UniGUI-Bench: A Real-World Diagnostic Benchmark — What Capabilities Do GUI Agents Most Need?

Anonymous ACL submission

Abstract

GUI agents powered by Multimodal Large Language Models (MLLMs) are advancing rapidly, yet most existing benchmarks evaluate only task-level success or failure—leaving a fundamental question unanswered: *what capabilities do GUI agents most need to succeed in real-world settings?* We introduce **UniGUI-Bench**, a benchmark that jointly verifies task-level outcomes and process-level capability diagnosis. It spans 6 device platforms and 23 application scenarios with bilingual support, comprising over 10,000 evaluation items. Its core design links the two evaluation axes: each task failure can be traced to specific capability deficiencies, enabling fine-grained attribution beyond binary pass/fail. Experiments on 14 frontier models reveal three findings: (1) model rankings diverge substantially from single-platform leaderboards under broader evaluation; (2) different scenarios impose distinct capability bottlenecks identifiable through our attribution framework; (3) targeted training on diagnosed bottleneck dimensions yields consistent improvements on both UniGUI-Bench and OSWorld, closing the evaluation–improvement loop.

1 Introduction

Multimodal large language models (MLLMs) have enabled GUI agents to perceive, reason, and act within graphical user interfaces across mobile (Zhang et al., 2025; Wang et al., 2024), desktop (Xie et al., 2024; Bonatti et al., 2024), and web (Zheng et al., 2024; He et al., 2024) platforms (Qin et al., 2025; Hong et al., 2024; Xu et al., 2024; Wang et al., 2025a; Xue et al., 2026). As these agents approach real-world deployment, evaluation must move beyond ranking models toward diagnosing *which specific capabilities* determine practical task success. However, existing benchmarks (Zhou et al., 2023; Xie et al., 2024; Rawles et al., 2024; He et al., 2024; Deng et al., 2023; Bonatti et al., 2024; Lu et al., 2025) fall short of this goal in three ways. **(i) Fragmented platform coverage.** Most benchmarks target a single platform—OSWorld (Xie et al., 2024) for desktops, AndroidWorld (Rawles et al., 2024) for mobile, Mind2Web (Deng et al., 2023) for browsers—missing cross-platform diversity. **(ii) Narrow scenario coverage.** Task designs are constrained by what can be automatically verified rather than what users actually need, yielding a narrow slice of real-world usage. **(iii) No capability-level diagnosis.** Benchmarks measure only task-level success rates (Xie et al., 2024; Rawles et al., 2024; He et al., 2024; Bonatti et al., 2024); when a task fails, there is no way to tell whether the bottleneck lies in element grounding, intent reasoning, or dynamic adaptation. Recent efforts such as MMBench-GUI (Wang et al., 2025c) and GUI-CEval (Li et al., 2026) attempt finer-grained evaluation but assess capabilities *in isolation* from task execution—no existing benchmark connects capability diagnosis with task outcomes to answer *which* deficiency matters most.

To bridge this gap, we introduce **UniGUI-Bench**, a benchmark that jointly verifies task-level outcomes and process-level capability diagnosis (Figure 1). UniGUI-Bench covers 23 application scenarios across 6 device platforms with bilingual support, yielding over 10,000 evaluation items. Its core design is a *dual-axis, linked evaluation*: outcome-level verification judges task completion through step-level action samples, while process-level auditing diagnoses agent capabilities through items organized under a hierarchical taxonomy of 4 primary and 11 secondary dimensions. The two axes share the same data lineage, so that each task failure can be traced to specific capability deficiencies.

We evaluate 14 frontier models and arrive at three key findings. **(1)** Model rankings diverge substantially from single-platform leaderboards under UniGUI-Bench's broader evaluation. **(2)** Different scenarios impose distinct capability bottlenecks:

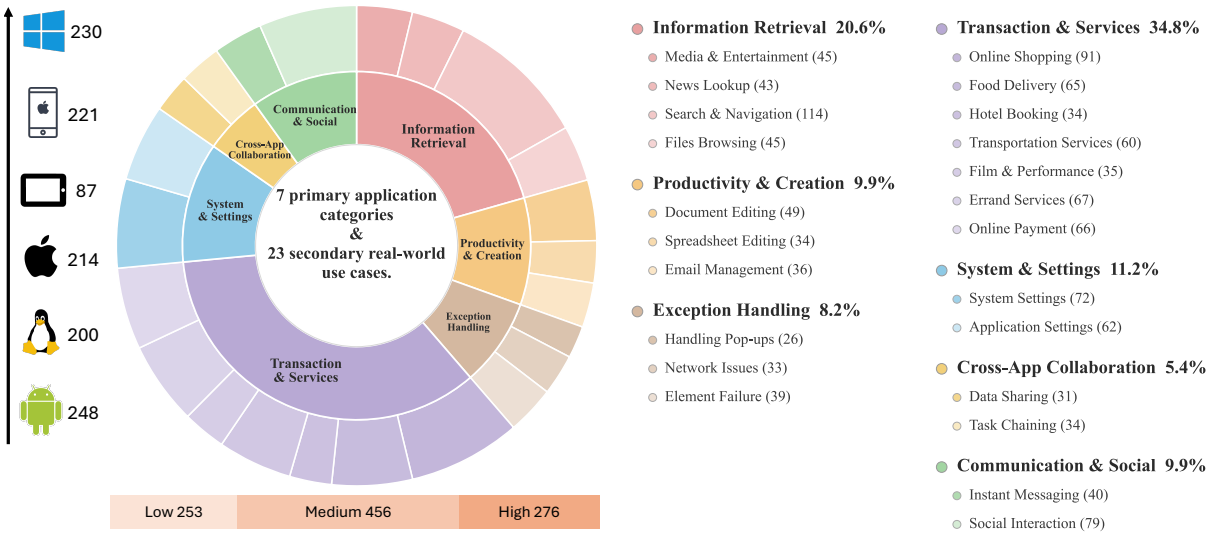

Figure 1: Overview of UniGUI-Bench: UniGUI-Bench comprises over 10,000 evaluation items spanning 6 device platforms, 7 primary application domains with 23 secondary scenarios, and 3 difficulty levels, with bilingual (Chinese & English) support.

Cross-App Collaboration failures stem from weak Task Completion Detection, Exception Handling from insufficient Dynamic Adaptation, and Productivity & Creation from degraded Intent Reasoning. **(3)** Targeted training on diagnosed bottleneck dimensions yields consistent improvements on both UniGUI-Bench and OSWorld, closing the evaluation–improvement loop. Our contributions are:

- **A diverse and realistic benchmark** spanning 23 scenarios across 6 platforms with bilingual support and over 10,000 evaluation items, covering scenarios underrepresented in prior benchmarks such as cross-application collaboration, content creation, and exception handling.
- **Fine-grained capability attribution linking process to outcome.** A dual-axis evaluation framework enables each task failure to be traced to specific capability deficiencies across 4 primary and 11 secondary dimensions, providing actionable, per-scenario prescriptions.
- **Actionable findings with cross-benchmark generalization.** Evaluation of 14 models reveals scenario-specific capability bottlenecks, and targeted training on diagnosed dimensions yields consistent improvements on both UniGUI-Bench and OSWorld.

2 Related Work

GUI Agents. GUI agents can be broadly grouped into three paradigms. Prompting-based methods (Zheng et al., 2024; Zhang et al., 2025; He et al., 2024; Wang et al., 2024) enhance VLM perception through visual prompting (Yang et al., 2023; Lu et al., 2024) without additional training, yet remain bounded by the base model's capabilities. Supervised fine-tuning (Wu et al., 2024; Lin et al., 2025b; Cheng et al., 2024; Qin et al., 2025; Hong et al., 2024; Xu et al., 2024) improves grounding and action prediction but is constrained by annotated data coverage. Reinforcement learning approaches (Lian et al., 2025; Luo et al., 2025; Wang et al., 2025a; Xue et al., 2026) further enable self-improvement through environment feedback. Despite these advances, agents across all paradigms are evaluated using only task-level success/failure signals, without capability-dimension diagnosis to identify which specific deficiencies hinder task completion. **GUI Benchmarks.** Existing benchmarks span web (Gou et al., 2025; Zhou et al., 2023; He et al., 2024; Deng et al., 2023), mobile (Rawles et al., 2024; Lu et al., 2025; Li et al., 2024; Guo et al., 2025b), desktop (Xie et al., 2024; Bonatti et al., 2024), and cross-platform (Dai et al., 2025; Bu et al., 2025) settings. Online benchmarks (Zhou et al., 2023; Xie et al., 2024; Rawles et al., 2024) deploy agents in interactive environments but are constrained by platform limitations

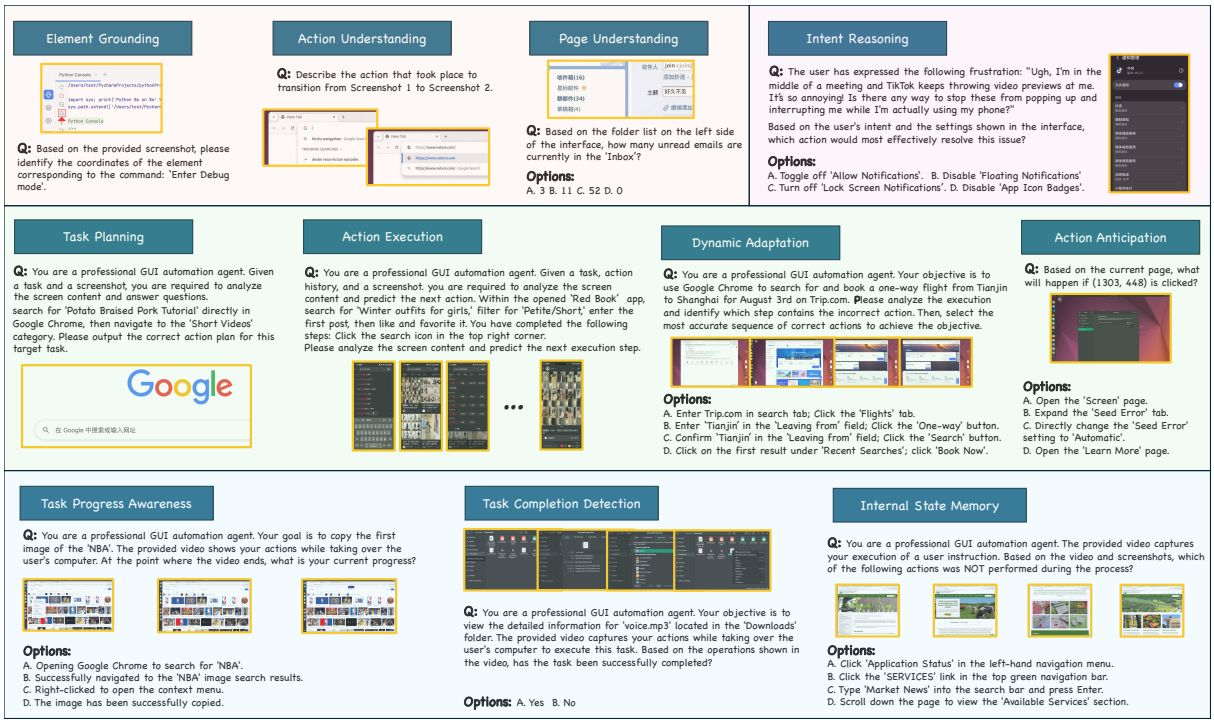

Figure 2: Illustration of the 11 process-level capability dimensions in UniGUI-Bench, organized into four primary categories: *Perception* (Element Grounding, Action Understanding, Page Understanding), *Reasoning* (Task Planning, Action Execution, Dynamic Adaptation, Action Anticipation), *State Management* (Task Progress Awareness, Task Completion Detection, Internal State Memory), and *Semantic Understanding* (Intent Reasoning). Each dimension is accompanied by a representative example showing the input screenshot, the evaluation prompt, and the expected response.

and costly verification (Xue et al., 2025; Lin et al., 2025a); offline benchmarks (Gou et al., 2025; Xue et al., 2025) circumvent these constraints via action matching but similarly lack capability-level diagnosis. Both paradigms remain confined to task-outcome assessment. A complementary line of work evaluates GUI element grounding in isolation (Jurmu et al., 2008; Cheng et al., 2024; Li et al., 2025; Gou et al., 2024; Nayak et al., 2025), without examining how grounding deficiencies propagate to task failures. Recent efforts attempt finer-grained evaluation: MMBench-GUI (Wang et al., 2025c) defines a capability hierarchy but does not couple it with task-level evaluation; GUI-CEval (Li et al., 2026) covers perception, grounding, and planning but is confined to mobile and assesses capabilities in isolation from task execution. No existing benchmark links capability diagnosis with task outcomes. UniGUI-Bench fills this gap by jointly performing outcome- and process-level evaluation with an explicit dependency link, enabling capability attribution that traces task failures to their root causes.

3 UniGUI-Bench

UniGUI-Bench is designed to diagnose GUI agents along two complementary axes: *system-level evaluation*, which measures end-to-end task success, and *process-level evaluation*, which provides fine-grained capability diagnosis. Accordingly, the benchmark comprises two types of data. We first collect human-operated GUI trajectories spanning diverse platforms and scenarios for system evaluation (§3.1). We then derive step-level capability samples from these trajectories for process evaluation (§3.2). This shared data lineage ensures that the two evaluation axes are inherently linked, enabling direct attribution from task-level failures to specific capability deficiencies.

3.1 System Evaluation Data

The system evaluation data consists of high-quality GUI trajectories, each constructed through three stages: task design, trajectory recording, and step-level annotation (Figure 3).

Task Design. We define tasks across 7 primary categories and 23 fine-grained scenarios that reflect everyday user activities (*e.g.*, information re-

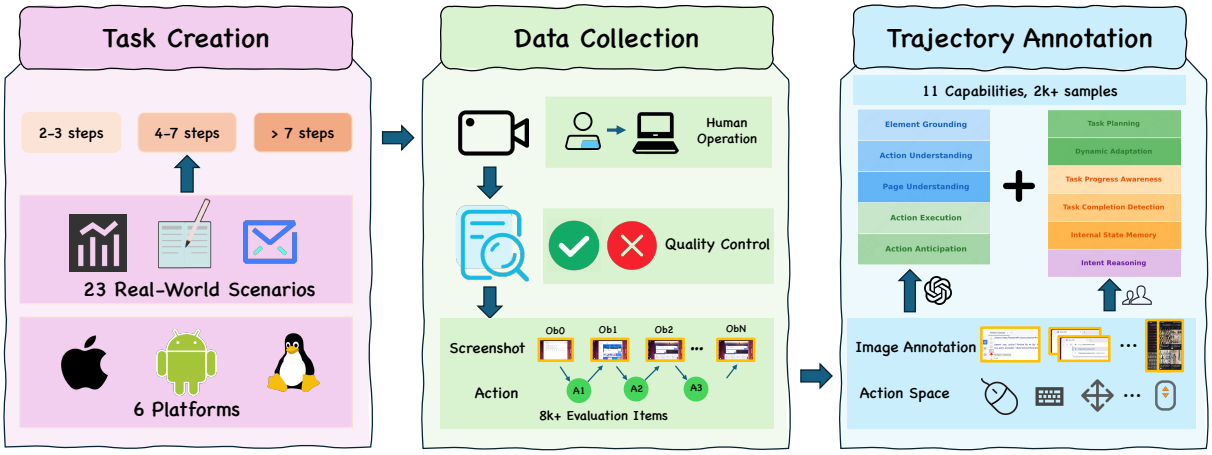

Figure 3: Data collection pipeline of UniGUI-Bench. System evaluation data is constructed through task design, trajectory recording, and step-level annotation; process evaluation data is then derived via automatic or manual capability sample construction, followed by quality control.

trieval, content creation, communication and social networking). Tasks are collected on six operating systems—Windows, macOS, Android, iOS, iPadOS, and Ubuntu—spanning three execution carriers: native applications, web browsers, and OS-level operations. Each task is manually assigned a complexity level based on the number of necessary steps: *low* (2–3 steps), *medium* (4–7 steps), and *high* (>7 steps). Crucially, every task is designed so that exactly one deterministic execution path leads to successful completion, ensuring that step-level evaluation admits a unique ground-truth trajectory and eliminating ambiguity in correctness judgments.

Trajectory Recording. Screen recordings are captured on the designated device, scenario, and carrier, simulating GUI-agent task execution. Each recording must strictly match the specified device, scenario, carrier, and task description, ensuring a unique and reproducible execution path. Additional specifications regarding video format, recording speed, and risk mitigation are provided in the Appendix.

Trajectory Annotation. Each recorded trajectory is decomposed into step-level annotations: (1) *State screenshots*—each screenshot strictly captures the completion state of the preceding action, ensuring visual consistency; (2) *Semantic and API action descriptions*—the next action at each step is described semantically according to a predefined action vocabulary, and the target UI element is annotated with a bounding-box coordinate via manual labeling. The full action vocabulary is detailed in the Appendix.

3.2 Process Evaluation Data

Building upon the annotated trajectories, we construct step-level evaluation samples targeting eleven capability dimensions organized into four categories: Perception, Reasoning, State Management, and Semantic Understanding (Figure 2; detailed definitions in Appendix Table 7). Depending on the nature of each capability, samples are generated either automatically via LLM-assisted pipelines or entirely through manual construction. The detailed construction procedure for each dimension is provided in Appendix B. All automatically generated data undergo manual quality inspection and correction (see Appendix A).

In total, UniGUI-Bench comprises over 10,000 evaluation items: more than 8,000 step-level action samples derived from human-demonstrated trajectories for system evaluation, and over 2,200 capability-targeted items across the eleven dimensions for process evaluation.

4 Experiments

We evaluate 14 mainstream VLMs on UniGUI-Bench to progressively answer four questions: (1) How do current GUI agents perform across diverse real-world platforms and scenarios? (2) What capability profiles do models exhibit across 11 fine-grained dimensions? (3) Which capability deficiencies most critically explain task failures at the model and scenario levels? (4) Does our diagnostic framework generalize beyond UniGUI-Bench? The following subsections address these questions in turn—from system-level performance (§4.2) to process-level capability profiling (§4.3), capability-

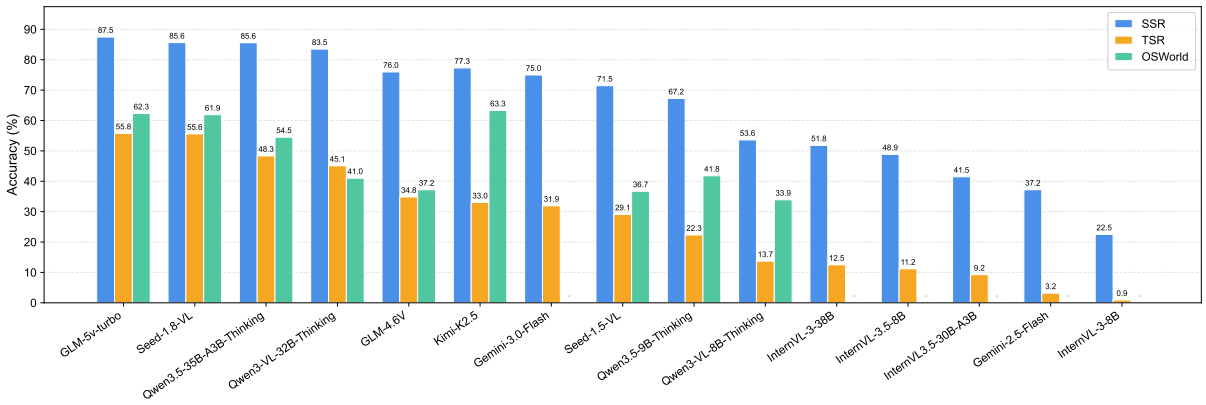

Figure 4: System-level evaluation results across 14 VLMs on UniGUI-Bench and OSWorld benchmarks. Model rankings on UniGUI-Bench diverge from OSWorld, notably for Kimi-K2.5.

to-outcome attribution (§4.4), and cross-benchmark generalization validated through targeted training (§4.5).

4.1 Experimental Setup

Evaluated Models. We benchmark 14 VLMs spanning both proprietary and open-source families: the Seed series (Seed-1.5-VL (Guo et al., 2025a), Seed-1.8-VL (Seed, 2026)), Qwen series (Qwen3-VL-8B/32B-Thinking, Qwen3.5-35B-A3B-Thinking) (Bai et al., 2025), Kimi-K2.5 (Team et al., 2026), GLM series (GLM-4.6V-Flash, GLM-5v-turbo) (Hong et al., 2025), Gemini series (Gemini-2.5-Flash, Gemini-3.0-Flash) (Comanici et al., 2025), and InternVL series (InternVL-3-8B/38B (Zhu et al., 2025), InternVL-3.5-8B/30B-A3B (Wang et al., 2025b)). Models vary in scale from 8B to 35B+ parameters, covering the current spectrum of GUI agent architectures.

Evaluation Metrics. We adopt two hierarchical metrics to evaluate GUI agent performance at different granularities:

- **Step Success Rate (SSR)**: A step is considered successful if and only if both the selected element and the executed action are correct. SSR is the proportion of successful steps over all evaluated steps.
- **Task Success Rate (TSR)**: A task is considered successful if and only if all of its constituent steps are correct. TSR is the proportion of successful tasks over all evaluated tasks.

We additionally report OSWorld (Xie et al., 2024) scores where available, enabling cross-benchmark comparison.

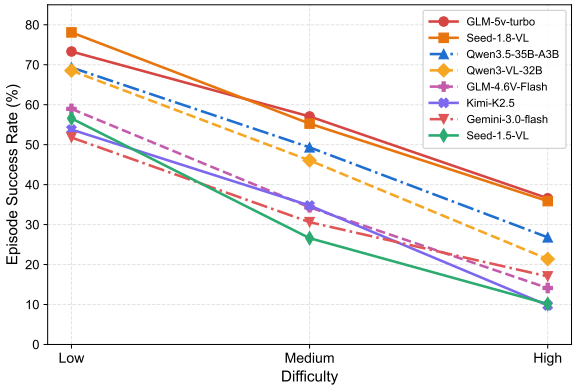

Figure 5: Task success rate (%) by difficulty level. Performance drops sharply from easy to hard tasks across all models.

The detailed evaluation protocol for both system- and process-level evaluation is provided in Appendix C.

4.2 System-Level Evaluation

Figure 4 presents overall SSR and TSR. GLM-5v-turbo leads on both metrics (87.5% SSR, 55.8% TSR), followed by Seed-1.8-VL and Qwen3.5-35B-A3B. Notably, model rankings diverge from OSWorld: Kimi-K2.5, despite leading the OSWorld leaderboard (63.3%), falls behind on UniGUI-Bench, confirming that broader platform coverage can expose gaps hidden by single-platform evaluations. Performance also degrades steeply with task difficulty (Figure 5): VLMs approach 80% TSR on low-difficulty tasks but drop below 40% on high-difficulty ones.

By device platform. Table 1 reveals a consistent mobile–desktop gap, likely because lower mobile resolutions make target elements easier to localize.

Table 1: Task success rate (%) by device platform.

Model	And.	iOS	iPad	Ubu.	Win.	Mac
GLM-5v-turbo	**71.4**	67.4	56.3	**62.5**	**46.1**	29.4
Seed-1.8-VL	70.6	**68.8**	**64.4**	53.0	43.9	**36.0**
Qwen3.5-35B	60.1	57.9	48.3	58.0	33.5	31.8
Qwen3-VL-32B	56.9	57.0	46.0	60.5	27.0	23.8
GLM-4.6V	53.2	50.7	40.2	39.0	16.1	11.2
Kimi-K2.5	35.1	35.3	29.9	54.0	29.1	14.0
Gemini-3.0	45.6	48.9	29.9	37.5	18.7	8.4
Seed-1.5-VL	42.3	39.8	27.6	33.5	16.5	12.6

Among PC platforms, Ubuntu consistently outperforms Windows and macOS, reflecting the larger volume of Ubuntu-based training data.

Table 2: Step success rate (%) by action type.

Model	Click	Type	Scroll	Press	Wait	Oth.
GLM-5v-turbo	**90.6**	85.4	61.2	47.3	45.8	4.8
Seed-1.8-VL	90.1	**93.2**	**68.4**	90.1	**58.3**	2.4
Qwen3.5-35B	88.2	85.8	49.5	76.3	41.7	3.6
Qwen3-VL-32B	85.4	88.7	47.5	82.4	54.2	3.6
GLM-4.6V	79.5	77.1	38.3	16.0	37.5	1.2
Kimi-K2.5	78.9	81.8	65.8	45.8	54.2	10.8
Gemini-3.0	75.2	84.9	58.7	**92.4**	45.8	**12.1**
Seed-1.5-VL	72.6	82.3	39.8	71.8	33.3	2.4

By action type. All models achieve high accuracy on *click* and *type* (above 70%), but *scroll* and *press* exhibit substantial inter-model variance (Table 2), indicating that GUI-specific motor actions remain a differentiating challenge.

By application scenario. Table 3 reveals pronounced variation: Transaction & Services and System & Settings are well-handled (top scores above 64%), whereas Cross-App Collaboration (best: 13.9%), Productivity & Creation (best: 39.5%), and Exception Handling (best: 40.8%) remain around or below 40%, indicating that multi-application coordination, complex creative workflows, and recovery from unexpected states pose the greatest challenge.

The above results characterize *what* happens but do not explain *why*—we address this next.

4.3 Process-Level Capability Profiling

Table 4 reports scores across all 11 dimensions. Three patterns stand out. (1) **No model dominates uniformly**: Seed-1.8-VL leads on average (84.2%) with strengths in State Management dimensions (Internal State Memory 89.5%, Task Progress Awareness 80.5%); Kimi-K2.5 tops Action Anticipation (82.0%) and Dynamic Adaptation (80.5%); GLM-5v-turbo leads Element Grounding (94.0%); Gemini-3.0-Flash leads Intent Reasoning (94.0%). (2) **Page Understanding is nearly saturated among top models** (>96%), while **Task Completion Detection and Task Progress Awareness exhibit the widest cross-model variance**, indicating high sensitivity to training data. (3) **State Management differentiates top-tier models**: Internal State Memory and Task Progress Awareness separate Seed-1.8-VL from its closest competitors, suggesting that coherent state tracking across long trajectories remains a key bottleneck.

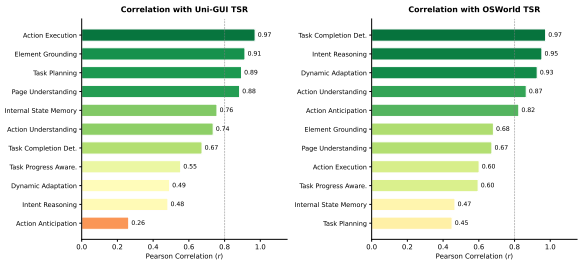

Figure 6: Pearson correlation between each capability dimension and task success rate on UniGUI-Bench (left) and OSWorld (right). The two benchmarks exhibit distinct correlation structures.

These profiles characterize each model's capability landscape but do not reveal which deficiencies most critically determine task success.

4.4 Capability Attribution: From Profiles to Diagnosis

We now link capability profiles to task outcomes—the core diagnostic contribution of UniGUI-Bench.

Capability–Outcome Correlation. UniGUI-Bench task success correlates strongly (r >0.85) with Task Planning, Action Execution, Internal State Memory, and Element Grounding (Figure 6, left), confirming that task planning and state memory remain essential even in an offline, single-step prediction setting. Dimensions such as Action Anticipation and Dynamic Adaptation show weaker correlations, as the single-step paradigm reduces the demand for forward simulation and real-time adaptation.

Scenario-Level Attribution. For the three most challenging scenarios (all below 40% TSR), we compare capability accuracy on failed tasks against overall accuracy to pinpoint which dimensions degrade most sharply (Figure 7). Three distinct patterns emerge: (1) **Cross-App Collaboration** exhibits the largest drop on Task Completion

Table 3: Task completion rates (%) across 7 application scenarios for all evaluated models. **Bold** marks the best score per scenario. Cross-App Collaboration remains the hardest scenario (best: 13.9%), while Transaction & Services and System & Settings see the highest scores.

Model	Cross-App Collab.	Productivity & Creation	Exception Handling	Information Retrieval	Comm. & Social	Transaction & Services	System & Settings
GLM-5v-turbo	10.8	37.0	37.8	**58.3**	**61.3**	66.5	**64.2**
Seed-1.8-VL	**13.9**	**39.5**	**40.8**	57.5	60.5	**67.0**	57.5
Qwen3.5-35B-A3B	9.2	37.0	31.6	49.8	52.1	56.5	58.2
Qwen3-VL-32B	6.2	34.5	23.5	47.4	42.0	54.6	58.2
GLM-4.6V-Flash	3.1	16.0	20.4	38.9	37.0	46.2	32.8
Kimi-K2.5	6.2	18.5	18.4	38.9	34.5	33.5	56.0
Gemini-3.0-Flash	1.5	8.4	27.6	38.9	25.2	43.5	27.6
Seed-1.5-VL	3.1	14.3	22.5	36.8	31.1	33.0	31.3

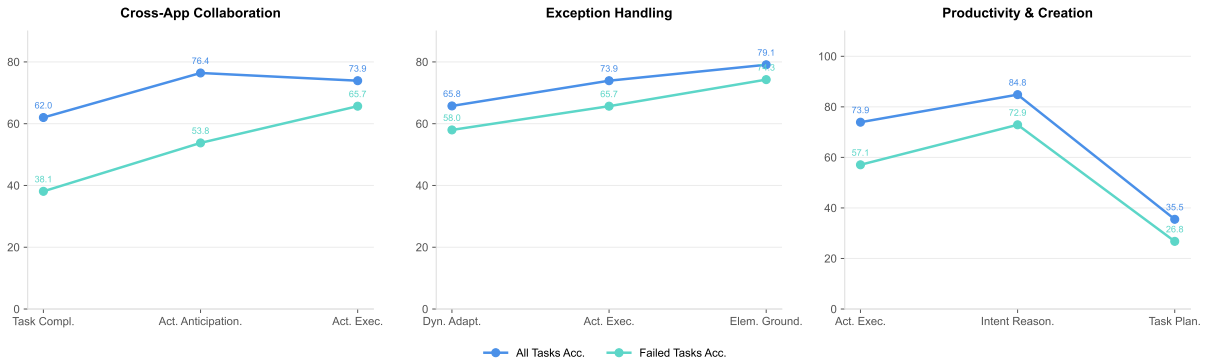

Figure 7: Capability accuracy on failed tasks vs. all tasks for three challenging scenarios. The gap between overall accuracy (blue) and failed-task accuracy (green) pinpoints the capability dimensions whose degradation most directly contributes to each scenario's failures.

Detection—models frequently misjudge whether a sub-task has finished before switching applications; (2) **Exception Handling** declines most on Dynamic Adaptation—models fail to adjust plans when unexpected states arise; (3) **Productivity & Creation** degrades on Intent Reasoning—models struggle to decompose complex creative instructions into correct action sequences. These bottlenecks point to the need for *targeted, scenario-aware training data curation* rather than uniform capability improvement.

4.5 Cross-Benchmark Generalization

We validate whether these capability dimensions capture general competencies through two analyses.

Cross-Benchmark Correlation. Applying the same analysis to OSWorld (Xie et al., 2024) (Figure 6, right) reveals a complementary pattern: task success correlates most strongly with Task Completion Detection, Action Understanding, and Dynamic Adaptation (r >0.9), while Internal State Memory and Task Planning are less discriminative as top models have narrowed the gap on these dimensions. The two benchmarks producing distinct yet meaningful correlation structures confirms that our taxonomy captures transferable competencies.

Targeted Training Validation. We select three dimensions—Element Grounding, Task Completion Detection, and Dynamic Adaptation—and construct dimension-specific training data via GRPO (Zhihong Shao, 2024) or DPO (Rafailov et al., 2023), totaling ∼34k samples (Appendix D). We employ reinforcement learning to maintain the model's general capabilities across diverse evaluation benchmarks. Each dimension is trained independently on a single objective. As shown in Table 5, all three runs yield consistent task-level gains on both UniGUI-Bench and OSWorld. Notably, Dynamic Adaptation training produces the largest TSR improvement (+8.6%) despite a slight capability score decline, confirming that enhanced

Table 4: Process-level capability scores across 11 dimensions for all evaluated models. **Bold** marks the best score per dimension. Page Understanding is nearly saturated (>96% for top models), while Task Completion Detection and Task Progress Awareness exhibit the widest cross-model variance.

Model	Avg	Perception			Reasoning				State Management			Sem.
		Elem. Ground.	Act. Underst.	Page Underst.	Task Plan.	Act. Exec.	Dyn. Adapt.	Act. Antic.	Task Prog.	Task Compl.	Int. State	Intent Reason.
Seed-1.8-VL	**84.2**	88.0	88.0	97.5	**83.7**	84.0	72.5	73.5	**80.5**	**78.0**	**89.5**	91.5
Kimi-K2.5	80.9	85.0	**90.0**	97.0	74.8	75.5	**80.5**	**82.0**	68.0	72.5	72.5	92.0
Qwen3.5-35B-A3B	78.3	88.5	85.5	**99.5**	63.6	82.5	68.5	74.0	55.0	70.5	82.0	92.0
GLM-5v-turbo	78.0	**94.0**	87.5	98.5	68.5	**88.0**	66.5	77.5	45.0	74.5	71.5	86.5
Qwen3-VL-32B	76.0	85.5	80.0	97.5	64.0	79.0	61.0	73.0	65.5	59.0	81.5	89.5
Gemini-3.0-Flash	74.9	91.0	89.5	96.5	69.0	77.0	59.5	79.5	49.5	44.0	74.0	**94.0**
Seed-1.5-VL	71.7	81.5	84.0	95.0	64.6	79.0	42.0	69.0	62.0	59.0	71.5	81.5
GLM-4.6V-Flash	66.3	90.0	76.5	93.0	24.8	74.5	49.5	66.0	61.0	40.5	71.5	82.5
Gemini-2.5-Flash	65.2	44.0	68.5	97.0	65.9	38.5	62.5	75.0	47.0	48.5	81.5	89.0
Qwen3-VL-8B	61.9	55.5	69.5	87.0	39.9	55.0	42.0	67.5	56.0	52.5	72.0	84.0
InternVL-3-38B	49.2	62.0	20.5	85.5	32.6	37.0	27.0	55.5	42.5	51.0	55.0	72.5
InternVL-3.5-8B	43.2	44.0	20.5	79.0	8.8	37.0	27.0	55.5	33.0	43.5	48.0	63.5
InternVL3.5-30B-A3B	42.5	34.0	17.0	79.0	17.2	42.0	18.5	64.0	37.5	46.0	45.5	67.0
InternVL-3-8B	35.6	26.5	4.5	73.5	2.8	27.5	24.5	58.0	34.5	30.5	47.0	62.0

Table 5: Targeted training results (%) on Qwen3-VL-8B. Each row independently adds one training objective to the base model. Colored deltas show the gain over the base.

Configuration	UniGUI-Bench		OSWorld	Cap. (Before→After)
	TSR	SSR		
Base	13.7	53.6	33.9	–
+Elem. Ground. (GRPO)	18.1 ↑4.4	59.3 ↑5.7	37.5 ↑3.6	55.5→**59.5** ↑
+Compl. Det. (DPO)	18.6 ↑4.9	**66.6** ↑13.0	35.5 ↑1.6	30.5→**56.0** ↑
+Dyn. Adapt. (DPO)	**22.3** ↑8.6	63.0 ↑9.4	**36.4** ↑2.5	42.0→40.5 ↓

adaptability benefits overall task success even when the dimension-specific metric does not fully capture it.

Both analyses converge on the same conclusion: the capability dimensions are not benchmark-specific artifacts but general competencies that transfer across settings, validating UniGUI-Bench as a diagnostic tool that not only identifies *what* fails but also guides *how* to improve.

5 Conclusion

We present UniGUI-Bench, a diagnostic benchmark for GUI agents that pairs outcome-level verification with step-level process auditing across 6 platforms, 23 scenarios, and a hierarchical taxonomy of 4 primary and 11 secondary capability dimensions. Evaluation of 14 VLMs reveals that model rankings diverge substantially from single-platform leaderboards and that different scenarios impose distinct capability bottlenecks identifiable through capability attribution. Targeted training on diagnosed dimensions yields consistent improvements on both UniGUI-Bench and OSWorld, closing the evaluation–improvement loop. UniGUI-Bench thus serves as both a comprehensive diagnostic tool and actionable guidance for building more capable GUI agents.

Limitations

UniGUI-Bench adopts an offline evaluation paradigm: models predict the next action from a fixed screenshot–action history rather than interacting with a live environment. This design choice is deliberate—offline evaluation decouples annotation from execution, enabling far greater task diversity across 6 platforms, 23 scenarios, and bilingual settings than online benchmarks can practically achieve, since the latter are constrained by the availability of deterministic verification scripts that must be written per-task. However, offline evaluation does not capture error-recovery dynamics or compounding effects of earlier mistakes on later steps, which are important aspects of real-world agent deployment. Extending the diagnostic framework to an online setting while preserving comparable task diversity remains an important direction for future work.

References

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, and 1 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Rogerio Bonatti, Dan Zhao, Francesco Bonacci, Dillon Dupont, Sara Abdali, Yinheng Li, Yadong Lu, Justin Wagle, Kazuhito Koishida, Arthur Bucker, and 1 others. 2024. Windows agent arena: Evaluating multi-modal os agents at scale. *arXiv preprint arXiv:2409.08264*.

Wendong Bu, Yang Wu, Qifan Yu, Minghe Gao, Bingchen Miao, Zhenkui Zhang, Kaihang Pan, Yunfei Li, Mengze Li, Wei Ji, and 1 others. 2025. What limits virtual agent application? omnibench: A scalable multi-dimensional benchmark for essential virtual agent capabilities. *arXiv preprint arXiv:2506.08933*.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. 2024. 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)*, pages 9313–9332.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Yutong Dai, Krithika Ramakrishnan, Jing Gu, Matthew Fernandez, Yanqi Luo, Viraj Prabhu, Zhenyu Hu, Silvio Savarese, Caiming Xiong, Zeyuan Chen, and 1 others. 2025. Scuba: Salesforce computer use benchmark. *arXiv preprint arXiv:2509.26506*.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114.

Aarash Feizi, Shravan Nayak, Xiangru Jian, Kevin Qinghong Lin, Kaixin Li, Rabiul Awal, Xing Han Lù, Johan Obando-Ceron, Juan A. Rodríguez, Nicolas Chapados, David Vázquez, Adriana Romero-Soriano, Reihaneh Rabbany, Perouz Taslakian, Christopher Pal, Spandana Gella, and Sai Rajeswar. 2025. Grounding computer use agents on human demonstrations. *CoRR*, abs/2511.07332.

Boyu Gou, Zanming Huang, Yuting Ning, Yu Gu, Michael Lin, Weijian Qi, Andrei Kopanev, Botao Yu, Bernal Jiménez Gutiérrez, Yiheng Shu, and 1 others. 2025. Mind2web 2: Evaluating agentic search with agent-as-a-judge. *arXiv preprint arXiv:2506.21506*.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. 2024. Navigating the digital world as humans do: Universal visual grounding for gui agents. *arXiv preprint arXiv:2410.05243*.

Dong Guo, Faming Wu, Feida Zhu, Fuxing Leng, Guang Shi, Haobin Chen, Haoqi Fan, Jian Wang, Jianyu Jiang, Jiawei Wang, and 1 others. 2025a. Seed1. 5-vl technical report. *arXiv preprint arXiv:2505.07062*.

Yuan Guo, Tingjia Miao, Zheng Wu, Pengzhou Cheng, Ming Zhou, and Zhuosheng Zhang. 2025b. Atomic-to-compositional generalization for mobile agents with a new benchmark and scheduling system. *arXiv preprint arXiv:2506.08972*.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. Webvoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6864–6890.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and 1 others. 2024. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14281–14290.

Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, and 1 others. 2025. Glm-4.5 v and glm-4.1 v-thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*.

Marko Jurmu, Sebastian Boring, and Jukka Riekki. 2008. Screenspot: multidimensional resource discovery for distributed applications in smart spaces. In *MobiQuitous*.

Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. 2025. Screenspot-pro: Gui grounding for professional high-resolution computer use. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 8778–8786.

Wei Li, William Bishop, Alice Li, Chris Rawles, Folawiyo Campbell-Ajala, Divya Tyamagundlu, and Oriana Riva. 2024. On the effects of data scale on ui control agents. *Advances in Neural Information Processing Systems*, 37:92130–92154.

Yang Li, Yuchen Liu, Haoyu Lu, Zhiqiang Xia, Hongzhen Wang, Kaiyang Han, Changpeng Yang, Jinyang Wu, Jiaming Xu, Runyu Shi, and 1 others. 2026. Gui-ceval: A hierarchical and comprehensive chinese benchmark for mobile gui agents. *arXiv preprint arXiv:2603.15039*.

Shuquan Lian, Yuhang Wu, Jia Ma, Yifan Ding, Zihan Song, Bingqi Chen, Xiawu Zheng, and Hui Li. 2025. Ui-agile: Advancing gui agents with effective reinforcement learning and precise inference-time grounding. *arXiv preprint arXiv:2507.22025*.

Haojia Lin, Xiaoyu Tan, Yulei Qin, Zihan Xu, Yuchen Shi, Zongyi Li, Gang Li, Shaofei Cai, Siqi Cai, Chaoyou Fu, and 1 others. 2025a. Cuarewardbench: A benchmark for evaluating reward models on computer-using agent. *arXiv preprint arXiv:2510.18596*.

Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Stan Weixian Lei, Lijuan Wang, and Mike Zheng Shou. 2025b. Showui: One vision-language-action model for gui visual agent. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 19498–19508.

Zhaoyang Liu, JingJing Xie, Zichen Ding, Zehao Li, Bowen Yang, Zhenyu Wu, Xuehui Wang, Qiushi Sun, Shi Liu, Weiyun Wang, Shenglong Ye, Qingyun Li, Xuan Dong, Yue Yu, Chenyu Lu, YunXiang Mo, Yao Yan, Zeyue Tian, Xiao Zhang, and 11 others. 2025. Scalecua: Scaling open-source computer use agents with cross-platform data. *CoRR*, abs/2509.15221.

Quanfeng Lu, Wenqi Shao, Zitao Liu, Lingxiao Du, Fanqing Meng, Boxuan Li, Botong Chen, Siyuan Huang, Kaipeng Zhang, and Ping Luo. 2025. Guiodyssey: A comprehensive dataset for cross-app gui navigation on mobile devices. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22404–22414.

Yadong Lu, Jianwei Yang, Yelong Shen, and Ahmed Awadallah. 2024. Omniparser for pure vision based gui agent. *arXiv preprint arXiv:2408.00203*.

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

Shravan Nayak, Xiangru Jian, Kevin Qinghong Lin, Juan A Rodriguez, Montek Kalsi, Rabiul Awal, Nicolas Chapados, M Tamer Özsu, Aishwarya Agrawal, David Vazquez, and 1 others. 2025. Ui-vision: A desktop-centric gui benchmark for visual perception and interaction. *arXiv preprint arXiv:2503.15661*.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, and 1 others. 2024. Androidworld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*.

Bytedance Seed. 2026. Seed1. 8 model card: Towards generalized real-world agency. *arXiv preprint arXiv:2603.20633*.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, and 1 others. 2026. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*.

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

Junyang Wang, Haiyang Xu, Jiabo Ye, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *arXiv preprint arXiv:2401.16158*.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, and 1 others. 2025b. Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*.

Xuehui Wang, Zhenyu Wu, JingJing Xie, Zichen Ding, Bowen Yang, Zehao Li, Zhaoyang Liu, Qingyun Li, Xuan Dong, Zhe Chen, and 1 others. 2025c. Mmbench-gui: Hierarchical multi-platform evaluation framework for gui agents. *arXiv preprint arXiv:2507.19478*.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and 1 others. 2024. Os-atlas: A foundation action model for generalist gui agents. *arXiv preprint arXiv:2410.23218*.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, and 1 others. 2024. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094.

Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu, and Caiming Xiong. 2024. Aguvis: Unified pure vision agents for autonomous gui interaction. *arXiv preprint arXiv:2412.04454*.

Taofeng Xue, Chong Peng, Mianqiu Huang, Linsen Guo, Tiancheng Han, Haozhe Wang, Jianing Wang,

Xiaocheng Zhang, Xin Yang, Dengchang Zhao, and 1 others. 2026. Evocua: Evolving computer use agents via learning from scalable synthetic experience. *arXiv preprint arXiv:2601.15876*.

Tianci Xue, Weijian Qi, Tianneng Shi, Chan Hee Song, Boyu Gou, Dawn Song, Huan Sun, and Yu Su. 2025. An illusion of progress? assessing the current state of web agents. *arXiv preprint arXiv:2504.01382*.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. 2023. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *arXiv preprint arXiv:2310.11441*.

Chi Zhang, Zhao Yang, Jiaxuan Liu, Yanda Li, Yucheng Han, Xin Chen, Zebiao Huang, Bin Fu, and Gang Yu. 2025. Appagent: Multimodal agents as smartphone users. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–20.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*.

Qihao Zhu Runxin Xu Junxiao Song Mingchuan Zhang Y.K. Li Y. Wu Daya Guo Zhihong Shao, Peiyi Wang. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, and 1 others. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, and 1 others. 2025. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*.

A Quality Control

To ensure high data quality, we implement quality control measures for both manually constructed and automatically generated data.

Manual Cross-Checking. Independent annotators cross-check the questions and answers for accuracy and unambiguity, and provide suggestions for correction.

Expert Inspection. For tasks automatically constructed using large language models, data experts conduct manual quality inspection to ensure data quality. Specifically, annotators examine each screenshot and its associated question to determine (1) whether the question is answerable given the screenshot, and (2) whether the answer is unique. Samples that fail the inspection are either revised by experts or replaced with newly constructed data. The inspection results are reported in Table 6.

Table 6: Quality inspection accuracy by task type.

Task Type	Accuracy (%)
Element Grounding	83.0
Action Understanding	94.9
GUI Understanding	83.9
Action Prediction	74.8

B Capability Dimension Construction

This section details the construction procedure for each of the eleven process-level capability dimensions (Table 7).

B.1 Perception

Element Grounding. This task evaluates the ability to identify and localize a target UI element within a screenshot, *i.e.*, predicting its bounding-box coordinates. We randomly sample *click* steps from the collected trajectories, extract the corresponding screenshots and bounding boxes, and prompt an LLM to rephrase each direct instruction into an indirect yet unambiguous form (*e.g.*, "*Click [Recently Played]*" → "*Check the songs you have listened to recently*"), requiring semantic inference rather than surface-level label matching. All samples undergo manual review to discard ambiguous references and overly trivial targets.

Action Understanding. This task evaluates the ability to recognize executed actions from visual state transitions. We randomly sample an action step and present its preceding and succeeding screenshots, asking what operation was performed between the two frames. Samples with a visible mouse cursor overlapping the target region are discarded to prevent trivial shortcuts.

Page Understanding. This task assesses the ability to parse GUI structure, identify interface components, and reason about their functions. We randomly sample screenshots and prompt an LLM to generate multiple-choice questions at three difficulty levels in equal proportion: *Easy* questions target direct recognition of visible labels, icons, or prominent UI elements; *Medium* questions require contextual understanding of UI states, hierarchical relations, or navigation logic; *Hard* questions involve numerical calculation, layout analysis, or multi-step reasoning (*e.g.*, summing mail counts across multiple folders). All generated questions undergo manual verification to ensure answerability from the given screenshot and correctness.

Table 7: Eleven process-level capability dimensions in UniGUI-Bench.

Category	Capability	Format	Description	Construction
Perception	Element Grounding	Grounding	Identify and localize a target UI element by predicting its bounding-box coordinates. Direct instructions are rephrased into indirect forms to require semantic inference.	Auto
	Action Understanding	Open-ended	Recognize the executed action between two consecutive screenshots (before and after states).	Auto
	Page Understanding	MCQ	Parse GUI structure and reason about component functions, with questions at three difficulty levels (Easy / Medium / Hard).	Auto
Reasoning	Task Planning	Open-ended	Generate a complete step-by-step plan given a task description, application knowledge (video), and the initial state.	Manual
	Action Execution	Open-ended	Predict the precise next low-level action given the current screenshot, action history, and task goal.	Auto
	Dynamic Adaptation	MCQ	Detect execution errors or goal mismatches and suggest corrective actions from erroneous or misaligned trajectories.	Manual
	Action Anticipation	MCQ	Predict the expected interface change after a given click action is performed.	Auto
State Management	Task Progress Awareness	MCQ	Judge task completion progress from trajectory slices; difficulty is increased by modifying sub-goals.	Manual
	Task Completion Detection	MCQ	Determine whether a task has been fulfilled, with negatives created by altering goals or truncating trajectories.	Manual
	Internal State Memory	MCQ	Trace back the preceding operation path that led to the current state from a mid-to-late trajectory point.	Manual
Semantic Understanding	Intent Reasoning	MCQ	Infer the user's true intention behind ambiguous or incomplete instructions to select the correct action.	Manual

B.2 Reasoning

Task Planning. This task evaluates high-level semantic action prediction and planning capability through open-ended questions, all manually constructed. We provide a task description together with a video demonstrating the cascading relationships among different tabs and options within the target application, serving as prior knowledge. The agent is then given an initial-state screenshot (or a textual description of the starting state) and must output a complete step-by-step plan to accomplish the task.

Action Execution. This task evaluates low-level executable action prediction. Given the current screenshot, the history of semantic actions, and the task goal, the agent must predict the precise next action to execute. Samples are randomly drawn from trajectory steps.

Dynamic Adaptation. This task simulates model–environment interaction scenarios to assess adaptability to unexpected changes and failure recovery, using manually constructed MCQ questions. We employ two strategies: (1) leveraging trajectories that were intentionally recorded with erroneous steps—we clip the trajectory at the error point and ask the agent to identify the mistake and suggest corrective actions; (2) pairing normal (error-free) trajectories with modified task descriptions that are inconsistent with the executed path or goal, and asking the agent to detect the mismatch and determine the correct course of action.

Action Anticipation. This task assesses the ability to anticipate the outcome of a given action, testing forward reasoning over UI state transitions. We randomly sample a *click* step, provide the screenshot along with the click region, and prompt an LLM to generate a description of the expected interface change as well as three distractor options.

B.3 State Management

Task Progress Awareness. This task evaluates the ability to track task completion progress in real time—distinguishing completed steps from pending ones. All samples are manually constructed. We clip trajectory slices from annotated trajectories and ask the agent to judge the current completion status against the original task description. To increase difficulty, we further modify the task description by adding, removing, or altering sub-goals, so that the agent must reason about partial progress under shifted objectives.

Task Completion Detection. This task evaluates whether the agent can judge if a given task has been fulfilled. We construct positive (completed) samples directly from original task descriptions

paired with their full trajectories. Negative (incomplete) samples are created via two strategies: (1) modifying the task requirements by adding, removing, or altering sub-goals while keeping the original trajectory, and (2) truncating the trajectory while keeping the original task description unchanged.

Internal State Memory. This task evaluates the ability to maintain historical information across time steps during multi-step task execution. From the original trajectory, we select a state in the mid-to-late stage as the query point and ask the agent to trace back the preceding operation path or entry point that led to the current state, presented as MCQ questions.

B.4 Semantic Understanding

Intent Reasoning. This task evaluates the ability to infer the true intention behind ambiguous, implicit, or incomplete user instructions. For a given step or state in the original trajectory, we manually construct an instruction that does not explicitly reveal the execution intent, and the agent must reason about the user's underlying goal to select the correct next action, presented as MCQ questions.

C Evaluation Protocol and Action Space

Evaluation Protocol. For system-level evaluation, models receive a task instruction, action history, and the current screenshot, and are required to predict the next action. Step correctness is judged by matching both the action type and the target element against human annotations. For process-level evaluation, models answer targeted capability questions derived from the annotated trajectories, with accuracy computed per dimension.

Table 8: Desktop action space (Windows, macOS, Ubuntu).

Action	Description
`click [x,y]`	Left-click at coordinate $[x, y]$.
`doubleClick [x,y]`	Double left-click at coordinate $[x, y]$.
`tripleClick [x,y]`	Triple left-click at coordinate $[x, y]$.
`rightClick [x,y]`	Right-click at coordinate $[x, y]$.
`hover [x,y]`	Move cursor to coordinate $[x, y]$.
`press [key(s)]`	Press a key or key combination.
`type [text]`	Type the given text string.
`scroll [dir.]`	Scroll in the given direction.
`drag [x1,y1,x2,y2]`	Drag from $[x_1, y_1]$ to $[x_2, y_2]$.
`wait [seconds]`	Wait for the specified duration.

Action Space. We define two platform-specific action spaces for desktop (Windows, macOS, Ubuntu) and mobile (Android, iOS, iPadOS) devices, respectively. During evaluation, the model receives a task instruction, action history, and a screenshot at resolution $W \times H$, and is required to predict the next action.

Table 9: Mobile action space (Android, iOS, iPadOS).

Action	Description
`click [x,y]`	Tap at coordinate $[x, y]$.
`longPress [x,y]`	Long press at coordinate $[x, y]$.
`type [text]`	Type the given text string.
`scroll [dir.]`	Swipe in the given direction.
`wait [seconds]`	Wait for the specified duration.

For both action spaces, the model is instructed to first analyze the screen content and then output the predicted action enclosed in `<actions>...</actions>` tags.

D Targeted Training Data Construction

We construct dimension-specific training data for the four selected capability dimensions as follows:

- **Element Grounding** (15k samples, GRPO (Zhihong Shao, 2024)): These samples are constructed based on Ground-CUA (Feizi et al., 2025), leveraging its precise bounding box information for target elements. We define accuracy and format rewards, where the accuracy reward requires the model's predicted coordinate to fall within the ground-truth bounding box. GRPO is chosen over DPO because grounding tasks have unambiguous correctness criteria.
- **Task Completion Detection** (13k samples, DPO (Rafailov et al., 2023)): These samples are developed by adapting trajectories from ScaleCUA (Liu et al., 2025). Two types of contrastive pairs are constructed. For positive termination cases, the base model rolls out a non-terminate response as the rejected sample. For negative termination cases, we split the trajectory at the 3/4 point and use the premature terminate action as the rejected response.
- **Dynamic Adaptation** (5.6k samples, DPO): Also derived from ScaleCUA trajectories, contrastive pairs are constructed by cross-splicing the last four steps of two different task trajectories. Seed-1.8-VL predicts the correct next step as the chosen response, while the cross-spliced step serves as the rejected response.