LAZY MCFV: OBSERVATION-AWARE SELECTION WITH EXACT QUERY PRUNING FOR MULTI-VIEW GUI GROUNDING

Anonymous authors
Paper under double-blind review

ABSTRACT

GUI grounding connects natural-language instructions to actionable locations in interface screenshots. Multi-view observation and magnification improve the perception of small interface elements, yet different views may produce conflicting predictions. Existing aggregation methods treat these predictions as isolated coordinates and discard the observation context from which they originate. We argue that the generating view is an essential component of a grounding prediction. Minimal Common Field-of-View (MCFV) is a training-free, observation-aware selection rule that preserves this provenance by pairing each candidate with its generating view and selecting the finest view whose observation scope contains all candidate locations. It automatically retains detailed local observations when they are sufficient and broader contextual views when ambiguity requires additional coverage. The same containment structure enables Lazy MCFV to achieve exact query pruning without changing the selector's output. We establish exact output preservation and show that the required observations are determined by the selector itself rather than heuristic confidence estimates. Experiments on ScreenSpot-Pro demonstrate improved selection from shared observations and output-preserving query reduction without additional training or verification.

1 INTRODUCTION

Graphical user interface (GUI) agents carry out instructions by identifying and operating on controls in software interfaces. Progress in web, desktop, and mobile environments has made accurate visual interaction an increasingly important part of agent research (Zhou et al., 2024; Koh et al., 2024; Xie et al., 2024; Rawles et al., 2025). GUI grounding provides the connection between an instruction and an action: given a screenshot and a natural-language description, it predicts where the agent should interact (Cheng et al., 2024; Gou et al., 2025). A sound action plan can still fail when it is grounded to the wrong element (Zheng et al., 2024). Professional applications make this problem particularly demanding, with small icons, repeated controls, and dense menus spread across high-resolution screens (Li et al., 2025).

Existing research improves grounding through model learning and inference-time observation. GUI-specific models learn instruction–location alignment from screen data (Cheng et al., 2024; Gou et al., 2025; Wu et al., 2025b), strengthen visual representations, or connect perception directly to actions (Hong et al., 2024; Lin et al., 2025; Wu et al., 2025a). With a frozen model, cropping and magnification offer a complementary way to expose small targets. Recent approaches obtain several local views, aggregate their predictions, or adapt the next crop to the model's current uncertainty (Zhang et al., 2026a;b; Tang et al., 2026; Yao et al., 2026). These methods improve what the model can observe, but they raise two linked questions: which answer should be retained when the views disagree, and which observations are actually needed to make that choice?

Cropping changes context as well as resolution. Enlarging a toolbar may reveal an icon while removing the panel that distinguishes it from similar controls. Several fine views may consequently agree on a nearby distractor, even when a broader view identifies the requested element. Existing aggregation and refinement methods address disagreement through spatial clustering, model scores, or visual verification. However, these strategies operate after the association between each prediction

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.

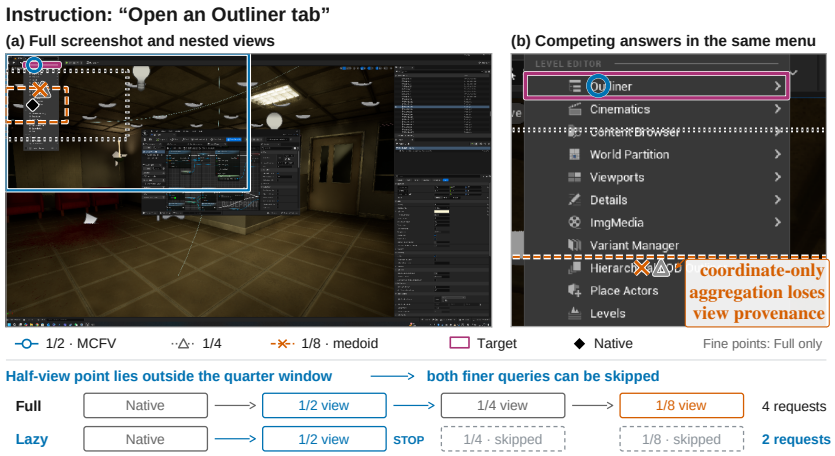


Figure 1: Observation-aware selection and exact pruning on ScreenSpot-Pro. MCFV selects the half-view prediction on *Outliner*; medoid selects a finer-view distractor.

and its generating observation has been discarded. Without this provenance, the decision cannot distinguish an answer supported by broad contextual evidence from one produced by a narrow crop that excludes the disambiguating context. The observations may therefore contain useful answers that coordinate-only decisions fail to recover.

We address this gap by treating provenance as part of the prediction. MCFV is an observation-aware selection principle that preserves this provenance by pairing each candidate with its generating view and selecting the finest generating view whose observation scope contains all local candidate locations. The rule is view-conditioned rather than coordinate-only: whether a candidate is retained depends on the spatial scope of the observation that produced it, not merely on its distance to other candidates. MCFV retains detailed local views when they are sufficient and broader contextual views when ambiguity requires additional coverage. This improvement comes from selecting among existing predictions, without additional model calls or learned verification.

The same containment structure enables exact query pruning. For nested views, once an observed candidate excludes all finer views from MCFV selection, their unseen predictions cannot alter the final decision. *Lazy MCFV* exploits this property to prune unnecessary queries while preserving the full selector's output. Answer selection and query pruning therefore follow from a unified observation-aware principle: how a prediction is selected determines which observations are needed to determine it, and a selector that retains provenance can certify which observations it does not need.

We establish coordinate- and scale-level output preservation and characterize which observations are information-theoretically necessary under a view-supported response model: every query Lazy MCFV issues is one that any always-correct deterministic algorithm must also issue. Experiments separate the two benefits: controls on shared observations measure selection quality, and independent Full/Lazy executions measure output agreement and computational savings. Complete-method comparisons with MVP, AutoFocus, and UI-Zoomer complement these tests. Controlled replay further examines whether confidence- and agreement-based stopping preserves the Full selector's decisions. Our contributions are:

- We introduce an observation-aware selection principle that treats the generating view as part of a grounding prediction and selects among model outputs based on their observation scope, without parameter updates or additional verification.
- We develop Lazy MCFV, an exact query pruning algorithm derived from the selection principle. We prove that it preserves the full selector's output and characterize the necessity of acquired queries under the view-supported response model.
- We conduct controlled evaluations on ScreenSpot-Pro across two grounding models and three output settings, demonstrating improved selection from shared observations and output-preserving query reduction with detailed accuracy, candidate, scale, stopping-rule, and efficiency analyses.

2 Related Work

GUI agents and grounding models. GUI agents must link language-level decisions to visual elements (Yao et al., 2023; Zheng et al., 2024), a connection studied through environments, benchmarks, and demonstration corpora (Deng et al., 2023; Zhou et al., 2024; Koh et al., 2024; Xie et al., 2024; Rawles et al., 2025; 2023; Li et al., 2024; Xu et al., 2025a). Grounding models learn instruction–location alignment (Cheng et al., 2024; Gou et al., 2025; Wu et al., 2025b), strengthen screen understanding (Hong et al., 2024; Baechler et al., 2024; You et al., 2024), couple grounding with action generation (Lin et al., 2025; Xu et al., 2025b; Qin et al., 2025), or predict with visual pointers (Wu et al., 2025a). MCFV operates on such models' predictions with their parameters fixed.

Adaptive computation and exact inference. Adaptive computation time and early-exit networks stop once an intermediate state is judged sufficient (**??**), confidence-based exiting extends to language models with statistical guarantees (**?**), selective prediction abstains when confidence is low (**?**), and self-consistency and test-time scaling spend additional inference to improve answers (Wang et al., 2023; Snell et al., 2025). These methods trade output fidelity, or a probabilistic guarantee of it, for computation. A different line seeks exactly the information needed to compute a fixed result: threshold algorithms access ranked lists only until a top-k answer is certified (Fagin et al., 2003), and speculative decoding preserves the output distribution while accelerating generation (**?**). Lazy MCFV belongs to this exact family. Its stopping rule is a geometric certificate that no remaining response can change a specified selector, not a confidence threshold; Appendix A.3 shows that the certificate does not extend to a medoid.

Multi-view and adaptive-zoom GUI grounding. Flexible-resolution encoders and guided visual search expose fine detail in large images (Ye et al., 2023; Wang et al., 2024; Bai et al., 2025; Wu & Xie, 2023). Within GUI grounding, cascaded search, focused cropping, and selection among regional predictions improve access to small targets (Li et al., 2025; Zhang et al., 2026a; Ye et al., 2025; Yao et al., 2026; Jiang et al., 2025; Fu et al., 2026; Luo et al., 2025; Lian et al., 2025). Two complete methods are closest to ours. MVP proposes attention-guided crops and clusters the coordinates predicted from the selected views and the full image (Zhang et al., 2026b): for a cluster set $\mathcal{G}$, it returns $G^* \in \arg\max_{G\in\mathcal{G}} |G|$ and $\widehat{\mathbf{p}} = \frac{1}{|G^*|}\sum_{\mathbf{p}_i\in G^*} \mathbf{p}_i$, with ties resolved by the attention ranks of the associated views. UI-Zoomer gates between global consensus and local refinement using agreement among sampled boxes b_i and token confidences c_i (Tang et al., 2026), scored for $N > 1$ candidates as $S = \frac{1}{N(N-1)}\sum_{i\neq j} \mathrm{IoU}(b_i, b_j) + \frac{1}{N}\sum_i c_i$; Zoom Consistency likewise treats geometric agreement as a confidence signal (Kim & Chelikavada, 2026). All of these decide from coordinates, scores, or their agreement after the generating views have been set aside. MCFV instead conditions selection on each candidate's generating window, and this retained provenance is what yields a stopping condition before every coordinate has been observed.

3 MCFV Selection and Lazy Acquisition

MCFV conditions the choice among candidate answers on the views that produced them. Lazy MCFV computes the same answer while avoiding queries that cannot affect it.

3.1 Fixed nested views

Given screenshot I of width W and height H and instruction q, a frozen grounding model first provides a full-image prediction, denoted Native, $\mathbf{n} \in [0,1]^2$. Native anchors K fixed crop windows, indexed from fine to coarse:

$$C_1 \subseteq C_2 \subseteq \cdots \subseteq C_K \subseteq [0,1]^2. \tag{1}$$

We use $K = 3$ and width/height ratios $1/8$, $1/4$, and $1/2$. These are side-length ratios; a quarter-side crop covers approximately one sixteenth of the image area. Each crop is queried using the same model and instruction. Subsequent predictions do not recenter the windows.

For crop pixel edges (l_i, t_i, u_i, b_i) and crop-normalized prediction $\mathbf{v}_i$, its coordinate on the original screenshot is

$$\mathbf{z}_i = \left(\frac{l_i + v_{ix}(u_i - l_i)}{W}, \frac{t_i + v_{iy}(b_i - t_i)}{H}\right). \tag{2}$$

Let P_i be the coordinate set returned for view i, with $|P_i| \leq 1$. The mapped coordinates are self-contained: $P_i \subseteq C_i$. Write $\mathcal{V} = \{i : |P_i| = 1\}$ and let $\mathbf{z}_i$ denote the element of P_i for $i \in \mathcal{V}$. Appendix B specifies the crop geometry and model settings. Target annotations are used only for evaluation.

3.2 Observation-aware selection

Introduce the default answer $(\mathbf{z}_{K+1}, K+1) = (\mathbf{n}, K+1)$. For $P = (P_1, \ldots, P_K)$, MCFV selects

$$r^*(P) = \min(\{i \in \mathcal{V} : \mathbf{z}_j \in C_i \text{ for all } j \in \mathcal{V}\} \cup \{K+1\}), \tag{3}$$

$$F(P) = (\mathbf{z}_{r^*(P)}, r^*(P)). \tag{4}$$

The largest window represented in the candidate set is feasible because it contains every smaller window. The minimum therefore chooses the finest view that covers all local candidate locations, and Equation 4 returns that view's own prediction. Native supplies the crop anchor and default answer; only local candidates enter the containment constraint.

The rule balances detail and spatial coverage. A fine-view point automatically lies inside all broader windows, whereas a coarse-view point can exclude a finer window. MCFV retains a small view when all candidate locations fit within it and selects a broader view otherwise. A Euclidean medoid instead minimizes $\sum_{j \in \mathcal{V}} \|\mathbf{z}_i - \mathbf{z}_j\|_2$ over candidate indices. It favors a point near the other answers, regardless of its generating window. MCFV is instead a function of the pairs $(\mathbf{z}_i, C_i)$, so identical coordinates produced from different windows can yield different selections. It can therefore select a broad-view answer even when fine-view predictions cluster elsewhere, as in Figure 1. The rule uses only coordinates and crop bounds; it adds no trained selector or confidence threshold.

3.3 Exact coarse-to-fine querying

Suppose an observed point lies outside the next finer window. That window and all its descendants are infeasible in Equation 3. Their unseen points also lie inside every retained feasible broader window, so they cannot change its feasibility. No remaining query can alter the final choice.

Algorithm 1 maintains a current answer and the set S of observed points. It tests containment before each query. If the test fails, it returns the current answer; otherwise, it queries the view and incorporates its returned candidate. All previous points remain in the test. For example, a half-view point inside the quarter window but outside the eighth still excludes the eighth, even if a later quarter-view prediction lies inside it. Testing only the latest point would lose this information.

Lazy makes at most K local queries. The listed implementation stores at most K points and performs $O(K^2)$ inexpensive membership tests. With $K = 3$, model inference dominates this geometric computation. A skipped view requires neither image preparation nor model inference. Full and Lazy differ only in how they compute Equation 4.

4 Theoretical Guarantees

We establish output preservation and the necessity of Lazy MCFV's queries. The proofs use the fixed support of each view's response, rather than any assumption about which prediction is correct. The second result concerns which observations an algorithm must access to determine the output, not how long each access takes.

Output preservation.

Proposition 1 (Output preservation)**.** *For fixed candidate sets $P_i \subseteq C_i$, $|P_i| \leq 1$, and nested windows satisfying Equation 1, Algorithm 1 returns $F(P)$, including both the predicted coordinate and selected scale.*

Proof idea. At a queried view, all earlier points lie inside its window; all later points must also lie inside it by nesting. A returned point thus becomes the finest feasible candidate seen so far. If the guard stops, its witness excludes every unqueried candidate while the current answer remains feasible. This invariant identifies the same final answer as Full MCFV.

Algorithm 1 Lazy MCFV

Require: Native $\mathbf{n}$ and fixed nested windows $C_1, \ldots, C_K$
$(\mathbf{p}, r) \leftarrow (\mathbf{n}, K+1)$; $S \leftarrow \varnothing$
for $i = K, K-1, \ldots, 1$ **do**
  if $\exists \mathbf{u} \in S : \mathbf{u} \notin C_i$ **then**
    return $(\mathbf{p}, r)$
  end if
  $P_i \leftarrow \mathrm{query}(i)$
  for each $\mathbf{z}_i \in P_i$ **do**
    $S \leftarrow S \cup \{\mathbf{z}_i\}$; $(\mathbf{p}, r) \leftarrow (\mathbf{z}_i, i)$
  end for
end for
return $(\mathbf{p}, r)$

Necessary queries. Fix Native and strictly nested, positive-area, half-open rectangles. The response model is view-supported: each view returns at most one coordinate, and that coordinate lies within the view. Let

$$\mathfrak{P}_{\leq 1}(C) = \{A \subseteq C : |A| \leq 1\}, \qquad \mathcal{Z} = \prod_{i=1}^{K} \mathfrak{P}_{\leq 1}(C_i). \tag{5}$$

A deterministic algorithm observes P_i only by querying i and must return the coordinate component $G(P)$ of $F(P)$ for every $P \in \mathcal{Z}$.

Proposition 2 (Instance-wise necessary queries)**.** *On every $P \in \mathcal{Z}$, any deterministic algorithm correct on all of $\mathcal{Z}$ must query every index queried by Algorithm 1. Lazy MCFV therefore minimizes the number of local queries on each input under this response model, even when only the final coordinate is required.*

Proof idea. For any queried index i, replace only P_i with $\{\mathbf{u}\}$, where $\mathbf{u} \in C_i \setminus C_{i-1}$ differs from the original output and $C_0 = \varnothing$. All broader observed points lie inside C_i, while $\mathbf{u}$ excludes every finer window. The new output is $\mathbf{u}$. An algorithm that skips i cannot distinguish these two inputs.

The support and nesting assumptions define the lower bound's scope. Equal windows or restricted discrete coordinate supports can admit fewer queries. Proposition 1 preserves the specified selector, including its grounding errors; Proposition 2 characterizes information access under the stated response model, not real-model latency: an algorithm that skips a query issued by Lazy MCFV lacks the information to determine the output on some admissible input. Appendix A gives full proofs and a counterexample showing that the same stopping condition need not preserve medoid selection. Independent execution in Section 5.4 tests output agreement and serial runtime with actual model calls.

5 Experiments

We evaluate whether MCFV selects better answers from the same observations, how it compares with complete multi-view methods, and whether Lazy preserves its output with less computation.

5.1 Experimental setup

Data and models. We use all 1,581 ScreenSpot-Pro tasks, comprising 977 text and 604 icon targets across six professional application domains (Li et al., 2025). We evaluate GUI-Actor-7B and Qwen2.5-VL-7B; the latter is run with point and box outputs. Box centers are used for point-in-target-box accuracy. UI-Zoomer and AutoFocus are compared with MCFV under matched box and point settings, respectively. MVP is evaluated with its complete view-proposal and aggregation procedure. Appendix B gives the model and image-processing settings.

Controls and measurements. Native predicts from the whole screenshot, while fixed zoom adds one local crop. Mean, coordinate median, medoid, maximum token score, and finest-/coarsest-view selection use the same local observations as Full MCFV. These comparisons isolate selection

from the benefit of additional views. Every accuracy uses the full task denominator. Paired 95% intervals use 10,000 source-path bootstrap resamples. All method and execution choices were fixed before final evaluation; Appendix B documents label availability during development. The intervals describe variation on the evaluated data, without adjustment for method selection.

Matched Qwen runs use one RTX 5090, BF16, and serial Hugging Face inference. Synchronized pipeline timing includes image reading, preprocessing, generation, and selection, but excludes model loading and result writing. Full and Lazy obtain their own model responses. Controls computed from recorded predictions have no independently measured latency.

5.2 Selection on shared observations

Table 1 shows that the strongest fixed scale depends on the model and output setting. Half is best for GUI-Actor and Qwen points, whereas eighth is best for Qwen boxes. MCFV improves over each fixed scale in all three configurations. In the box setting, it obtains 744 correct predictions versus 657 for fixed eighth, a gain of 5.50 percentage points. Always choosing the finest or coarsest view also falls below MCFV in both Qwen settings.

Table 1: Selection accuracy (%) on ScreenSpot-Pro. Multi-view selectors share the local observations within each column.

	GUI-Actor-7B	Qwen2.5-VL-7B	
Selector	Pointer	Point	Box
Native	44.21	27.07	18.47
Fixed $1/8$	56.48	44.15	41.56
Fixed $1/4$	56.67	45.92	40.99
Fixed $1/2$	57.18	46.74	34.79
Mean	52.94	42.06	34.47
Coordinate median	56.99	48.32	42.82
Medoid	57.24	48.83	43.14
Max token score	–	50.79	43.07
Finest view	–	47.00	42.38
Coarsest view	–	47.06	34.79
MCFV	**59.01**	**52.06**	**47.06**

Each column uses all 1,581 tasks. Native uses no local view; fixed zoom uses one. Multi-view selection uses the three-scale observations. Dashes denote unreported controls.

On the same local predictions, MCFV improves over medoid by 1.77 points with GUI-Actor, 3.23 with Qwen points, and 3.92 with Qwen boxes. Their respective intervals are $[0.57, 3.04]$, $[1.58, 4.87]$, and $[2.15, 5.69]$. The GUI-Actor result is 933 correct tasks versus medoid's 905 and fixed half's 904. In the box setting, 135 corrections outweigh 73 regressions, giving 62 additional correct predictions. These gains reflect the choice among existing answers rather than new observations.

Token-score selection is a stronger alternative in the point setting. It reaches 50.79%, versus MCFV's 52.06%; the 1.27-point difference has interval $[-0.19, 2.72]$, leaving the comparison inconclusive. With boxes, token-score selection reaches 43.07%, and the MCFV gain is 3.98 points ($[2.28, 5.69]$). Appendix C.1 reports both target types, including the point-icon subgroup where token scores are more accurate.

5.3 Complete multi-view and adaptive-zoom methods

Adaptive-zoom methods determine their own observations and cannot be run on shared views, so Table 2 compares complete inference pipelines rather than isolated selectors, with each method's output format beside its name; Table 1 isolates selection alone. The matched comparisons are UI-Zoomer versus MCFV with boxes and AutoFocus versus MCFV with points. The same output label does not imply identical image budgets or prompts across independently evaluated methods.

AutoFocus retains uncertainty-based proposals, staged sampling, crop refinement, and visual selection. Lazy MCFV obtains 823 correct predictions versus AutoFocus's 751, a 4.55-point gain ($[2.53, 6.58]$), with 5,184 versus 21,098 requests. In that paired run, requests fall by 75.43% and

Table 2: Complete-method results on ScreenSpot-Pro with Qwen2.5-VL-7B. UI-Zoomer reports three-seed means, MVP uses its own complete pipeline; its forward count is not interchangeable with the model-request unit used here.

Method	Text	Icon	All	Requests/task
UI-Zoomer (box)	52.27	18.93	39.53	9.00
Full MCFV (box)	61.11	24.34	47.06	3.98
Lazy MCFV (box)	61.11	24.34	47.06	3.32
MVP (point)	44.73	13.08	32.64	–
AutoFocus (point)	63.97	20.86	47.50	13.34
Lazy MCFV (point)	69.40	24.01	52.06	3.28

pipeline time by 76.03%. These are complete-method differences; the smaller savings attributable to exact pruning alone are isolated below. Text and icon accuracy both improve, while Creative is the one application domain with a one-task decrease.

UI-Zoomer uses sampled-box agreement and token confidence to choose between global consensus and local refinement. Under the shared box setting, MCFV reaches 47.06%, compared with its three-seed mean of $39.53 \pm 0.54\%$. Per-seed gains range from 7.02 to 8.10 points. The independent MVP reproduction obtains 516 correct predictions (32.64%) with attention-guided views and spatial clustering. Table 1 complements these complete-method results by holding observations fixed. Appendix C reports seed-specific and domain-level results.

5.4 Exact output-preserving execution

Independent Full/Lazy execution tests the computational contribution. In each Qwen setting, all 1,581 final coordinates and selected scales agree. Native and every local response observed by both versions also agree. Table 3 reports these task-level checks alongside measured cost.

Table 3: Independent Full/Lazy execution on ScreenSpot-Pro.

Output	Execution	Requests	Mean (s)	Accuracy (%)
Box	Full	6,297	3.708	47.06
	Lazy	5,242	3.264	47.06
Point	Full	6,165	9.989	52.06
	Lazy	5,184	8.253	52.06

Lazy removes 1,055 box requests (16.75%) and 981 point requests (15.91%). Mean pipeline time falls by 11.97% and 17.37%, respectively. Containment prunes at least one view on 691 box tasks and 631 point tasks. Every task in these pruned groups is faster with Lazy in the recorded runs. Across all tasks, Lazy is faster on 1,199 box tasks and 1,111 point tasks; it is slower on the others. The method preserves the answer but does not guarantee lower latency on each individual task.

The reduction also appears beyond the mean. Box-setting median latency falls from 3.475 to 3.232 seconds and the 95th percentile from 5.000 to 4.667 seconds. For points, the corresponding changes are 8.056 to 7.310 and 17.079 to 13.722 seconds. Lazy has a lower mean under every recorded execution-order stratum. Appendix C.4 retains the full summaries, including maximum times and negative paired differences.

Controlled stopping-rule replay. We compare stopping rules on recorded Full predictions for all 1,581 ScreenSpot-Pro tasks in the Qwen box setting. Every rule follows the same fixed coarse-to-fine schedule and applies MCFV to its acquired prefix. We report all prespecified confidence thresholds and an adjacent-prediction agreement rule; Appendix E gives the protocols and complete results.

In Table 4, confidence stopping at $\tau = 0.9$ retains 47.06% accuracy but changes one coordinate-and-scale decision, saving only three requests. At $\tau = 0.5$, savings reach 33.43% and accuracy falls to 41.37%. Agreement changes 642 coordinate-or-scale outputs, with 31 corrections and 53 regressions in target correctness. Lazy preserves all recorded coordinates and scales while saving 1,055 requests (16.75%); its replay matches the independent Full/Lazy execution task by task.

Table 4: Stopping-rule controls on recorded Qwen box predictions.

Stopping rule	Coordinate/scale mismatches	Requests	Accuracy (%)
Full MCFV	0/0	6,297	47.06
Confidence $\tau = 0.5$	834/835	4,192	41.37
Confidence $\tau = 0.7$	89/91	6,089	46.81
Confidence $\tau = 0.9$	1/1	6,294	47.06
Agreement $\epsilon = 0.02$	638/642	5,555	45.67
Lazy MCFV	0/0	5,242	47.06

5.5 View-count ablation

Does the third view improve selection enough to justify its cost? We apply the unchanged MCFV rule to every two-view subset of the fixed scales and compare them with the three-view configuration (Table 5). All decisions and query counts are reconstructed from the same Full predictions. The corresponding three-view replay agrees with independent execution, while two-view replays provide no new latency measurements.

Table 5: Scale subsets on all 1,581 tasks per output setting. Counts include Native.

Output	Local scales	Correct	All (%)	Text	Icon	Full req.	Lazy replay
Box	$\{1/8, 1/4\}$	704	44.53	57.52	23.51	4725	4279
	$\{1/8, 1/2\}$	719	45.48	59.77	22.35	4725	4114
	$\{1/4, 1/2\}$	690	43.64	58.14	20.20	4725	4361
	$\{1/8, 1/4, 1/2\}$	744	47.06	61.11	24.34	6297	5242
Point	$\{1/8, 1/4\}$	746	47.19	63.87	20.20	4637	4281
	$\{1/8, 1/2\}$	804	50.85	67.96	23.18	4637	4091
	$\{1/4, 1/2\}$	810	51.23	68.58	23.18	4637	4287
	$\{1/8, 1/4, 1/2\}$	823	52.06	69.40	24.01	6165	5184

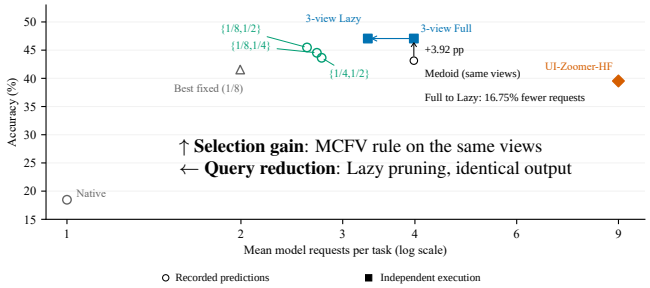


Figure 2: Accuracy–request trade-off in the Qwen box setting.

With box outputs, $\{1/8, 1/2\}$ is the strongest pair, reaching 45.48% with 4,114 Lazy requests. Adding the quarter view yields 25 more correct answers and raises accuracy to 47.06%, with a paired gain of 1.58 points ($[0.57, 2.66]$). The two-view option uses 21.52% fewer requests than the three-view configuration. Figure 2 places this choice alongside the gains from selection and exact pruning.

For point outputs, $\{1/4, 1/2\}$ is the most accurate pair at 51.23%. Three views add 13 correct answers, but the 0.82-point interval $[-0.25, 1.90]$ leaves this increment inconclusive. The pair $\{1/8, 1/2\}$ requires fewer requests still, at 4,091, with 50.85% accuracy. The preferred pair therefore depends on the output setting and budget. Appendix C.2 reports all paired intervals. These comparisons vary view count within the fixed scale schedule, not the crop ratios themselves.

5.6 Candidate and scale analysis

Selecting an available correct answer. At least one local candidate is correct on 867 box-setting tasks. Table 6 isolates selection on this same set: medoid chooses correctly on 682 tasks and MCFV on 744. Conditional selection success increases from 78.66% to 85.81%, and missed available

answers decrease from 185 to 123. The resulting 33.51% reduction concerns this selection-error category, not all grounding errors. The remaining 714 tasks have no correct local candidate for either rule to select. Target labels enter only this retrospective analysis.

Table 6: Selection on the 867 box-setting tasks containing a correct local candidate.

Outcome	Medoid	MCFV
Correct candidate selected	682	744
Available correct candidate missed	185	123
Conditional selection success (%)	78.66	85.81

Choosing between detail and context. Relative to medoid's generating view, MCFV chooses a larger view on 447 tasks, with 68 corrections and 39 regressions. It chooses a smaller view on 638 tasks, with 67 corrections and 34 regressions. The two directions contribute 29 and 33 net correct predictions. Thus the gain is not explained by always preferring broader context. In the box setting, both target types improve over medoid: text accuracy is 61.11% versus 56.40%, and icon accuracy is 24.34% versus 21.69%. Appendix D shows corrections and regressions in both scale directions.

The GUI-Actor results show how selection improves on the best fixed scale. MCFV selects the eighth, quarter, and half views on 1,252, 159, and 170 tasks. On the eighth-selected tasks, it obtains 857 correct answers versus 829 for fixed half; on the quarter-selected tasks, the counts are 46 versus 45. Half-selected answers coincide with fixed half. These branches account for all 29 additional correct answers over the strongest fixed-scale control, with most coming from retaining fine detail when the candidate positions fit within the smaller view.

Application domains and target size. Appendix C.3 reports the box-setting breakdown by application domain and annotated target area. MCFV exceeds the UI-Zoomer seed mean in all six domains and gains 20, 14, 4, 11, 13, and zero correct predictions over medoid in CAD, Creative, Development, OS, Office, and Scientific, respectively; the Scientific tie consists of 14 corrections and 14 regressions. Across four pre-specified target-area bins, the gain over medoid is positive in the three bins below 0.20% of screenshot area (5.79, 4.33, and 3.69 percentage points, with paired intervals excluding zero) and reverses on the 55 largest-area tasks (-7.27 points, interval $[-20.00, 5.45]$). The positive aggregate therefore contains both improvements and task-level trade-offs, and the breakdown describes where the gains occur rather than establishing that area alone determines the preferable selector.

6 Conclusion

MCFV pairs each grounding prediction with its generating view and selects the finest candidate-bearing view containing all local candidate locations. On ScreenSpot-Pro, this rule improves the use of shared observations across two grounding models. The same containment structure determines when further responses cannot change the answer. Lazy MCFV exploits it to preserve Full MCFV's coordinate and selected scale while reducing model requests and measured serial inference time. Theoretical guarantees characterize this exact computation under fixed, self-contained responses. The fixed anchor and incorrect coarse predictions remain limits of the current selection rule. The central result is a direct connection between how an answer is selected and which visual observations are needed to determine it.

AI use statement

In this work, we used generative AI tools for <assist in the writing of proofs, design or provide feedback on research methodology or experiments, assist with translation>. We have not used generative AI tools for <generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, implement methods, clean and reformat dataset, support qualitative and thematic data analysis, interpret results>. Additionally, we used generative AI tools for <draft parts of a research paper, summarize or analyse existing literature, edit a research paper to

improve readability, identify relevant literature>. We have reviewed all AI-assisted work. LLM-generated text was verified for correctness by authors". We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility Statement

Section 3 defines MCFV and Lazy acquisition; Section 4 states their guarantees. Appendix A gives complete proofs, Appendix B specifies crop geometry and experimental settings, and Appendices C–D provide supplementary quantitative results and recorded examples. Appendix E specifies the stopping-rule replay and reports all prespecified operating points and return-rule controls. The source code has been uploaded as supplementary material. All baseline models and datasets used in this study are publicly available, such as Qwen2.5-VL-7B, GUI-Actor-7B, MVP, UI-Zoomer, AutoFoucs and ScreenSpot-Pro.

References

Gilles Baechler, Srinivas Sunkara, Maria Wang, Fedir Zubach, Hassan Mansoor, Vincent Etter, Victor Cărbune, Jason Lin, Jindong Chen, and Abhanshu Sharma. ScreenAI: A vision-language model for UI and infographics understanding. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2024. URL https://arxiv.org/abs/2402.04615.

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*, 2025. doi: 10.48550/arXiv.2502.13923. URL https://arxiv.org/abs/2502.13923.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. SeeClick: Harnessing GUI grounding for advanced visual GUI agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9313–9332, 2024. doi: 10.18653/v1/2024.acl-long.505. URL https://aclanthology.org/2024.acl-long.505/.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2Web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2306.06070.

Ronald Fagin, Amnon Lotem, and Moni Naor. Optimal aggregation algorithms for middleware. *Journal of Computer and System Sciences*, 66(4):614–656, June 2003. doi: 10.1016/S0022-0000(03)00026-6. URL https://www.wisdom.weizmann.ac.il/~naor/PAPERS/middle_agg.pdf.

Zichuan Fu, Shirong Wang, Wenlin Zhang, Guojing Li, Yimin Deng, Jingtong Gao, Junjia Qi, Hanyu Yan, Yefeng Zheng, Xiaopeng Li, Wanyu Wang, Xian Wu, and Xiangyu Zhao. GUI-Lens: Coarse-to-fine cropping for GUI grounding with general-purpose VLMs. *arXiv preprint arXiv:2608.03270*, 2026. URL https://arxiv.org/abs/2608.03270.

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

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. CogAgent: A visual language model for GUI agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. URL https://arxiv.org/abs/2312.08914.

Zhiyuan Jiang, Shenghao Xie, Wenyi Li, Wenqiang Zu, Peihang Li, Jiahao Qiu, Siqi Pei, Lei Ma, Tiejun Huang, Mengdi Wang, and Shilong Liu. Zoom in, click out: Unlocking and evaluating the

potential of zooming for GUI grounding. *arXiv preprint arXiv:2512.05941*, 2025. doi: 10.48550/arXiv.2512.05941. URL https://arxiv.org/abs/2512.05941.

Keon Kim and Krish Chelikavada. Zoom consistency: A free confidence signal in multi-step visual grounding pipelines. *arXiv preprint arXiv:2604.15376*, 2026. URL https://arxiv.org/abs/2604.15376.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024. URL https://arxiv.org/abs/2401.13649.

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

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. On the effects of data scale on UI control agents. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2406.03679.

Shuquan Lian, Yuhang Wu, Jia Ma, Yifan Ding, Zihan Song, Bingqi Chen, Xiawu Zheng, Hui Li, and Rongrong Ji. UI-AGILE: Advancing GUI agents with effective reinforcement learning and precise inference-time grounding. *arXiv preprint arXiv:2507.22025*, 2025. URL https://arxiv.org/abs/2507.22025.

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

Tiange Luo, Lajanugen Logeswaran, Justin Johnson, and Honglak Lee. Visual test-time scaling for GUI agent grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. URL https://arxiv.org/abs/2505.00684.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, Wanjun Zhong, Kuanye Li, Jiale Yang, Yu Miao, Woyu Lin, Longxiang Liu, Xu Jiang, Qianli Ma, Jingyu Li, Xiaojun Xiao, Kai Cai, Chuang Li, Yaowei Zheng, Chaolin Jin, Chen Li, Xiao Zhou, Minchao Wang, Haoli Chen, Zhaojian Li, Haihua Yang, Haifeng Liu, Feng Lin, Tao Peng, Xin Liu, and Guang Shi. UI-TARS: Pioneering automated GUI interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025. URL https://arxiv.org/abs/2501.12326.

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

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. Android in the Wild: A large-scale dataset for android device control. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2307.10088.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b623663fd9b874366f3ce019fdfdd44-Abstract-Conference.html.

Fei Tang, Bofan Chen, Zhengxi Lu, Tongbo Chen, Songqin Nong, Tao Jiang, Wenhao Xu, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. UI-Zoomer: Uncertainty-driven adaptive zoom-in for GUI grounding. *arXiv preprint arXiv:2604.14113*, 2026. URL https://arxiv.org/abs/2604.14113.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024. URL https://arxiv.org/abs/2409.12191.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal LLMs. *arXiv preprint arXiv:2312.14135*, 2023. URL https://arxiv.org/abs/2312.14135.

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

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

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems*, 2024. URL https://arxiv.org/abs/2404.07972.

Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. AgentTrek: Agent trajectory synthesis via guiding replay with web tutorials. In *International Conference on Learning Representations*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/c681fb2bf1d785fbc766f3ea14758aab-Abstract-Conference.html.

Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. Aguvis: Unified pure vision agents for autonomous GUI interaction. In *Proceedings of the International Conference on Machine Learning*, 2025b. URL https://arxiv.org/abs/2412.04454.

Ruilin Yao, Shegnwu Xiong, Tianyu Zou, Shili Xiong, and Yi Rong. AutoFocus: Uncertainty-aware active visual search for GUI grounding. *arXiv preprint arXiv:2605.02630*, 2026. URL https://arxiv.org/abs/2605.02630.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2210.03629.

Jiabo Ye, Anwen Hu, Haiyang Xu, Qinghao Ye, Ming Yan, Guohai Xu, Chenliang Li, Junfeng Tian, Qi Qian, Ji Zhang, Qin Jin, Liang He, Xin Lin, and Fei Huang. UReader: Universal OCR-free visually-situated language understanding with multimodal large language model. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2841–2858, 2023. doi:

10.18653/v1/2023.findings-emnlp.187. URL https://aclanthology.org/2023.findings-emnlp.187/.

Xianhang Ye, Yiqing Li, Wei Dai, Miancan Liu, Ziyuan Chen, Zhangye Han, Hongbo Min, Jinkui Ren, Xiantao Zhang, Wen Yang, and Zhi Jin. GUI-ARP: Enhancing grounding with adaptive region perception for GUI agents. *arXiv preprint arXiv:2509.15532*, 2025. URL https://arxiv.org/abs/2509.15532.

Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, and Zhe Gan. Ferret-UI: Grounded mobile UI understanding with multimodal LLMs. In *Proceedings of the European Conference on Computer Vision*, 2024. URL https://arxiv.org/abs/2404.05719.

Borui Zhang, Bo Zhang, Bo Wang, Wenzhao Zheng, Yuhao Cheng, Liang Tang, Yiqiang Yan, Jie Zhou, and Jiwen Lu. BAMI: Training-free bias mitigation in GUI grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 34596–34605, 2026a. URL https://openaccess.thecvf.com/content/CVPR2026/html/Zhang_BAMI_Training-Free_Bias_Mitigation_in_GUI_Grounding_CVPR_2026_paper.html.

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, 2026b. URL https://openaccess.thecvf.com/content/CVPR2026/html/Zhang_MVP_Multiple_View_Prediction_Improves_GUI_Grounding_CVPR_2026_paper.html.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. GPT-4V(ision) is a generalist web agent, if grounded. In *Proceedings of the International Conference on Machine Learning*, 2024. URL https://arxiv.org/abs/2401.01614.

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

A Proofs and Scope of the Guarantees

A.1 Proof of Proposition 1

Proof. Whenever Algorithm 1 queries index i, every point observed at a broader index lies in C_i; otherwise the guard would already have stopped. Every point returned by a later, finer query also lies in C_i by self-containment and nesting. Thus, whenever $P_i = \{\mathbf{z}_i\}$, index i is feasible for the complete tuple P. The stored candidate becomes the smallest feasible candidate-bearing index seen so far.

Suppose the guard stops before querying index i. Its witness lies outside C_i and every finer window. No unqueried index can therefore satisfy Equation 3. The stored candidate remains feasible, because every point in an unqueried finer window also belongs to its broader generating window. It is consequently the smallest feasible candidate-bearing index in the complete tuple, giving $F(P)$.

If the loop finishes, it has examined every index. The same invariant identifies the last stored local candidate as the smallest feasible one. When $\mathcal{V} = \varnothing$, the initialization returns $(\mathbf{n}, K+1)$, exactly as prescribed by Equation 3. Both the coordinate and the selected index agree with Full MCFV in every case. □

A.2 Proof of Proposition 2

Proof. Set $C_0 = \varnothing$. Under the stated rectangle assumptions, each band $C_i \setminus C_{i-1}$ has positive area. For an index i queried by Lazy MCFV on P, choose

$$\mathbf{u} \in C_i \setminus C_{i-1}, \qquad \mathbf{u} \neq G(P). \tag{6}$$

Such a point exists because the band contains more than one coordinate. Construct P' by replacing only P_i with $\{\mathbf{u}\}$.

At the query of i, all broader observed points lie in C_i by the guard condition. All finer points also lie in C_i by nesting. Hence index i is feasible for P'. The point $\mathbf{u}$ lies outside every finer window, so no finer candidate is feasible. It follows that $G(P') = \mathbf{u} \neq G(P)$.

Suppose another deterministic algorithm skipped i on P. All responses along its query path are unchanged in P'. It follows the same path, never queries i, and produces the same coordinate on both inputs. That contradicts correctness on all of $\mathcal{Z}$. The argument holds for every index acquired by Lazy MCFV, so every always-correct deterministic algorithm must make at least those queries on P. □

A.3 Why the certificate does not preserve a medoid

Consider three nested windows and two fixed candidate points:

$$C_3 = [0.25, 0.75)^2, \quad C_2 = [0.375, 0.625)^2, \quad C_1 = [0.4375, 0.5625)^2,$$

$$\mathbf{z}_3 = (0.70, 0.50), \qquad \mathbf{z}_1 = (0.50, 0.50).$$

Two completions set $\mathbf{z}_2 = (0.40, 0.50)$ or $\mathbf{z}_2 = (0.60, 0.50)$. MCFV returns $\mathbf{z}_3$ in both, since $\mathbf{z}_3 \notin C_2$. The Euclidean distance sums for $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3)$ are $(0.30, 0.40, 0.50)$ in the first completion and $(0.30, 0.20, 0.30)$ in the second. Thus the unique medoid changes from $\mathbf{z}_1$ to $\mathbf{z}_2$ while the observed coarse point remains fixed (Figure 3). An unseen response irrelevant to MCFV can change medoid selection. This example concerns the particular stopping condition, rather than all possible exact algorithms for medoid.

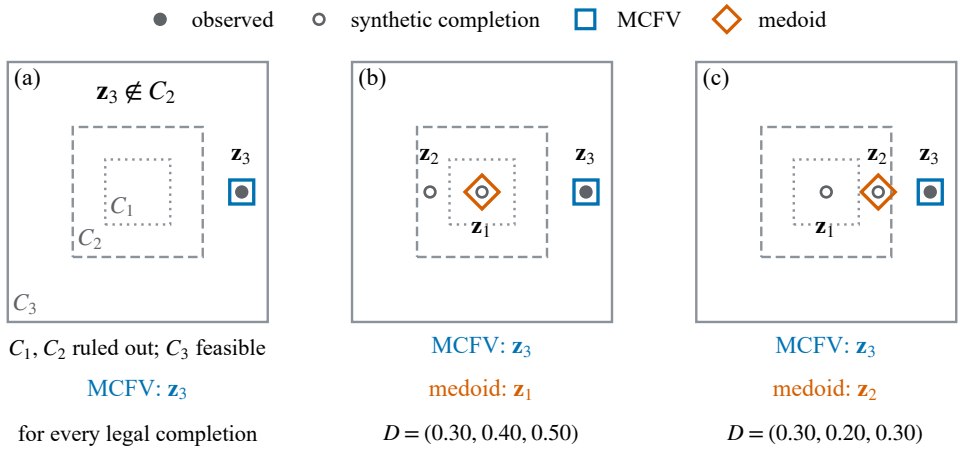


Figure 3: An unseen response can change the unique medoid without changing MCFV. D lists each candidate's sum of distances to the others. The two completions illustrate the selector-specific stopping condition.

A.4 Nesting and coordinate support

Output preservation permits equal windows. The lower-bound proof requires more: each band $C_i \setminus C_{i-1}$ must contain a coordinate different from the current output. Strictly nested, positive-area, half-open rectangles satisfy this condition, whereas a restricted finite set of patch centers may not.

For example, let $K = 2$ and $C_1 = C_2$. A finest-first algorithm queries index 1 and returns its point whenever P_1 contains one, since that candidate is feasible regardless of P_2. Otherwise it queries index 2 and applies Equation 3. This algorithm is exact on the full candidate-set space, but uses one query on singleton-P_1 inputs for which the descending algorithm uses two. Proposition 1 still applies; the strict-nesting assumption in Proposition 2 excludes this case.

The product space specifies which response tuples the response model permits. It does not assert independent model errors or that a real model realizes every tuple. Query minimality also assigns one unit to each local query; it does not imply minimum latency when image costs differ or inference is parallelized.

B Experimental Settings

B.1 Crop geometry

For reciprocal scale $d \in \{8, 4, 2\}$, crop dimensions are

$$w_d = \max(1, \lfloor W/d \rfloor), \qquad h_d = \max(1, \lfloor H/d \rfloor). \tag{7}$$

With Native $\mathbf{n} = (n_x, n_y)$, the crop origin is

$$l_d = \min\{\max\{\mathrm{round}(Wn_x - w_d/2), 0\}, W - w_d\}, \tag{8}$$

$$t_d = \min\{\max\{\mathrm{round}(Hn_y - h_d/2), 0\}, H - h_d\}. \tag{9}$$

Rounding uses the ties-to-even convention. Set $u_d = l_d + w_d$ and $b_d = t_d + h_d$. The crop is translated within the screenshot without padding. Containment uses the actual integer-derived edges, with no additional tolerance. Pointer and point coordinates use half-open supports; box centers use closed supports. Coordinates are confined to the processed view before Equation 2 maps them to the screenshot.

B.2 Models and inference

Table 7 summarizes the settings that affect the image evidence and computational cost. Native and MCFV's local predictions use the same instruction and greedy decoding. Point-setting crops use bicubic resizing to the original image dimensions before processor resizing. The box setting uses its specified local pixel budget. The shared Qwen implementation preserves visual-attention segment boundaries and processes equally sized segments together.

Table 7: Model and image-processing settings.

Configuration	Setting
GUI-Actor-7B	BF16; rank-0 visual-pointer decoding; maximum 5,720,064 pixels.
Qwen2.5-VL-7B	One RTX 5090; BF16; serial Hugging Face inference; TF32 disabled.
Qwen point	Processor range 3,136–12,845,056 pixels; maximum 1,024 generated tokens; repetition penalty 1.05.
Qwen box	Global resizing 2M–4.8M pixels; local resizing 1M–4M; processor range 10,000–5,000,000 pixels; maximum 128 tokens; repetition penalty 1.0.
MCFV	Three fixed side ratios $1/8$, $1/4$, $1/2$; greedy Native and local predictions; top-$p = 1$.

Complete methods. UI-Zoomer-HF retains eight global samples at temperature 0.9, gate threshold $\tau = 1.5$, crop factor $\gamma = 1.0$, minimum crop edge 512 pixels, and 75% candidate retention. It uses seeds 114514, 114515, and 114516. AutoFocus-HF retains uncertainty proposals, visual refinement and selection, and staged temperatures 0.1, 0.3, 0.5, and 0.75, with task seeds 20260910 plus task position. Its Native input is shared with the matched MCFV setting; subsequent refinement follows its own procedure. MVP retains attention-guided proposals and cluster-based point aggregation. The HF adaptations are serial evaluations rather than measurements of the original authors' serving systems.

Selection controls. Mean and coordinate median operate on the local coordinates. Euclidean medoid minimizes the sum of distances, breaking ties in the order quarter, eighth, half. The finest-/coarsest-view controls choose by generating-view size. Maximum token score chooses the largest length-normalized model score: the geometric mean of generated-token probabilities for boxes, including EOS, and reciprocal joint-coordinate perplexity for points. The latter is not a calibrated probability of target correctness. These controls reuse recorded Full predictions and add no inference.

B.3 Scoring, data use, and timing

Accuracy is the fraction of all 1,581 tasks whose final point lies in the annotated target box intersected with the image canvas, including its boundary. No task is removed from the denominator. Paired wins and losses count tasks on which only one method is correct. The main 95% intervals are percentile intervals from 10,000 source-path resamples. Pro contains 1,581 source-path groups. The GUI-Actor, Full/Lazy, matched Qwen, and cached-control analyses use seeds 20260910, 20260912, 20260915, and 20260920, respectively. Source paths define the resampling groups rather than verified image-content deduplication.

Method development used label-exposed observations from 1,016 Pro tasks. The remaining 565 tasks reflect the availability of multiscale observations rather than an independently sampled holdout. Final evaluations fix the method and execution settings but reuse this benchmark population. Intervals are descriptive and are not adjusted for method selection or simultaneous comparisons. The area analysis in Appendix C.3 uses 5,000 resamples per group with seed 20260917.

Every timing comparison uses independently executed arms on the same tasks. Full/Lazy point order alternates across tasks; the AutoFocus/Lazy point run is collected separately, also with alternating order. The box experiment rotates five arms: three UI-Zoomer seeds, Full MCFV, and Lazy MCFV. Timers use CUDA synchronization and cover the image-to-answer pipeline. Model loading and result writing are excluded. Full/Lazy and complete-method reductions are each computed from totals within their corresponding paired run.

C Supplementary Quantitative Results

All tables in this appendix use the same ScreenSpot-Pro population as the main text. Model settings remain fixed within each comparison.

C.1 Model-score selection

Table 8 compares maximum token score with MCFV on identical observations. The point-setting gap is smaller than the medoid gap and its overall interval crosses zero. On point-setting icons, score selection obtains 150 correct tasks versus MCFV's 145. With boxes, the aggregate MCFV gain is positive, while the icon-only interval remains inconclusive. The results therefore distinguish improvements over coordinate aggregation from improvements over model confidence.

Table 8: Maximum token score versus MCFV. Gains and paired 95% intervals are in percentage points.

Output	Group	Tasks	Score (%)	MCFV (%)	W/L	Gain [interval]
Box	All	1581	43.07	47.06	126/63	$+3.98\ [2.28, 5.69]$
	Text	977	55.89	61.11	90/39	$+5.22\ [2.97, 7.47]$
	Icon	604	22.35	24.34	36/24	$+1.99\ [-0.50, 4.47]$
Point	All	1581	50.79	52.06	79/59	$+1.27\ [-0.19, 2.72]$
	Text	977	66.84	69.40	61/36	$+2.56\ [0.61, 4.50]$
	Icon	604	24.83	24.01	18/23	$-0.83\ [-2.98, 1.16]$

C.2 Paired view-count comparisons

Section 5.5 and Table 5 report all two-view subsets and the three-view configuration. The comparisons below give the paired uncertainty for each increment, using the same 10,000 source-path bootstrap resamples as the selector controls.

Relative to $\{1/8, 1/4\}$, $\{1/8, 1/2\}$, and $\{1/4, 1/2\}$, the three-view box gains are $+2.53$ $[1.27, 3.80]$, $+1.58$ $[0.57, 2.66]$, and $+3.42$ $[1.96, 4.87]$ points. The point gains are $+4.87$ $[3.48, 6.26]$, $+1.20$ $[0.06, 2.34]$, and $+0.82$ $[-0.25, 1.90]$. The final interval crosses zero. All subsets retain the full benchmark denominator and use the same recorded observations.

C.3 Seeds, application domains, and target size

Table 9 reports the three UI-Zoomer executions. Each uses its own observations, as do Full and Lazy. The paired gains remain positive across seeds. Table 10 and Figure 4 report all six application domains for both complete-method comparisons. With boxes, MCFV exceeds the UI mean in each domain and ties medoid on Scientific. The tie consists of 14 wins and 14 losses. With points, Creative has one fewer correct task than AutoFocus; the other five domains improve.

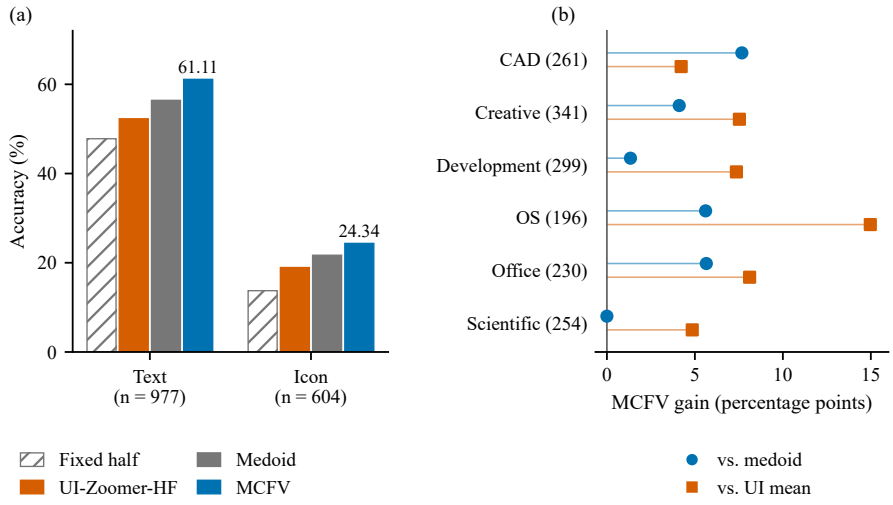


Figure 4: Box-setting results by target type and application domain. (a) Text and icon accuracy. (b) MCFV gains over medoid and the UI-Zoomer seed mean. Parentheses give task counts.

Table 9: UI-Zoomer-HF seed results and paired MCFV gains in the box setting.

Seed	Correct	Accuracy (%)	Requests	Mean (s)	MCFV gain [95% interval]
114514	616	38.96	14,229	13.055	$+8.10$ $[5.76, 10.44]$
114515	633	40.04	14,226	13.064	$+7.02$ $[4.74, 9.30]$
114516	626	39.60	14,225	13.038	$+7.46$ $[5.19, 9.68]$

Table 10: Accuracy (%) by target type and application domain. UI is the three-seed mean.

		Box setting			Point setting	
Group	Tasks	Medoid	UI	MCFV	AutoFocus	MCFV
Text	977	56.40	52.27	61.11	63.97	69.40
Icon	604	21.69	18.93	24.34	20.86	24.01
CAD	261	37.16	40.61	44.83	44.44	54.41
Creative	341	31.67	28.25	35.78	43.11	42.82
Development	299	48.16	42.14	49.50	43.48	46.49
OS	196	43.88	34.52	49.49	41.84	49.49
Office	230	60.00	57.54	65.65	67.83	73.48
Scientific	254	42.91	38.06	42.91	47.24	51.18

The MVP reproduction yields 437 correct text tasks and 79 correct icon tasks, totaling 516. Its reported mean pipeline time is 11.8722 seconds per task. The runner's 6.00 forwards per task use a different counting unit from the autoregressive requests in Table 2. MVP's attention-guided view ranking is part of this complete method and is not available in the shared fixed-view observations used for the selector controls.

Target size. Table 11 tests how the selection gain is distributed across target sizes. We group the same 1,581 tasks by annotated target area as a percentage of screenshot area and retain all four prespecified bins. Paired intervals use 5,000 source-path bootstrap resamples within each bin. MCFV gains 15, 29, and 22 correct answers in the first three bins, with improvements of 5.79, 4.33, and 3.69 percentage points. These groups account for the positive selection gain on small and medium-area targets.

Table 11: MCFV versus medoid by target area in the box setting. W/L counts MCFV-only/medoid-only correct tasks.

Target area	Tasks	MCFV hits	Medoid hits	W/L	Gain [95% interval] (pp)
$< 0.01\%$	259	57	42	19/4	$+5.79$ $[2.32, 9.65]$
0.01–0.05%	670	273	244	60/31	$+4.33$ $[1.49, 7.02]$
0.05–0.20%	597	379	357	51/29	$+3.69$ $[0.67, 6.53]$
$\geq 0.20\%$	55	35	39	5/9	-7.27 $[-20.00, 5.45]$

The largest-area group reverses the trend: MCFV selects 35 correct answers versus medoid's 39 among 55 tasks, a difference of -7.27 points with interval $[-20.00, 5.45]$. Its four lost answers offset part of the 66 gained in the other bins, giving the overall net gain of 62. Target-type composition also varies with area, so this breakdown describes where the observed gains occur rather than establishing that area alone determines which selector is preferable.

C.4 Measured execution cost

Table 12 summarizes the within-run latency distributions. The point Full/Lazy pair and point AutoFocus/Lazy pair are separate measurements. In the latter, AutoFocus and Lazy average 107.261 and 25.714 seconds; their total requests are 21,098 and 5,184. All cost reductions use the corresponding paired run.

Table 12: Independent Full/Lazy serial latency, in seconds per task.

Output	Execution	Mean	Median	95th percentile	Maximum
Box	Full	3.708	3.475	5.000	8.456
	Lazy	3.264	3.232	4.667	7.121
Point	Full	9.989	8.056	17.079	38.882
	Lazy	8.253	7.310	13.722	30.935

Box Full/Lazy total times are 5,862.911 and 5,161.125 seconds; point totals are 15,792.388 and 13,048.489 seconds. The reductions of 11.97% and 17.37% use these unrounded totals. They are not averages of task-wise percentage changes. The box paired time differences range from -0.631 to 4.292 seconds, and point differences from -0.729 to 14.766 seconds. Negative differences are retained. The recorded Full/Lazy mean time is lower for Lazy under each execution-order stratum; these measurements characterize the tested serial implementation.

D Recorded Selection Cases

Figures 5 and 6 show a correction and a regression in each scale direction, using the same local predictions for MCFV and medoid. The examples illustrate mechanisms rather than their population frequencies. Target annotations and enlarged details are for display only. Finer predictions shown alongside a stopping decision come from Full execution; Lazy does not acquire them.

The Outliner example in Figure 1 also chooses a larger view. Its half-view point is $(0.057971, 0.041209)$, inside the annotated target. The quarter and eighth windows start at normalized vertical coordinates 0.062963 and 0.125463, respectively, so neither contains the half-view

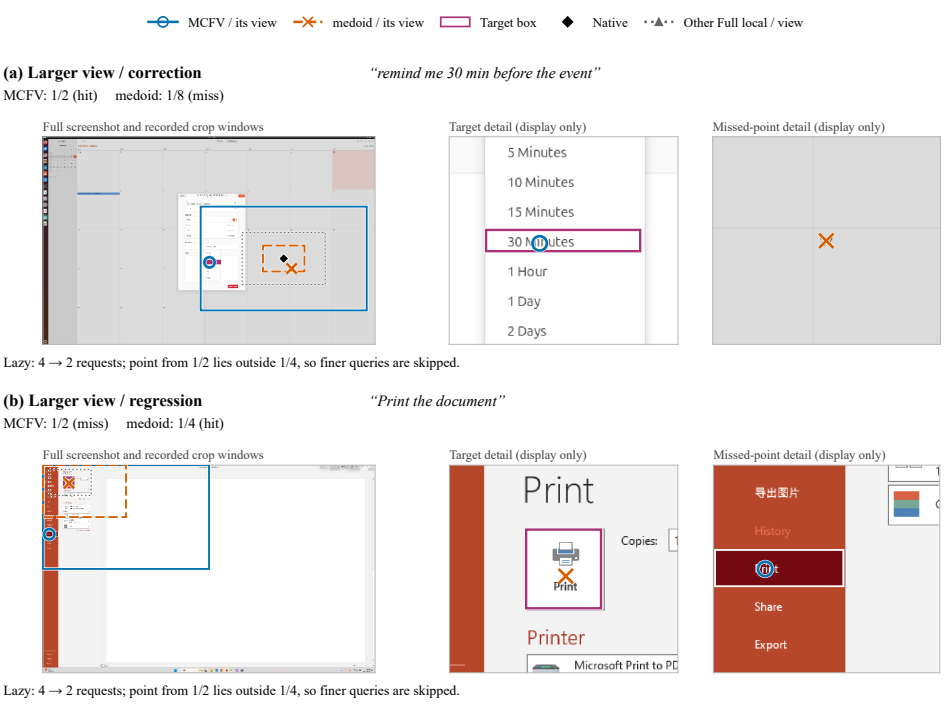


Figure 5: Larger-view selection. The reminder example is corrected by the half-view answer, while the printing example is harmed by a wrong half-view prediction. Both half-view points exclude the finer windows and yield the same exact pruning condition.

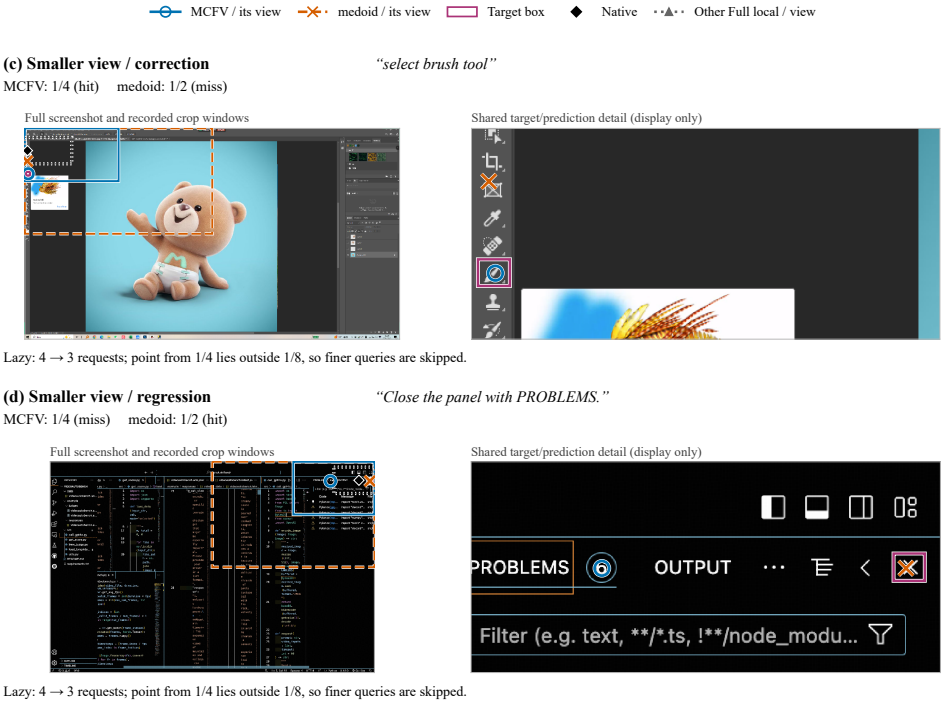


Figure 6: Smaller-view selection. The quarter-view prediction corrects the brush-tool task but selects the wrong control in the panel-closing task. In both cases, it lies outside the eighth window, making that final query unnecessary for MCFV.

point. The two fine predictions lie near another menu item, and medoid selects the eighth-view point. Full uses four requests including Native; Lazy uses Native and the half view only. The supplied example record preserves the exact crop edges and unrounded predictions.

E STOPPING-RULE CONTROLS

E.1 PROTOCOL AND METRICS

Replay protocol. We replay the completed ScreenSpot-Pro predictions for Qwen2.5-VL-7B with box outputs on CPU. Native anchors, crop windows, response-acceptance criteria, box-center mapping, and fallback remain unchanged. Local views follow the order $1/2 \to 1/4 \to 1/8$. The primary comparison applies the MCFV selection rule to each acquired candidate prefix, retaining the original generating-view indices and Native fallback. The replay consumes recorded responses and requires no additional model inference.

Stopping rules. Confidence stopping terminates when an accepted local candidate has a token score strictly greater than τ, where $\tau \in \{0.5, 0.7, 0.9\}$. The score is the geometric mean of the raw generated-token probabilities, including EOS. Agreement stopping terminates when two consecutive queried local views both produce accepted candidates whose Euclidean distance in normalized full-image coordinates is strictly less than $\epsilon = 0.02$. A response without an accepted local candidate interrupts this consecutive-pair test. Native is excluded from the agreement comparison. All prespecified thresholds are reported.

Metrics. All configurations retain all 1,581 tasks, comprising 977 text and 604 icon targets, and the same nine missing final outputs, which are scored as incorrect. Accuracy uses closed-boundary point-in-box evaluation against the annotated target intersected with the image canvas. Coordinate mismatches compare the stored numerical coordinates exactly; scale mismatches compare their generating views. An output mismatch is the union of these events. Corrections and regressions count changes in target correctness relative to Full MCFV. Request counts include Native and every acquired local response, including responses without an accepted candidate. Early stopping is counted only when at least one scheduled local query remains.

E.2 RESULTS WITH PREFIX MCFV

Table 13 reports the complete primary comparison. At $\tau = 0.7$, confidence stopping saves 208 requests but changes 91 coordinate-or-scale outputs, with one correction and five regressions. At $\tau = 0.9$, two tasks terminate early and one output changes; all target-correctness decisions remain the same as Full. Agreement stopping produces 31 corrections and 53 regressions, yielding 22 fewer correct predictions. These counts distinguish changes in the selected answer from changes in target correctness.

Table 13: Stopping-rule replay with prefix MCFV.

Rule	All	Text	Icon	Requests	C/S	$+/-$	Early
Full MCFV	47.06	61.11	24.34	6,297	0/0	0/0	0
Confidence $\tau = 0.5$	41.37	56.50	16.89	4,192	834/835	49/139	1,189
Confidence $\tau = 0.7$	46.81	60.80	24.17	6,089	89/91	1/5	135
Confidence $\tau = 0.9$	47.06	61.11	24.34	6,294	1/1	0/0	2
Agreement $\epsilon = 0.02$	45.67	59.88	22.68	5,555	638/642	31/53	742
Lazy MCFV	47.06	61.11	24.34	5,242	0/0	0/0	691

All, Text, and Icon are accuracies (%). C/S counts coordinate/scale mismatches; $+/-$ counts corrections/regressions relative to Full MCFV. Early counts tasks that omit at least one remaining local query.

Lazy replay matches the recorded independent execution task by task, including its final coordinates, selected scales, 744 correct predictions, and 5,242 requests. The stopping-rule replay measures query counts and decision agreement for the tested Qwen box configuration; runtime results come from the independent Full/Lazy execution reported in the main text.

E.3 Effect of the return rule

The return-current variant returns the last accepted local prediction, falling back to Native when no local candidate is accepted. It uses the same stopping tests and acquisition traces as the corresponding prefix-MCFV control. Table 14 additionally includes return-current without early stopping.

Table 14: Stopping-rule replay with return-current.

Rule	All	Text	Icon	Requests	C/S	$+/-$	Early
Confidence $\tau = 0.5$	41.30	55.68	18.05	4,192	1,103/1,104	70/161	1,189
Confidence $\tau = 0.7$	42.13	54.45	22.19	6,089	739/741	47/125	135
Confidence $\tau = 0.9$	42.38	54.66	22.52	6,294	668/669	47/121	2
Agreement $\epsilon = 0.02$	42.82	55.58	22.19	5,555	1,222/1,227	74/141	742
No early stopping	42.38	54.66	22.52	6,297	668/669	47/121	0

Columns follow Table 13. All mismatch and correction/regression counts use Full MCFV as the reference.

Without early stopping, return-current achieves 42.38% accuracy and differs from Full MCFV on 668 coordinates and 669 selected scales. Under the fixed coarse-to-fine schedule, this variant returns the finest accepted local candidate. The differences therefore arise from the return rule even when all scheduled views are acquired. Keeping prefix MCFV in the primary comparison isolates the effect of the stopping condition.