_Q(b_k) = (k+1)Q. \tag{4}$$

A width transition adds one merged token column. With grid height H_G, it adds H_G visual tokens. More generally, the change in visual-token count is $\Delta N = H_G^+ W_G^+ - H_G^- W_G^-$.

Other families. Qwen2.5-VL uses the same rule with $Q = 28$ (Bai et al., 2025b). For both Qwen families, first round each dimension to the nearest multiple of Q, with a minimum of Q. If the rounded area exceeds the upper pixel budget $A_{\max}$, scale the input dimensions by $\sqrt{A_{\max}/(HW)}$ and round each down to a multiple of Q. If it is below $A_{\min}$, scale by $\sqrt{A_{\min}/(HW)}$ and round up. Thus, when a budget is active, a change in width can also change the processed height. GUI-Actor first scales the image to approximately 3200×1800 pixels in area while preserving aspect ratio, truncating both dimensions to integers. It then applies the 28-pixel rounding rule with upper budgets 12,845,056 and 5,720,064 in succession. The latter limit is specified in the checkpoint configuration.

LocateAnything uses a pre-merge patch budget of 25,600 from its checkpoint configuration. If $\lfloor W/14 \rfloor \lfloor H/14 \rfloor > 25{,}600$, it scales both dimensions using the square root of the budget divided by this count, then rounds dimensions upward to multiples of 28. The final patch count can slightly exceed that scaling budget. InternVL3.5 considers layouts with c columns and r rows, where $c, r \geq 1$ and $cr \leq 12$, and minimizes $|W/H - c/r|$. Candidates are considered in increasing tile count. At equal aspect-ratio error, a candidate replaces the current choice when $HW > \frac{1}{2}\, 448^2 cr$. The screenshot is resized to $(448c, 448r)$ and split into tiles, with a 448×448 thumbnail added for multiple tiles. For every model, transition detection evaluates the complete size mapping, including budget scaling and any pre-resize. A transition is a change in its returned grid or tile layout across consecutive raw dimensions.

B.2 Concentration of switching

Table 7 reports the primary-panel switching rates.

Table 7: **UI-element switching increases at preprocessing transition points.** Control averages the two adjacent controls. Transition is the switch rate across the predicted transition, and Excess is their paired difference. Rates are screenshot-weighted percentages, and excess is in percentage points. Brackets give 95% confidence intervals from 10,000 screenshot-cluster bootstrap replicates. The All models row weights models equally.

Model	Control	Transition	Excess [95% CI]
Qwen3-VL	11.37	20.64	9.27 [7.68, 10.88]
UI-Venus-1.5	8.91	15.85	6.95 [5.43, 8.47]
Qwen2.5-VL	16.76	27.98	11.22 [9.40, 13.11]
GTA1	11.91	22.72	10.81 [9.06, 12.54]
UI-TARS-1.5	14.27	25.61	11.33 [9.55, 13.09]
GUI-Actor	9.42	19.55	10.14 [8.43, 11.86]
LocateAnything	6.39	13.42	7.03 [5.63, 8.47]
InternVL3.5	13.39	30.41	17.02 [14.99, 19.07]
All models	**11.55**	**22.02**	**10.47 [9.81, 11.14]**

All-position baseline. For the gray bars in Fig. 3, Qwen3-VL, GTA1, LocateAnything, and InternVL3.5 use 300 shared screenshots, 100 per benchmark, with one target and a balanced padding axis per screenshot. Let b be the first padding amount in $[6, 2048]$ satisfying the four-input transition pattern. Each greedy scan covers 65 consecutive amounts starting at $\max(0, b - 32)$. The baseline averages all 64 one-pixel steps, including transitions, within screenshots and then across screenshots. Distance estimates require four parsed central inputs and use adjacent pairs with two parsed clicks. These scans are separate from the full primary panel used for transition and adjacent-control bars. Each panel weights its displayed models equally.

B.3 Holding the configuration fixed

We process each input in (I^-, I^+) under both neighboring configurations G^- and G^+. Grid-based models resize to the requested dimensions and bypass further resizing. InternVL3.5 uses the requested tile layout. The different-configuration comparison pairs $I^- \to G^-$ with $I^+ \to G^+$. The shared-configuration comparison averages the two within-configuration switch rates. All four predictions utilize consistent controlled rendering and coordinate mapping, with grid or layout validated through preprocessing metadata.

Fig. 8 illustrates the intervention and its results.

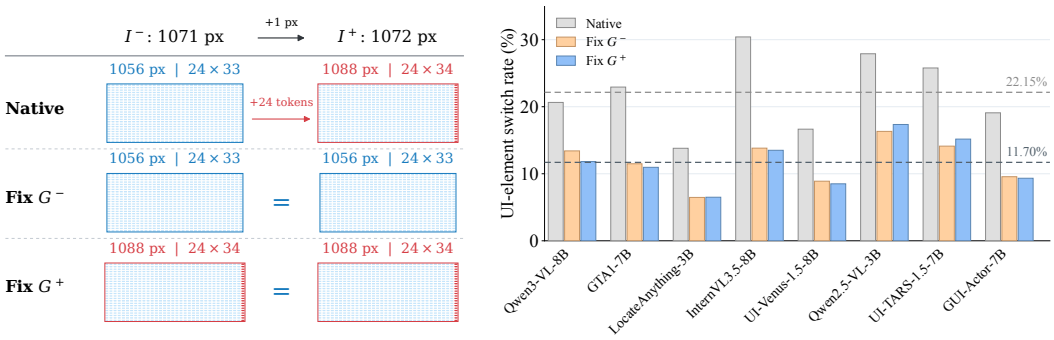


Figure 8: **Fixing preprocessing configurations reduces switching.** Left: a width-only example. Native assigns each input its own configuration; Fix G^- and Fix G^+ assign one configuration to both inputs. Equal signs indicate equal grid dimensions. Right: switching rates across eight models. All three conditions use controlled rendering; Native corresponds to Different in Table 8, rather than Standard. Dashed lines mark the Native mean (22.15%) and the mean of both fixed conditions (11.70%).

Explicit resizing can change image pixels relative to standard preprocessing even at matching grid dimensions. Table 8 therefore separates the standard transition rate from the controlled baseline. Both choices of shared configuration reduce switching in every model. Their mean is 11.70%, a reduction of 10.45 percentage points $[9.79, 11.13]$ from the controlled baseline. Its paired difference from standard adjacent controls is 0.15 points $[-0.25, 0.56]$.

Input-tensor check. We compare standard and controlled preprocessing on both inputs of the first window in each of the five benchmark strata, giving ten inputs per model. Image tensors differ on ten UI-Venus-1.5 inputs, eight GUI-Actor inputs, and two LocateAnything inputs, despite matching grid metadata, token IDs, and attention masks. This verifies that matching grid dimensions need not produce identical pixels and motivates using the same controlled procedure for every intervention condition.

Table 8: **Configuration assignment on the same transition pairs.** Rates are screenshot-weighted percentages. Standard uses the released preprocessing path. The remaining columns use controlled rendering. Different assigns G^- and G^+ to their respective inputs. The last two columns assign one configuration to both inputs.

Model	Standard	Different	Both G^-	Both G^+
Qwen3-VL	20.64	20.64	13.41	11.82
UI-Venus-1.5	15.85	16.65	8.89	8.50
Qwen2.5-VL	27.98	27.90	16.32	17.35
GTA1	22.72	22.93	11.49	10.96
UI-TARS-1.5	25.61	25.78	14.12	15.17
GUI-Actor	19.55	19.09	9.56	9.33
LocateAnything	13.42	13.80	6.47	6.50
InternVL3.5	30.41	30.41	13.82	13.50

B.4 Moving transition points

The modified rule selects $G_\phi(H, W-16)$ for width padding and $G_\phi(H-16, W)$ for height padding, then renders the actual input in that configuration. The offset is fixed before inference. Both rules retain the same rendering operation, including the InternVL3.5 thumbnail and the LocateAnything intermediate budget resize at the actual input size.

For a native transition at padding amount b, we evaluate $b-2, b-1, b, b+1$ and the same four amounts plus 16 under both rules. At the native location, the native rule must select $[G_0, G_0, G_1, G_1]$ and the modified rule $[G_0, G_0, G_0, G_0]$. At the shifted location, they must select $[G_1, G_1, G_1, G_1]$ and $[G_0, G_0, G_1, G_1]$, respectively. Geometry alone selects the first transition at or after the primary one satisfying these conditions. This moves the window for 0 Qwen3-VL, 18 Qwen2.5-VL, 210 LocateAnything, and 1 InternVL3.5 annotations. All primary annotations remain.

All 16 predictions per annotation use one GPU, batch size one, and counterbalanced condition order. Excess switching is calculated with Eq. 3 at each location. Native minus modified excess gives the native-location decrease, and modified minus native gives the shifted-location increase. Paired bootstrap inference follows App. A.3. Both effects have positive 95% intervals for every model and within each benchmark.

Released-rule inputs reproduce native image tensors exactly. When the rules select identical configurations, their tensors also match. Switching on these identical inputs is 0.10% for Qwen3-VL, 0.37% for Qwen2.5-VL, and zero for the other two models. Restricting to annotations with all 16 clicks assigned to UI elements retains 1,465, 917, 1,569, and 757 annotations, respectively. Both effects remain positive with 95% intervals above zero in each model.

B.5 UI-ELEMENT SELECTION PROBABILITIES

On the 300 shared screenshots used for the all-position scans, Qwen3-VL, GTA1, LocateAnything, and InternVL3.5 each receive four neighboring inputs and generate 20 responses per input at temperature one and top-p one, giving 96,000 sampled responses. The window construction and preprocessing match the scans; decoding otherwise follows App. A.1. Response categories are assigned UI elements, unassigned clicks, and unparsed outputs. For category counts n_{jc} at input j and $N = 20$, an unbiased estimate of squared L_2 distance between category probabilities is

$$\widehat{d}(j,k) = \sum_c \left[\frac{n_{jc}(n_{jc}-1) + n_{kc}(n_{kc}-1)}{N(N-1)} - \frac{2n_{jc}n_{kc}}{N^2} \right]. \tag{5}$$

Each paired increase subtracts the mean of the two adjacent-control distances. All screenshots contribute, including those without a greedy switch. Averaged equally across models, distance rises from 0.0364 on adjacent controls to 0.1267 on transition pairs, a paired increase of 0.0902 (95% CI $[0.0718, 0.1095]$). These are squared probability distances, not percentages. Intervals use 10,000 screenshot-cluster bootstrap replicates, jointly resampling shared screenshots for the model average. Each model has a positive increase (Table 9), including under Bonferroni-adjusted 98.75% intervals. The assigned-element contribution is positive in every model with 95% intervals above zero. For InternVL3.5, it is 0.0266; unassigned and unparsed categories contribute another 0.0299, so assignment coverage also matters.

Table 9: **Transitions change output-category probabilities.** Paired squared L_2 increase over adjacent controls, with 95% confidence intervals; each model uses all 300 screenshots.

Model	Increase [95% CI]
Qwen3-VL	0.0601 [0.0340, 0.0889]
GTA1	0.1814 [0.1288, 0.2371]
LocateAnything	0.0629 [0.0336, 0.0950]
InternVL3.5	0.0565 [0.0431, 0.0701]

C MECHANISM ANALYSIS

C.1 CONTROLLED INPUTS

Every controlled image is rendered from one raw screenshot using bicubic interpolation. Along an axis, raw pixel u maps to $x = (u + \frac{1}{2})s - \frac{1}{2} + o$, where s is scale and o is content offset. Outside-content regions repeat the nearest edge pixel. Integer shifts copy pixels exactly. Clicks are mapped back by inverting this transform, and all inputs bypass further model-side resizing.

For the grid, resizing, and activation-patching comparisons, every input uses the raw transition-left screenshot I^-. Qwen3-VL contributes 1,992 windows where exactly one grid dimension grows by one stride. The other 18 windows change both dimensions under budget scaling. UI-Venus-1.5 contributes 2,005 windows. As defined in Sec. 3.2, X^- and X^+ resize this same screenshot to the smaller and larger grids, and X^{pad} extends X^- with edge pixels to the larger grid. Thus (X^-, X^{pad}) preserves existing pixels while enlarging the grid, and (X^{pad}, X^+) holds the grid fixed while changing resizing.

The content-shift scans use all 2,010 annotations along both axes. The canvas is the native processed image plus one stride along the scanned axis. Offsets are $0.5, 1, 2, 4, 8, 12, 16, 20, 24, 28, 32$ processed pixels, each compared with the unshifted input. The half-pixel shift uses interpolation. All

comparisons use the same prompt, decoding settings, and screenshot weighting. Intervals use 10,000 benchmark-stratified screenshot resamples.

Pixel perturbations. On the horizontal scan canvas, JPEG uses quality 75 or 95. Independent Gaussian noise has standard deviation 2 or 4 on the 0–255 color scale, with rounding and clipping to that range. The interpolation control uses bilinear rendering at the reference geometry. Relative visual-feature change is $\|V' - V\|_2 / \|V\|_2$, where V and V' contain the reference and perturbed visual tokens. Tokens are compared at corresponding positions, including for the half-pixel and one-pixel shifts.

C.2 Magnification and target-position controls

These controls test whether changes in target position account for the resizing effect. The canvas extends the larger grid by one stride along the changed axis. The reference X^{ref} places the smaller-scale rendering at an offset of one stride. Magnified inputs use the larger-grid scale, anchored either at the left content edge (X_{corner}) or at the target center (X_{target}). A shift-only input uses the reference scale and the target displacement induced by corner-anchored magnification. Each is compared with the reference.

Target-anchored magnification switches 17.29% of Qwen3-VL clicks, compared with 21.29% for corner-anchored magnification and 20.84% for the shift-only input. UI-Venus-1.5 gives 13.85%, 16.19%, and 16.22%. Substantial switching therefore persists with the target center fixed.

C.3 Scale-change dose response

This analysis reuses the native predictions from the GridPC stability panel (App. F). The panel has 96 screenshots, each padded with 0–64 repeated-edge pixels on the right or bottom. Padding by t pixels along an axis gives content scale $s(t) = P(t)/(L + t)$ along that axis, where L is the raw length and $P(t)$ is the processed length. For each axis, we form all 2,080 pairs of padding amounts $a < b$ and compute $|\Delta s| = |s(b)/s(a) - 1|$. Pairs are grouped by whether the processed size changes. Bin edges are 0.05, 0.1, 0.2, 0.5, 1, 2, and 4%. Bin rates weight screenshots equally. Fig. 4(c) plots each bin at its mean $|\Delta s|$ and omits bins with fewer than 50 screenshots.

For the matched comparison, each one-pixel transition pair receives the same-grid rate of its bin. Observed and predicted rates are averaged within screenshots, then across screenshots. Intervals use 10,000 benchmark-stratified screenshot resamples, and each resample recomputes same-grid rates.

Within a configuration, one-pixel size changes alter content scale by about 0.07%; at transition points, the average change is 2.1% because the processed size jumps by one stride. For Qwen3-VL, one-pixel transition pairs switch 17.97% of clicks, compared with 18.08% predicted from same-grid pairs matched by scale-change bin. The observed-minus-predicted difference is -0.11 percentage points (95% CI $[-3.58, 3.54]$). For UI-Venus-1.5, the corresponding rates are 14.84% and 13.00%, a difference of $+1.85$ points (95% CI $[-1.87, 5.69]$). In the same scans, transition pairs exceed same-grid one-pixel pairs by 5.93 and 7.89 percentage points, respectively. Both matched-comparison intervals include zero; this does not establish equivalence between transition and same-grid rates.

Same-grid switching rises from 12.60% at $|\Delta s| \approx 0.07\%$ to 19.99% at 0.7% for Qwen3-VL, and from 7.15% to 11.55% for UI-Venus-1.5 (Fig. 4(c)). At matched $|\Delta s|$ in the bins of Fig. 4(c), pairs with different grids switch at most 4.0 points more often than same-grid pairs for Qwen3-VL and at most 1.1 points for UI-Venus-1.5. The Qwen3-VL gap is largest at the smallest scale changes. This small grid effect agrees with the grid-expansion comparison in Sec. 3.2. Transition rates in this panel differ from those in Sec. 3.1 because the screenshots differ.

C.4 Activation patching

We replace features of X^{ref} with those of X_{target} at selected token positions and compare the resulting click with the unpatched reference. Both models supply merged visual tokens to the language model and inject three intermediate vision-layer features into early language-model layers. Each replacement includes the merged token and all three injected features. Replacing all positions exactly reproduces the prediction on X_{target}, verifying the replacement implementation. Relative to X^{ref}, these predictions switch 17.13% of Qwen3-VL clicks and 13.98% of UI-Venus-1.5 clicks.

The target region contains tokens overlapping the target box in either input, expanded by one token in every direction. Context is its complement. A radius-k region contains positions within Chebyshev distance k of the target region, measured by the larger of the horizontal and vertical token offsets. Far context is more than 16 tokens away. Random context masks select 25%, 50%, or 75% of non-target positions using a fixed per-window seed. Clicks use the reference coordinate map. The input geometries coincide at the target center and differ by at most one stride elsewhere.

C.5 TILE-BASED PREPROCESSING IN INTERNVL3.5

InternVL3.5 adds a global thumbnail to its tile layout. Across 2,009 transition windows, we construct paired inputs from the same raw transition-left screenshot. Changing the layout while preserving content tiles still changes the thumbnail and switches 29.45% of clicks, close to 30.45% for the full comparison (Table 10). Thus this construction does not cleanly separate layout changes from resizing. At a fixed layout, shifts of one token stride (28 pixels) or one tile (448 pixels) do not show the switching reduction seen in the two token-grid models; the thumbnail also changes. These results limit generalization of the alignment mechanism. Accuracy is about 20%, and 18% of outputs lack a parsed click in this setting.

Table 10: **Controlled inputs for InternVL3.5.** Switch rates (%) with 95% screenshot-cluster bootstrap intervals. Fixed-layout inputs use the larger layout and its unshifted reference; shifts follow the axis that gains tiles.

Comparison	Switch [95% CI]
Full transition comparison	30.45 [28.30, 32.67]
Layout change, unchanged content tiles	29.45 [27.35, 31.65]
Resizing at a fixed layout	33.16 [30.99, 35.39]
Content shift by 14 pixels	27.65 [25.59, 29.80]
Content shift by 28 pixels	28.03 [25.95, 30.15]
Content shift by 448 pixels	27.16 [25.05, 29.40]

D INFERRING PREPROCESSING RULES FROM CLICKS

Sampling and inference. The 300 screenshots in Sec. 4.1 are selected by metadata and a fixed hash ordering, independently of outputs and transition points. Each benchmark contributes 50 discovery and 50 held-out screenshots, with balanced padding axes. UI-Vision contains 34 Basic, 34 Functional, and 32 Spatial screenshots. Eligible images satisfy $(W + 64)(H + 64) < 7{,}000{,}000$. Each model receives 19,500 primary queries at padding amounts 0–64 and 3,600 additional queries repeating amounts 0, 17, and 43 four times. The split and schedule are fixed before inference.

Open-weight models use BF16 and greedy decoding. Qwen3-VL, Qwen2.5-VL-7B-Instruct, and UI-Venus-1.5 use batch size one, while GTA1 uses batch size 16. Qwen2.5-VL-7B predicts absolute boxes with a 128-token limit and the Qwen2.5-VL preprocessing in Table 5. The other models follow App. A.1. GPT-5.4 nano and GPT-5.6 Luna receive PNG images at original detail, temperature zero, reasoning disabled, and a 40-token completion limit. Their identifiers are `gpt-5.4-nano-2026-03-17` and `gpt-5.6-luna`. Both receive the benchmark instruction after this prompt:

> You are an expert GUI grounding model. Given a screenshot and an instruction, point to the UI element that should be clicked. Return only one point as [x, y], where x and y are normalized integers from 0 to 1000 relative to the full image. For an element with area, return the center point.

Coordinates are mapped to padded-image pixels. A comparison requires two parsed clicks. Closed-weight outputs outside the requested $[0, 1000]^2$ range are invalid.

Rule scoring and uncertainty. For each of the 228 candidate rules in Sec. 4.1, let b_i indicate a predicted transition and z_i indicate movement above 1% of the original diagonal. Within group $b \in \{0, 1\}$, let n_b count valid comparisons and $s_b = \sum_{i:b_i=b} z_i$. We use $\hat{p}_b = (s_b + 1/2)/(n_b + 1)$

and select the highest score with $\hat{p}_1 > \hat{p}_0$:

$$\ell(Q, r) = \sum_{b=0}^{1} [s_b \log \hat{p}_b + (n_b - s_b) \log(1 - \hat{p}_b)]. \tag{6}$$

Here r is floor, ceiling, nearest with half-up ties, or nearest with ties-to-even. Rule stability repeats the search on 10,000 discovery bootstrap samples, resampling screenshots within benchmarks. Held-out intervals use the same resampling scheme, keep the selected rule fixed, and recompute pooled event-rate differences.

A separate null test circularly shifts each screenshot's valid event sequence by a random nonzero offset, leaving singleton sequences fixed. Each of 1,000 replicates repeats the entire search and measures its gain over a single-rate Bernoulli model. The p-value is $(1 + k)/1001$, where k counts null gains at least as large as observed. It equals $1/1001$ for all six models. This tests alignment of events with candidate rules. Held-out prediction and parameter stability remain separate criteria. The single-rate model gives every valid comparison the same event probability. Circular shifts maintain the movement sequence in each screenshot while adjusting alignment with predicted transitions.

Coverage and repeatability. Table 11 reports held-out denominators and identical-input variability. Qwen2.5-VL-7B has 491 unparsed primary outputs, leaving 9,271 discovery and 9,384 held-out comparisons. Two held-out screenshots have no valid comparisons. GPT-5.4 nano has three such held-out screenshots. For all four open-weight models, the inferred rule matches actual grid changes on all 9,600 held-out size comparisons, including those without parsed clicks. Qwen3-VL and UI-Venus-1.5 each have 300 transitions, and GTA1 and Qwen2.5-VL-7B each have 348. The repeat rate measures variability while keeping the image constant. Specifically, closed-weight models can adjust their clicks without any change in input. The differences in held-out rates evaluate the predictive value of the inferred rule, whereas the repeat rates illustrate this additional variability.

Table 11: **Held-out movements and identical-input variability.** Transition and Other give large-movement events / valid one-pixel comparisons. Repeat is the event percentage among valid pairs of five responses to an identical input, pooled over 900 inputs. The threshold is 1% of the original diagonal throughout. Repeat pairs share responses and are descriptive.

Model	Transition	Other	Repeat (%)
Qwen3-VL	72 / 300	1,268 / 9,283	0.27
GTA1	82 / 348	1,185 / 9,252	1.46
Qwen2.5-VL-7B	175 / 343	3,000 / 9,041	0.76
UI-Venus-1.5	55 / 300	784 / 9,300	0.00
GPT-5.4 nano	197 / 292	4,697 / 9,101	38.79
GPT-5.6 Luna	94 / 300	2,481 / 9,294	20.01

Comparison with API documentation. The public API documentation lists both closed-weight models as patch-based, with $\lceil W/32 \rceil \times \lceil H/32 \rceil$ patches at original detail. For GPT-5.4 nano, the documentation also specifies at most 10,000 patches and a maximum side of 6,000 pixels. Our probe inputs require at most 6,741 patches and have a maximum side of 3,904 pixels, so these limits are inactive. GPT-5.6 preserves image dimensions below 65,535 pixels per side. Under these conditions, the documented rule is a 32-pixel ceiling rule. The rule inferred for GPT-5.4 nano matches it. For GPT-5.6 Luna, the inferred stride matches, but floor rounding predicts each transition one pixel before the documented ceiling transition. The 32-pixel ceiling rule is the second most frequent Luna selection across bootstrap resamples (27.68%), after the selected floor rule (40.97%). The documentation describes image-token accounting, so this comparison supports agreement with the documented patch rule rather than verification of internal processing.

E SHIFT-CONSISTENCY PROBE

Inputs and scores. Qwen3-VL and UI-Venus-1.5 use the horizontal shift canvas in App. C.1, with 32 extra processed pixels on the right. The unshifted prediction is the answer. Probes shift content right by 8, 16, and 24 pixels, repeating the left edge and shortening right padding. Qwen2.5-VL-3B, GTA1, and UI-TARS-1.5 use 28 extra pixels on the right and offsets of 7, 14, and 21 pixels. These five models preserve the full resized image across probes. InternVL3.5 uses a tiled extension with the same 7-, 14-, and 21-pixel offsets within its native tile layout. It crops pixels leaving the right edge and generates the thumbnail from each canvas, including the unshifted canvas used for the answer. Prompts, layouts, and greedy decoding stay fixed across probes. In this analysis, UI-TARS-1.5

uses a 128-token output limit and frequency penalty 1.0. GTA1 uses its reference one-dimensional language-model positions. All clicks are mapped to raw coordinates.

The inconsistency score counts probes assigned to a different element or no element. An unassigned answer makes every probe inconsistent. Token risk is the mean negative log probability of all generated answer tokens, with missing scores assigned high risk. Combined risk averages the two risk ranks. For N answers, let r_{token} and r_{shift} be ascending ranks, using average ranks for ties. The combined score is $(r_{\text{token}} + r_{\text{shift}})/(2N)$. Higher scores indicate greater estimated risk. Ranking places the two signals on a common scale without fitting weights or requiring target labels. An inconsistency score of zero defines a consistent answer. A single-probe variant uses only the half-stride shift. The parser-free variant counts a probe when its click is more than 1% of the original diagonal from the answer, or either click is unparsed.

Evaluation. Each of the 2,010 annotations contributes one answer. AUROC treats incorrect answers as positives and averages tied ranks. Accuracy and AUROC pool annotations in this analysis. Uncertainty uses 2,000 benchmark-stratified screenshot-cluster bootstrap replicates, retaining all targets per sampled screenshot and sharing draws for paired differences. The three-probe and single-probe variants require three and one extra forward passes, respectively. For Fig. 7, class densities are normalized separately and smoothed with a Gaussian kernel of bandwidth 0.05, reflected at zero and one. Metrics use unsmoothed scores.

Table 12: **Consistency under content shifts in six models.** Consistent reports the share of consistent answers; cons. / incons. gives accuracy in the two groups.

			Acc. (%)	AUROC		
Model	Acc. (%)	Consistent (%)	cons. / incons.	Token	Combined	Gain [95% CI]
Qwen3-VL	46.3	48.3	74.0 / 20.4	0.822	0.862	0.040 [0.027, 0.053]
UI-Venus-1.5	60.3	61.9	78.1 / 31.4	0.767	0.806	0.039 [0.026, 0.052]
Qwen2.5-VL-3B	20.8	20.8	57.5 / 11.2	0.755	0.850	0.095 [0.078, 0.112]
GTA1	50.6	51.0	74.5 / 25.7	0.623	0.755	0.132 [0.116, 0.148]
UI-TARS-1.5	41.7	40.9	71.8 / 21.0	0.748	0.821	0.073 [0.059, 0.087]
InternVL3.5	22.2	20.9	65.0 / 10.9	0.683	0.830	0.147 [0.127, 0.168]

Table 12 shows that consistency separates answers with different error rates in all six models. Table 13 retains the cheaper and parser-free controls. Among answers with token confidence at or above the median, consistent versus inconsistent accuracy is 82.1% versus 42.2% for Qwen3-VL and 84.8% versus 50.3% for UI-Venus-1.5. Across six models, the respective ranges are 62.4–84.8% and 13.9–50.3%.

InternVL3.5 target-cropping check. Combining shift consistency with token confidence improves error detection for InternVL3.5, matching the qualitative trend in the token-grid models. At least one probe clips the target box for 73 of 2,010 annotations. Removing these annotations leaves 1,937 targets. Recomputing risk ranks on this subset gives token-only and combined AUROC of 0.685 and 0.835, with a paired gain of 0.150 [0.128, 0.170]. This result measures consistency under the combined perturbation, which can also alter surrounding context and the thumbnail.

Table 13: **Probe variants for error detection.** All variants combine the probe with token confidence. Single uses one half-stride shift. Distance uses three parser-free probes. Accuracy at 50% coverage compares token confidence with the three-probe combination. Brackets give 95% bootstrap intervals.

Model	Single AUROC	Distance AUROC	Acc.@50% token / combined
Qwen3-VL	0.844 [0.825, 0.862]	0.844 [0.825, 0.861]	70.65 / 75.62
UI-Venus-1.5	0.784 [0.762, 0.806]	0.807 [0.785, 0.829]	78.31 / 82.29
Qwen2.5-VL-3B	0.824 [0.801, 0.846]	0.828 [0.806, 0.850]	33.33 / 36.72
GTA1	0.723 [0.700, 0.744]	0.750 [0.726, 0.772]	58.91 / 69.95
UI-TARS-1.5	0.799 [0.777, 0.818]	0.810 [0.787, 0.829]	59.60 / 66.27
InternVL3.5	0.812 [0.787, 0.836]	0.807 [0.783, 0.830]	32.64 / 37.41

F GRID-PHASE CANONICALIZATION

Scale and canvas. Let Q be the effective token stride. GridPC chooses the smallest integer $P \geq Q$ for which scale $s = Q/P$ permits a legal canvas containing $(sW + Q, sH + Q)$. For the two grid-based models,

$$C_W = Q\left\lceil (sW + Q)/Q \right\rceil, \qquad C_H = Q\left\lceil (sH + Q)/Q \right\rceil, \qquad C_W C_H \leq A_{\max}. \tag{7}$$

Here $Q = 32$, with $A_{\max} = 8{,}847{,}360$ for Qwen3-VL and $16{,}777{,}216$ for UI-Venus-1.5, as in App. A.1. The extra stride accommodates any phase offset. Canvas size depends on raw dimensions and scale, independently of the phase estimate. One period of P raw pixels becomes exactly one stride of Q processed pixels. This links the alignment estimate to the token grid even when the screenshot must be downscaled.

Phase and rendering. In the raw image, $e_x(i)$ sums absolute color differences between columns $i-1$ and i over rows that differ from their predecessor. The first row is retained. Define e_y analogously, retaining the first column. With $z_x = \sum_i e_x(i)\exp(2\pi \mathrm{i} i/P)$, the horizontal phase is $\phi_x = (P \arg z_x/2\pi) \bmod P$. If $|z_x| < 10^{-9}\max(1, \sum_i e_x(i))$, set $\phi_x = 0$. Vertical phase is computed identically. This circular mean measures where image edges fall within one period. Integer padding approximately cancels this phase before resizing. GridPC adds $o_x = (-\operatorname{round}(\phi_x)) \bmod P$ repeated-edge pixels on the left and o_y on top. It renders this padded image onto the selected canvas at scale s, using bicubic interpolation when $s < 1$ and exact pixels when $s = 1$, with edge replication outside the image. The pixel-center map is $x = (u + o_x + \frac{1}{2})s - \frac{1}{2}$, inverted to score clicks. Pad-to-grid uses the same scale, zero offsets, and the smallest legal canvas.

Ignoring duplicate rows and columns keeps repeated-edge padding from changing the phase estimate. At a fixed period, a content shift moves a nondegenerate phase estimate by the same amount modulo P. Compensation preserves alignment until a full-period wrap, scale change, or canvas change. On sampled screenshots, 94–96% of one-pixel shift steps yield identical processed images. Every tested right/bottom padding step preserving scale and canvas yields identical processed images.

Evaluation. The stability panel contains 96 primary-panel screenshots, 32 per benchmark, selected independently of predictions with one target each. Both axes are scanned at every integer amount, giving 128 adjacent pairs per screenshot for right/bottom padding (0–64) and 64 for left/top content shifts (0–32). Switching and correctness flips average all steps within screenshots, then weight screenshots equally. Identical processed inputs reuse one greedy prediction. Accuracy in Table 4 is evaluated separately on all 2,010 targets in the 1,450 original, unpadded screenshots. Paired intervals use 10,000 benchmark-stratified screenshot-cluster resamples.

F.1 INTERNVL3.5

For InternVL3.5, $Q = 28$ and the canvas rounds each dimension up to a multiple of 448, subject to a seven- or twelve-tile limit. The thumbnail is retained. Table 14 uses the same stability and accuracy panels. With seven tiles, GridPC uses 18% fewer visual tokens than native preprocessing. Its accuracy change is -0.55 percentage points $[-2.41, 1.28]$. With twelve tiles, the change is $+1.05$ points $[-0.86, 2.95]$ at 1.46 times the token count. These intervals support stability improvements without establishing an accuracy gain.

Table 14: **GridPC for InternVL3.5.** Columns follow Table 4; Tok. is the visual-token count relative to native preprocessing. The number in parentheses is the maximum number of tiles. The twelve-tile row is paired with a separate native run with 14.49% and 17.40% switching and 21.91% accuracy.

	Padding		Content shift			
Method	Switch	Flip	Switch	Flip	Acc.	Tok.
Native	14.50	4.38	17.12	5.44	21.74	1.00
Pad-to-grid (7)	0.42	0.11	20.75	4.61	20.40	0.82
GridPC (7)	0.47	0.13	1.25	0.33	21.19	0.82
GridPC (12)	0.20	0.09	0.86	0.33	22.96	1.46

G CORRECTNESS ACROSS PREPROCESSING TRANSITIONS

For a transition pair, let $c^-, c^+ \in \{0, 1\}$ denote target-box correctness. A correctness flip is $\mathbf{1}[c^- \neq c^+]$, including both correct-to-incorrect and incorrect-to-correct changes. Unparsed answers are

incorrect. Table 15 uses the different-configuration condition from App. B.3. Nearly equal mean accuracies coexist with a 9.72% correctness-flip rate because opposing changes offset one another.

Table 15: **Similar mean accuracies conceal correctness changes across transition pairs.** Each input is resized or tiled according to its selected preprocessing configuration. Correctness flip rate measures changes in correctness. Values are screenshot-weighted percentages. The All models row weights the eight models equally.

Model	I^- accuracy	I^+ accuracy	Correctness flip rate
Qwen3-VL-8B	46.18	46.12	8.83
UI-Venus-1.5-8B	59.52	59.64	7.93
Qwen2.5-VL-3B	20.98	21.34	8.28
GTA1-7B	49.82	49.31	9.59
UI-TARS-1.5-7B	41.40	42.37	11.39
GUI-Actor-7B	38.71	38.22	7.95
LocateAnything-3B	63.25	63.04	10.81
InternVL3.5-8B	20.80	20.79	13.00
All models	**42.58**	**42.61**	**9.72**