ENGINEERING Information Technology & Electronic Engineering
https://www.academax.com/EITEE; https://www.engineering.org.cn
https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=11344065
E-mail: eng.itee@zju.edu.cn

Review

https://doi.org/10.1631/ENG.ITEE.2026.0103

A platform-aware survey of execution, verification, and risk of graphical user interface agents

Yu WU[1], Yi YANG[2]✉

[1]*School of Artificial Intelligence, Wuhan University, Wuhan 430072, China*

[2]*ReLER, Collaborative Innovation Center of Artificial Intelligence, Zhejiang University, Hangzhou 310027, China*

Abstract: Graphical user interface (GUI) agents are widely used in general-purpose digital automation. Large language models, vision-language models, and multimodal foundation models can now interpret screenshots, follow natural-language instructions, and execute grounded actions in websites, mobile applications, and desktop operating systems. However, despite their different execution conditions, these settings are commonly grouped under one label. This survey reviews representative frameworks, models, datasets, and benchmarks of GUI agents through a common technical stack: observation, grounding, planning, memory, execution, and verification. We advocate the treatment of web, mobile, and desktop/operating-system agents as distinct operational regimes with different observabilities, action semantics, hidden-state dependences, execution costs, and side-effect risks. We also describe a shift from next-action prediction to dependable execution with increased focus on outcome verification, execution efficiency, and trustworthiness. The survey concludes with future directions in belief-state tracking, adaptive sensing, hybrid GUI–tool execution, and human oversight for reliable computer use.

Key words: Graphical user interface (GUI) agents; Operational regimes; Trustworthy evaluation; Postcondition verification

1 Introduction

Graphical user interface (GUI) agents are key components of general-purpose digital automation. New-generation large language models (LLMs), multimodal models, and vision-language models allow agents to interpret screenshots, follow natural-language instructions, and act in software environments that previously required scripts or engineered automation (Vaswani et al., 2017; Brown et al., 2020; Bommasani et al., 2021). Recent surveys have covered the foundation models, agent architectures, benchmarks, applications, and trustworthiness (Wang S et al., 2024; Hu et al., 2025; Nguyen D et al., 2025; Sager et al., 2026). Research and products have moved from static question answering and conversational interfaces to systems that can operate across websites, mobile applications, and desktop software (Park JS et al., 2023; Schick et al., 2023; Shinn et al., 2023; Yao et al., 2023). As most digital services and productivity tools are still accessed through GUIs, GUI agents are direct test cases of automation in everyday software.

Nevertheless, progress has been uneven. Web-oriented resources have introduced realistic browser tasks, dynamic websites, and visually grounded web interaction settings (Deng et al., 2023; Drouin et al., 2024; Koh et al., 2024; Pan YC et al., 2024; Zhou SY et al., 2024; de Chezelles et al., 2025). Mobile research has contributed large-scale demonstrations, Android emulators, and app-centric task environments (Rawles et al., 2023, 2025; Wang LY et al., 2024; Chai et al., 2025). Evaluations of desktop and operating-system (OS) benchmarks have expanded to multiapplication workflows, file system states, and general computer-use tasks (Styles et al., 2024; Xie TB et al., 2024b; Bonatti et al., 2025). Recent work at the model level has improved screen understanding, user-interface (UI) grounding, and action predictions for graphical interfaces such as CogAgent (Hong et al., 2024), SeeClick (Cheng KZ et al., 2024), ScreenAI (Baechler et al., 2024), ShowUI (Lin KQ et al., 2025), Ferret-UI 2 (Li ZH et al., 2025), ScreenSpot-Pro (Li KX et al., 2025), Aria-UI (Yang YH et al., 2025), R-VLM (Park J et al., 2025), and X-WebAgentBench (Wang P et al., 2025). Despite this progress, these environments are usually treated as variations of a single interaction problem with next-step prediction as the primary challenge. Such framing is no longer

✉ yangyics@zju.edu.cn
Yu WU, https://orcid.org/0000-0002-1680-8253
Yi YANG, https://orcid.org/0000-0002-0512-880X
CLC number: TP391.4

Received: Apr. 13, 2026; Revision accepted: June 6, 2026;
Crosschecked: June 17, 2026; Published online: Aug. 3, 2026

sufficient.

Besides their different visual presentations, browser pages, mobile apps, and multiwindow desktop workflows expose different portions of the underlying system state, encode interaction semantics into distinct action primitives, and impose different costs on error recovery. The visible page in web-based tasks often underspecifies the session state, server-side validation, access controls, quotas, and asynchronous updates. Mobile settings provide locally meaningful gestures that are entangled with device state, notifications, permissions, and personalization. Correctness in desktop and OS contexts may depend on the hidden file state, window focus, user identity, external tools, and costly or irreversible side effects. These differences indicate that GUI agents operate in distinct operational regimes, not as instances of a single uniform interface-control problem.

In this survey, we organize the literature into a common technical stack spanning observation, grounding, planning, memory, execution, and verification. The stack provides a basis for comparing systems, datasets, benchmarks, and evaluation practices across platforms while preserving the differences in observability, action semantics, hidden-state dependence, execution cost, and side-effect risk among the platforms. Under this framing, an agent must not merely predict the next click, tap, or keystroke, but must sustain the task state, decide the target for observation, verify the success or failure of outcomes, and recognize when uncertainty, cost, or irreversibility demands recovery, tool use, or human oversight.

Recent surveys have covered the benchmarks, foundation model architectures, training methods, applications, trustworthiness, and computer-use taxonomies of GUI agents (Wang S et al., 2024; Hu et al., 2025; Nguyen D et al., 2025; Shi YC et al., 2025; Zhang CY et al., 2025a; Sager et al., 2026). Table 1 compares the present review with representative recent surveys in terms of platform scope, main analytical lens, verification/risk, deployment emphasis, and positioning.

Our literature review is not structured into main categories such as benchmarks, model components, training paradigms, and application domains. Using a common technical stack with platform difference as an analytical variable, we instead examine where uncertainty, failure, and recovery are concentrated across environments. We connect this platform-aware framing to trustworthiness evaluation, deployment constraints, and industrial trajectories. Benchmark scores, architectural modules, and training recipes are only partially informative without reference to the hidden state and evidence of actual task completion.

We cover recent GUI agent systems, datasets, benchmarks, models, and evaluation protocols through this platform-aware lens. We argue that the correctness of web, mobile, and desktop/OS agents faces different challenges shaped by observability, action semantics, hidden-state dependence, and execution risk. We also identify priorities—postcondition verification, trustworthy evaluation, execution efficiency, adaptive sensing, and hybrid GUI–tool interactions—beyond local action prediction. The review is structured as shown in Fig. 1.

Our coverage focuses on the large-model era of GUI agents from 2023 onward, with selective inclusion of earlier precursors that have remained conceptually influential. Structured searches are performed in Google Scholar, arXiv, ACM Digital Library, IEEE Xplore, ACL Anthology, OpenReview, and proceedings or websites of major machine learning, computer vision, human–computer interactions, and software engineering venues. The search was last updated on May 10, 2026. Search terms include combinations of "GUI agent," "UI agent," "computer use agent," "web agent," "browser agent," "mobile agent," "Android agent," "OS agent," "desktop agent," "screen grounding," "visual grounding," "action

Table 1 Comparison with recent surveys on GUI agents and computer-use agents

Survey	Scope	Main analytical lens	Verification/Risk	Deployment	Positioning
Nguyen D et al. (2025)	Broad GUI agents	Components, datasets, evaluation, and benchmark reliability	Evaluation and reliability	Limited	Separating web, mobile, and desktop/OS regimes
Wang S et al. (2024)	Foundation-model GUI agents	Model capability, training, and data	Reliability and evaluation	Research-oriented	Emphasizing platform-specific observability, action semantics, and execution risk
Zhang CY et al. (2025a)	LLM-based GUI agents	Architecture, reasoning, and planning	Failures and evaluation	System-oriented	Adding platform-aware failure and verification analysis
Hu et al. (2025)	Trustworthy GUI agents	Safety, robustness, and evaluation	Main focus	Trustworthy deployment	Connecting risk to hidden state, side effects, and recovery cost
Sager et al. (2026)	Computer-use agents	Capabilities and challenges	Dependable behavior	Broad computer use	Providing finer web/mobile/desktop operational comparison
Shi YC et al. (2025)	OS and device-use agents	MLLM-based OS agents	Reliability and safety	Computing devices	Comparing desktop/OS with web and mobile under one stack
This review	Web, mobile, and desktop/OS	Common stack plus operational regimes	Hidden state, postcondition, and side effects	Benchmarks plus deployment	Using platform differences as an explanatory variable

LLM: large language model; MLLM: multimodal large language model

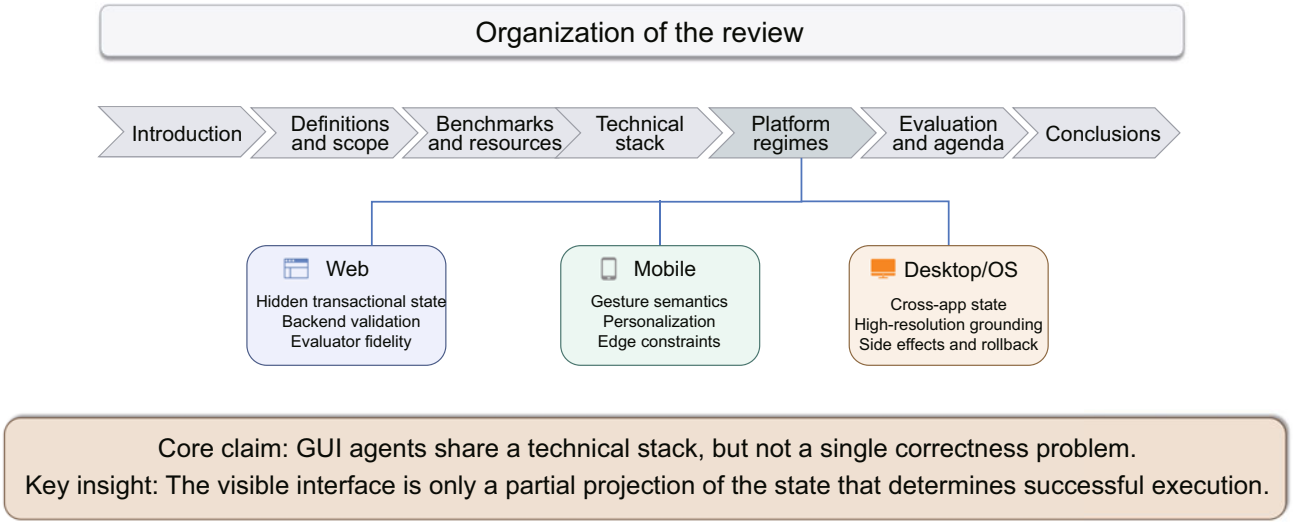

Fig. 1 Organization of the review. We first define the scope and organizing premise: GUI is an incomplete projection of the underlying system state. We then review benchmarks, models, and data across platforms through a common stack of observation, grounding, planning, memory, execution, and verification. Next, we compare web, mobile, and desktop/OS agents, followed by evaluation practices, industrial trajectories, results, and further research directions

prediction," "postcondition verification," "interactive benchmark," and "trustworthy agent." We cover benchmark papers, datasets, model and system papers, survey articles, and public technical or product reports that materially demonstrate how GUI-agent capabilities are defined, evaluated, and deployed. Works are included under four criteria: (1) focus on agents that observe or control graphical interfaces; (2) contribution of a benchmark, dataset, model, framework, evaluation protocol, deployment report, or conceptual analysis relevant to GUI-agent execution; (3) sufficient methodological details to identify the observation modality, action space, or evaluation setting; (4) assistance in clarifying differences across web, mobile, desktop/OS, or cross-platform settings. We exclude papers focused only on conventional robotic manipulation, purely text-only tools use with no GUI component, and low-level UI automation scripts with no language-conditioned or model-based GUI interaction. The review is intentionally selective rather than exhaustive, prioritizing resources that change what can be measured, reveal new failure modes, or clarify differences across web, mobile, and desktop/OS settings. Inclusion is guided by analytical relevance, not by completeness. Public technical reports, system cards, and product materials are included if they clarify capability definitions, deployment constraints, or evaluation practices that are not well captured in the current archival paper.

2 Definitions, scope, and organization perspective

2.1 What is a GUI agent?

We define a GUI agent as an autonomous or semiautonomous system that observes a human-facing interface and selects actions on that interface to accomplish a task objective. The interface may be a webpage, a mobile application, a desktop window, or a richer OS environment comprising multiple applications. Actions can be low-level, such as mouse clicks, taps, drags, keystrokes, or scrolls, or higher-level actions exposed through structured application programming interfaces (APIs), document object model (DOM) elements, accessibility trees, or tool calls (Hu et al., 2025; Nguyen D et al., 2025; Sager et al., 2026). The definition is broad by design, as the literature spans purely visual control, structure-assisted interaction, mixed GUI–tool execution, and agents that operate across more than one substrate. Table 2 previews representative benchmark and resource families across the three operational regimes, together with their main measurement capabilities and remaining limitations.

The scope can be precisely described by the following minimal formalization. Let $p \in \{\text{web}, \text{mobile}, \text{desktop/OS}\}$ denote the platform regime. A GUI environment can be represented as

$$\mathcal{E}_p = (\mathcal{S}_p, \mathcal{O}_p, \mathcal{A}_p, T_p, \Omega_p, C_p), \tag{1}$$

where $\mathcal{S}_p$, $\mathcal{O}_p$, and $\mathcal{A}_p$ are the latent system-state space, observation space, and action space, respectively, $T_p(s_{t+1} \mid s_t, a_t)$ is the state-transition kernel with s_t denoting the state at time t and a_t denoting the action at time t, $\Omega_p(s_t)$ is the observation function, and $C_p(s_t, a_t)$ is the execution cost or risk of an action. Given an instruction x, a GUI agent maintains a belief or memory state $b_t = B_\theta(x, o_{\leq t}, a_{<t})$ with $B_\theta(\cdot)$ denoting the belief or memory updating function, and chooses actions according to

$$a_t \sim \pi_\theta(a \mid x, o_{\leq t}, a_{<t}, b_t), \quad o_t = \Omega_p(s_t), \tag{2}$$

where $\pi_\theta(\cdot)$ is a policy function, o_t is the observation result at time step t, subscript $< t$ denotes all actions performed before time step t, excluding the current time step t, and $o_{\leq t}$ denotes all observations performed before and at time step t.

A task is completed when the postcondition $G_x(s_T) = 1$ holds in the underlying state; a plausible last local action is insufficient. Verification is a separate mapping $V_p(e_{\leq T}, x) \to \{\text{success}, \text{failure}, \text{uncertain}\}$, where the evidence e may include screenshots, DOM or accessibility structures, logs, tool outputs, file metadata, or human confirmation, and T is the final time step. This formulation clarifies why operationally

Table 2 Representative benchmark and resource families across GUI agent research

Platform	Representative resource	What becomes measurable	Persistent gap
Web	Mind2Web (Deng et al., 2023); WebArena (Zhou SY et al., 2024); VisualWebArena (Koh et al., 2024); BrowserGym (de Chezelles et al., 2025); WebArena Verified (El Hattami et al., 2025)	Browser workflows; visual browsing; executable tasks; evaluator auditing	Hidden transactions; backend checks; adversarial pages; recovery
Mobile	Android in the Wild (Rawles et al., 2023); AndroidWorld (Rawles et al., 2025); MobileAgentBench (Wang LY et al., 2024); AppAgent (Li YD et al., 2024); Mobile-Agent (Wang JY et al., 2024); AutoDroid-V2 (Wen et al., 2025)	Gesture control; dynamic emulators; exploration reuse; code-mediated actions	Personalization; privacy; interruptions; multiapp robustness
Desktop/OS	OSWorld (Xie TB et al., 2024b); Windows Agent Arena (Bonatti et al., 2025); WorkBench (Styles et al., 2024); OmniACT (Kapoor et al., 2024); UFO (Zhang CY et al., 2025b); ScreenSpot-Pro (Li KX et al., 2025)	Open computer use; workplace outcomes; high-resolution grounding; cross-app control	Tool arbitration; rollback; side effects; real-workflow evaluation

The table is selective and emphasizes resources that change what the field can measure

different platforms (web, mobile, and desktop/OS agents) differ in their observation, action, transition, and cost functions.

Two distinctions are important. First is the distinction between datasets, which provide fixed observations, demonstrations, or state–action pairs for static evaluations or supervised training, and environments, which are interactive and dynamic, meaning that actions change the state, observations are sequential, and the problem is a partially observable control process (Xie TB et al., 2024b; Zhou SY et al., 2024; Nguyen D et al., 2025). Second is the distinction between closed-world assumptions, which presume that all information necessary for solving the given task is contained within the benchmark itself, and open-world assumptions, which allow the existence of relevant information or state transitions outside the observed slice of the benchmark. Open-world assumptions often improve the realism but degrade the reproducibility of an evaluation.

2.2 Operational regimes and hidden states

GUI is an incomplete projection of the underlying system state that ultimately determines correctness. A visible form field does not reveal all server-side validation rules. A mobile screen frequently conceals background notifications, permissions, and app memory, all of which can alter the meaning of an identical gesture. A desktop workflow may appear locally coherent while silently editing the wrong file, using the wrong user account, or acting within the wrong window. Nevertheless, the model must observe a sufficient amount of the correctness-relevant state and discern any evidence of the intended state transition.

Web, mobile, and desktop/OS settings differ less in surface appearance than in where their main uncertainties concentrate. Uncertainty typically resides in the backend or transactional state of web agents, in gesture semantics, personalization, and local device context of mobile agents, and around structural heterogeneity, context switching, and costly side effects spanning tools and applications in desktop agents. These three concentrations of uncertainty will be repeatedly mentioned throughout this review because they (at least partially) explain why benchmark performance often transfers less effectively across platforms than raw scores suggest.

Herein, we use hidden state as an umbrella term for correctness-relevant information (backend state, device context, workflow history, and tool-mediated side effects) that is not fully recoverable from the currently visible interface. We divide the hidden state into four recurring categories: transactional states, including server-side validation, authentication, quotas, inventory, and other backend conditions that determine whether a visible web action actually commits; contextual device states, including permissions, notifications, logged-in accounts, cached preferences, and app memories that change the meaning of the same mobile gesture; workflow history states, including prior decisions, visited pages, discovered procedures, and unresolved subgoals that may no longer be visible; side-effect states, including files, messages, tool calls, and system settings whose consequences persist outside the current screen. In any benchmark or deployment scenario, one can then ask which state variables are exposed, which are merely inferred, and which are checked after execution. Concrete examples of platform-specific hidden states, typical failures, and corresponding verification or mitigation strategies are summarized in Table 3.

2.3 Review scope

This review covers the large-model era of GUI agents, which begins with the emergence of LLMs, multimodal LLMs (MLLMs), and large vision-language models. Automation of classical robotic processes, accessibility-driven automation, and deterministic testing frameworks remain important engineering baselines, especially in enterprise settings, but typically depend on stable schemas, scripting privileges, or narrow and predefined workflows. We focus on systems that must infer targets, plans, and recovery behaviors from partially structured or visually grounded observations with distribution shifts (Shi TL et al., 2017; Toyama et al., 2021; Shaw et al., 2023).

The scope is deliberately comparative. Instead of attempting an exhaustive catalog of every publication, we highlight the resources and systems that change what can be measured or reveal previously hidden bottlenecks. Benchmark

Table 3 Concrete cases illustrating platform-specific hidden states and verification needs

Regime	Representative case	Hidden state	Typical failure	Verification/Mitigation
Web	Checkout, booking, or enterprise form	Authentication; quotas; inventory; server-side validation; commit status	Visually successful but actually failed or wrong account transaction	Receipt/Order ID; backend status; evaluator audit (El Hattami et al., 2025)
Mobile	Settings, notification, or personalized app	Permissions; account identity; notifications; cached preferences	Same gesture triggers different behaviors	Adaptive sensing; permission checks; AndroidWorld-style dynamics (Rawles et al., 2025)
Desktop/OS	Spreadsheet-to-email workflow	File path; window focus; version; account; attachment metadata	Correct local steps but wrong file, window, or recipient	Metadata/Content checks; rollback; human confirmation (Styles et al., 2024; Cheng PZ et al., 2025)

design, evaluation fidelity, and evidence quality are treated with the same level of attention as raw architecture. When models, interfaces, and evaluation practices co-evolve, what becomes measurable tends to shape what becomes optimized.

3 Benchmarks and resources

Spanning multiple platforms, observation contracts, and definitions of success, GUI agent research extensively relies on benchmarks. Some resources are static datasets of screenshots, demonstrations, or instructions; others are executable tasks and dynamic state transitions in interactive environments. Some resources are highly instrumented and reproducible; others prioritize realism over strict control. Benchmarks have shifted from synthetic or narrowly scoped settings to more realistic, multimodal, and outcome-oriented settings (Wang S et al., 2024; Hu et al., 2025; Nguyen D et al., 2025). During 2025–2026, this trend has expanded to diagnostic and real-world resources such as BrowserArena for live web navigation, SafeArena for evaluating harmful misuse, AMEX for richer multi-annotation mobile supervision, MobileWorld and MobileWorldBench for more difficult mobile evaluations and semantic world modeling, and GUI-360° for desktop grounding, parsing, and action prediction (Anupam et al., 2025; Chai et al., 2025; Kong et al., 2025; Li SF et al., 2025; Mu et al., 2025; Tur et al., 2025).

3.1 Static datasets and pre-environment resources

Static resources remain important for fine-grained supervision, reproducible comparisons, and inexpensive ablation studies. For web tasks, datasets such as Mind2Web frame natural-language goals as mappings to action trajectories on real websites (Deng et al., 2023). In the wider GUI setting, screenshot-grounding resources such as CogAgent, SeeClick, and ScreenSpot-Pro can evaluate whether a model can localize elements or connect language to interface targets prior to full sequential control (Cheng KZ et al., 2024; Hong et al., 2024; Li KX et al., 2025). These resources enable the separation of grounding ability and long-horizon decision making, which is useful, because many systems fail at the former stage before the latter becomes relevant.

Static resources also support training of large UI-capable models. In pixel-to-action formulations such as Pix2Act, GUI control is framed as direct prediction from screenshots and instructions (Shaw et al., 2023), whereas newer datasets and synthetic corpora aim to support pretraining or fine-tuning for interface-centric perception. However, static datasets are known to under-represent hidden states, fail to fully capture temporal dependencies or interruptions, and commonly approximate task completion by action matching rather than outcome verification. This proxy can make the transition from benchmark evaluation to real deployment appear artificially smooth.

The coverage has been extended by static resources such as WebSRC and UnderstandingHTML, which treat web pages as structured reading and parsing objects. Meanwhile, UGIF, MUG, Falcon-UI, UIX, EDGE, UGround, and Read Anywhere Pointed support the following of UI-grounded instructions, screen reading, or understanding of text-rich interfaces (Chen XT et al., 2021, 2024; Venkatesh et al., 2022; Gur et al., 2023; Fan et al., 2024; Li T et al., 2024; Shen HW et al., 2024; Gou et al., 2025; Liu JP et al., 2025). Many later agents have adopted the screenshot corpora, component semantics, question–answer (QA) supervision, and cross-app navigation data supplied by MoTIF, Rico, Screen2Words, Screen2Vec, ScreenQA, MobileViews, GUI-Odyssey, and GUI-World (Deka et al., 2017; Li TJJ et al., 2021; Wang B et al., 2021; Burns et al., 2022; Gao LX et al., 2024; Chen DP et al., 2025; Hsiao et al., 2025; Lu QF et al., 2025). As confirmed in synthetic and broad-coverage corpora such as E-ANT, VideoGUI, and BigDocs, GUI supervision now spans screenshots, navigation traces, instructional videos, and document-like interface content (Lin KQ et al., 2024; Wang K et al., 2024; Rodriguez et al., 2025). Fig. 2 maps the benchmarks and resources onto static supervision, interactive environments, and outcome-oriented evaluation categories.

3.2 Interactive environments

Interactive environments expose the full control-loop nature of GUI agents. In such environments, the GUI agent must perform sequential decision-making under partial observability and evaluate whether an action invokes the intended change in the environment. The WebArena benchmarks introduce realistic and self-hosted browser tasks with executable evaluations (Zhou SY et al., 2024). VisualWebArena has extended this setup to visually grounded tasks that directly require screenshot interpretation (Koh et al., 2024). BrowserGym frames web-agent benchmarking as an ecosystem rather than a single

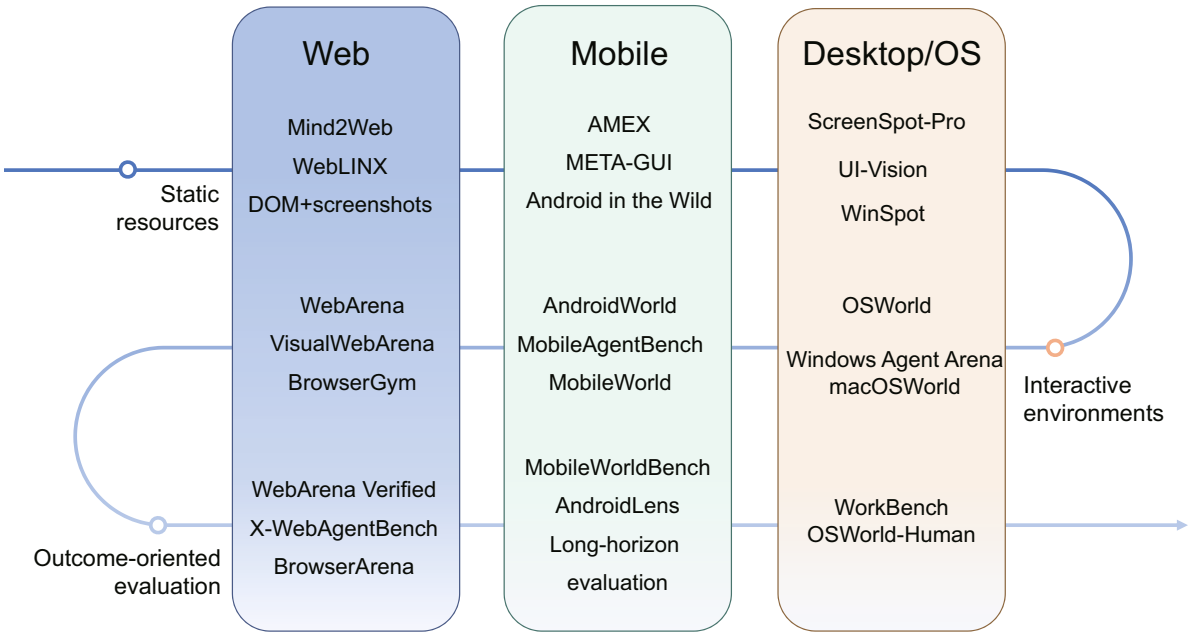

Fig. 2 Benchmark and resource map across the three operational regimes. The organization separates static supervision, interactive environments, and outcome-oriented evaluation instead of listing benchmarks as a flat list

benchmark, showing that task families, environment APIs, and evaluation assumptions all shape the meaning of a benchmark score (de Chezelles et al., 2025).

Mobile and desktop settings enable advanced environment-centric evaluations. AndroidWorld provides a dynamic Android environment with programmatic task construction and success checking (Rawles et al., 2025), whereas OSWorld brings open-ended computer-use tasks into a directly executable and multimodal benchmark setting (Xie TB et al., 2024b). Besides adding tasks, these resources elucidate the types of state transitions, interruptions, and hidden assumptions that are suppressed in static datasets. However, they complicate the evaluation because correctness cannot be inferred from a short local trace in a dynamic environment.

Recently developed related environments and benchmark variants have widened the range of stress conditions. WebLINX, WebVLN, NNetNav, RealWeb, VideoWebArena, MM-BrowseComp, WebArena-Lite-v2, and VeriWeb enable dialogue-conditioned navigation, complex demonstrations, lightweight realism, long-context web videos, multimodal browsing breadth, or verifiable information seeking (Chen Q et al., 2024; Lù et al., 2024; Murty et al., 2024; Zhang BL et al., 2024; Jang et al., 2025; OpenGVLab, 2025; Li SL et al., 2025; Liu SY et al., 2025). Mobile device frameworks such as MoTIF, Mobile-Env, AndroidLab, AndroidGen, and MemGUI-Bench clarify feasibility, environmental control, data scarcity, and dynamic memory variation (Burns et al., 2022; Zhang DY et al., 2023; Lai et al., 2025; Xu YF et al., 2025a; Liu GY et al., 2026). Broader-agent benchmarks such as AgentBench, GAIA, Shopping MMLU, MMT-Bench, ScienceBoard, and GAL are not limited to GUIs but are relevant because GUI control is increasingly evaluated within a larger ecosystem of multimodal and tool-using assistants (Jin et al., 2024; Liu X et al., 2024a; Mialon et al., 2024; Ying et al., 2024; Feng and Chen, 2026; Sun QS et al., 2026).

3.3 What benchmarks actually measure

A benchmark in GUI-agent research does not measure model capability in isolation; rather, it measures the performance of a system operating under a particular observation contract and evaluation protocol. Given the same natural-language instructions, the difficulty of an evaluation depends on whether the agent can access the DOM, the accessibility tree, programmatic success signals, or screenshots alone. Likewise, whether the evaluator checks trajectories, visible UI states, or backend outcomes determines the types of counted errors. For instance, WebArena Verified demonstrates that evaluator design can become a research bottleneck (El Hattami et al., 2025). Quantitative trend figures should therefore be read together with provenance on benchmark version, observation modality, tool or API access, evaluation metric, retry policy, and verifier or human intervention.

For this reason, benchmark diversity in GUI agent research is an integral component that cannot be simply abstracted as noise. Datasets, interactive environments, and outcome-oriented evaluations address different evaluation needs. Different benchmarks may reveal or fail to reveal hidden failure modes. For example, benchmarks that expose DOM or accessibility structure may overestimate transfer to screenshot-only deployment, whereas closed-world tasks may understate the recovery burden of open-world interactions.

4 Technical stack

Recent GUI agent systems can be analyzed through a common technical stack (Fig. 3) encompassing interface observation, grounding, planning and memory, action execution, and outcome verification (Wang S et al., 2024; Nguyen D et al., 2025; Sager et al., 2026). This decomposition mirrors the control loop common to all platforms while revealing the pressure points of each operational regime. As illustrated in Fig. 3,

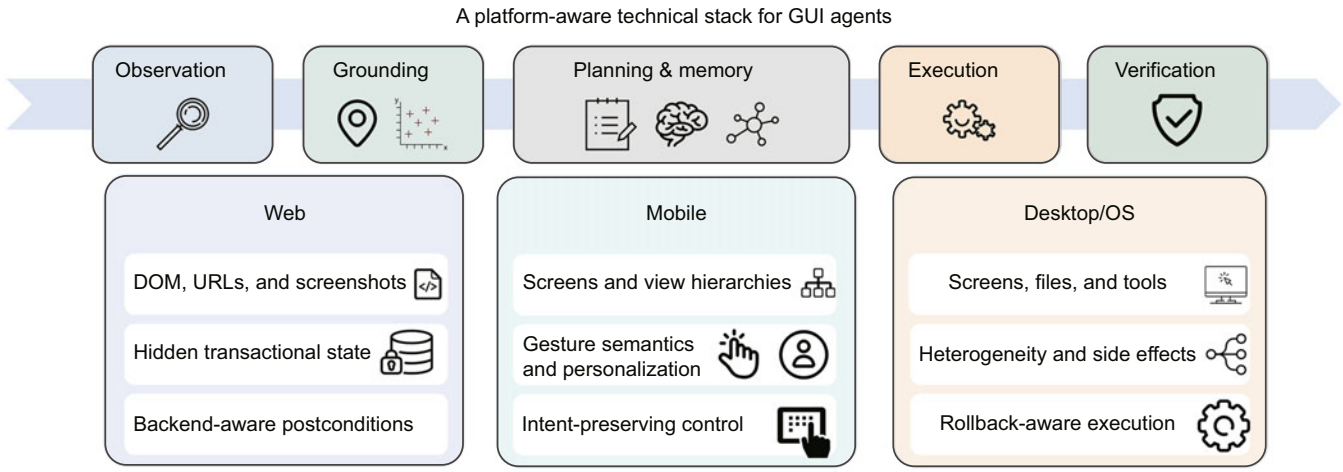

Fig. 3 A platform-aware technical stack for GUI agents. The high-level control loop is shared across platforms, but the main sources of uncertainty and the needed recovery strategies differ by operational regime

the primary points of failure and recovery differ among web, mobile, and desktop/OS environments.

4.1 Perception and observation contracts

The foundational layer determines what an agent can inspect prior to reasoning. Depending on the environment, observations can be provided through screenshots, hyper text markup language (HTML)/DOM trees, accessibility trees, optical character recognition outputs, hierarchical views, uniform resource locators, tool-returned metadata, or other system-state variables exposed by the benchmark. These observation channels are not equivalent. A DOM tree provides highly privileged and structured compression of a webpage, an accessibility tree offers semantic annotations if correctly implemented, and a raw screenshot preserves visual fidelity but discards all structural priors. Hybrid systems combining multiple channels can improve the robustness but complicate the interpretation of performance claims (de Chezelles et al., 2025; Hu et al., 2025; Nguyen D et al., 2025).

Observation design is a genuine research variable, not a minor implementation detail. Web agents matured earlier than other agents, partly because web benchmarks have historically exposed richer structured information. In contrast, mobile and desktop settings tend to rely heavily on screenshots and partial accessibility signals, directly placing the perception and grounding burdens on the model. Stronger performance in one setting may reflect a better observation contract, but does not necessarily imply stronger reasoning.

Recent research has diversified the observation substrate. In systems such as OmniParser, MobileVLM, and UI-Hawk, pure vision or vision-first parsing can recover actionable structures from screenshots and screen streams when platform metadata are weak or unavailable (Lu YD et al., 2024; Wu QZ et al., 2024; Zhang JW et al., 2025). Moreover, HTML-centric work has shown that structured web understanding remains important when the interface substrate is at least partially machine-readable (Gur et al., 2023).

4.2 Grounding

Grounding maps assign instructions to specific interface targets, coordinates, and control affordances, connecting observations to executable action. Early methods such as Pix2Act approached this problem through direct pixel-to-action prediction (Shaw et al., 2023). More recently, CogAgent, SeeClick, ShowUI, and FerretUI 2 have established grounding as a separate capability in general GUI operation (Cheng KZ et al., 2024; Hong et al., 2024; Li ZH et al., 2025; Lin KQ et al., 2025). Benchmarks such as ScreenSpot-Pro have revealed the difficulty of grounding professional and high-resolution computer-use interfaces, even by otherwise capable models (Li KX et al., 2025).

Grounding difficulty is platform-sensitive. Repeated design conventions or structural hints can sometimes be exploited in web environments. Mobile interfaces compress actions into gestures and local idioms, tightly coupling grounding with action semantics. In desktop environments, the challenges are amplified through interface density, visual clutter, and cross-window ambiguity. A single aggregate grounding score is rarely informative because it ignores the specific interface density, scale variation, and target ambiguity in each setting.

Grounding methods beyond generic heuristics have been recently established. As demonstrated in Set-of-Mark prompting, enhanced visual marking can improve the grounding of large multimodal models (Yang JW et al., 2023). Techniques such as iterative narrowing, GUI-SPOTLIGHT, Jedi, and Visual Test-time Scaling target more precise localization under clutter conditions, scale-variable conditions, or difficult test-time distributions (Nguyen A, 2024; Luo et al., 2025; Xie TB et al., 2025; Lei et al., 2025). UGround, AGUVIS, and Ponder & Press are universal pure-vision variants that treat grounding as a separate research problem, not merely as a preprocessing step in agent prompting (Gou et al., 2025; Wang YQ et al., 2025; Xu YH et al., 2025b).

4.3 Planning, memory, execution, and verification

GUI tasks are inherently sequential. Even in tasks involving simple individual actions, the overall success often requires subgoal formation, interruption recovery, searching, revisiting previous states, and state-tracking over long horizons. Planning allocates reasoning effort across these trajectories, while memory stores the relevant state across

observations, actions, discovered workflows, and user-specific contexts. AppAgent and its successor use explicit mechanisms for exploration, knowledge reuse, and retrieval over previously encountered interfaces (Li YD et al., 2024; Zhang C et al., 2025). Agent S also leverages reused experience and hierarchical planning for long-horizon computer use (Agashe et al., 2025).

Execution converts plans into concrete interventions. Defined by the environment, the action space may include point-and-click events, gestures, keystrokes, structured browser commands, or tool- and script-mediated operations. Representative systems are WebVoyager for screenshot-driven web interactions (He et al., 2024), Mobile-Agent for multimodal smartphone control (Wang JY et al., 2024), and UFO- and OSWorld-based systems for cross-window desktop execution (Xie TB et al., 2024b; Zhang CY et al., 2025b). The choice of execution modality is itself a main decision: as demonstrated in OmniACT, AutoDroid-V2, and OSWorld-MCP, GUI actions, code generation, or external tool calls may be more appropriate than a default click-based approach (Kapoor et al., 2024; Wen et al., 2025; Jia et al., 2026).

The final layer is verification. A plausible sequence of local actions does not guarantee a correct global outcome. For instance, a clicked button may not change the backend state, a drag may trigger an unintended gesture, and a file operation may complete the wrong target. To resolve these problems, WebArena Verified, WorkBench, and OS-Kairos shift the evaluation toward postcondition-checking, outcome-quality assessment, intervention decision support, and recovery planning (Styles et al., 2024; Cheng PZ et al., 2025; El Hattami et al., 2025). At the system level, verification can be implemented through precondition checks before irreversible actions, runtime monitoring during execution, and postcondition audits after execution. The focus has moved from generating plausible local actions to verifying the occurrence of the intended state transitions.

Control aspects of the technical stack are covered by a wide range of systems, from single-turn automation (TinyClick) and explicit task decomposition (dynamic planning for GUI automation) to mobile-specific planning (MobileFlow and MobA), voice-command execution (GPTVoiceTasker), and earlier automation baselines (AutoTask and Droidbot-GPT) (Pan LH et al., 2023; Wen et al., 2023; Nong et al., 2024; Song et al., 2024; Vu et al., 2024; Zhang SQ et al., 2024; Zhu et al., 2024; Pawlowski et al., 2025). Recent work has added spatiotemporal memory (GUI-KV), exploration strategies (GUI-Explorer and UI-Evol), progress-aware rewards (ProgRM), world-model or code-assisted planning (FPWC and ScaleTrack), and explicit recovery mechanisms (BacktrackAgent and GTA1) (Huang J et al., 2025; Wu QZ et al., 2025; Xie B et al., 2025; Yin et al., 2025; Zhang DY et al., 2025; Zhang ZY et al., 2025; Huang KH et al., 2025; Yang Y et al., 2026). Success detection and self-refinement through dedicated verifiers, multimodal auto-validation, and diagnostics to narrow the reasoning–execution gap have received increasing attention, confirming that fluent verbal reasoning does not automatically translate into correct GUI behavior (Du et al., 2023; Azam et al., 2024; Dong et al., 2025).

5 Web agents

Early research on GUI agents was often performed in web environments, which offer easier data collection, more reproducible environment construction, and a clearer path from structure-heavy parsing to screenshot-driven multimodal interaction than mobile and desktop settings. Browser-based tasks have assisted work on instruction following, multimodal grounding, long-horizon planning, and outcome-based evaluation (Deng et al., 2023; Drouin et al., 2024; Koh et al., 2024; Zhou SY et al., 2024; de Chezelles et al., 2025).

5.1 Benchmarks and systems

Web-agent benchmarks have moved from static datasets to interactive and outcome-oriented evaluations. Mind2Web is one of the first large-scale resources for evaluating generalist web agents on real websites, and it frames the problem as a mapping task of natural-language intents to executable action trajectories (Deng et al., 2023). Later, WebArena introduces a controlled and self-hosted browser environment with executable tasks, advancing the evaluation from offline action matching to online performance assessment (Zhou SY et al., 2024). VisualWebArena designs tasks in which success depends directly on screenshot interpretation, removing the need for privileged DOM access and increasing the emphasis on visual perception (Koh et al., 2024).

Later benchmarks have further widened this scope. WorkArena and BrowserGym extend the coverage to enterprise-style workflows and benchmark ecosystems beyond single fixed-task sets (Drouin et al., 2024; de Chezelles et al., 2025). WebCanvas explores the challenges of online evaluation on live and changing websites (Pan YC et al., 2024). More recent efforts, such as X-WebAgentBench, BrowserArena, and SafeArena, have extended evaluations to multilingual and truly open web settings, explicitly incorporating safety and misuse prevention as evaluation criteria (Anupam et al., 2025; Tur et al., 2025; Wang P et al., 2025).

Web agent systems range from structure-assisted to pure-vision approaches. WebVoyager shows that large multimodal models can browse and complete tasks using visual observations and textual prompts as the main inputs (He et al., 2024). This work draws on broader agent frameworks such as ReAct, Reflexion, and Toolformer, which provide reasoning, tool use, and self-correction patterns that directly apply to web tasks (Schick et al., 2023; Shinn et al., 2023; Yao et al., 2023). WebGPT also shows that the web is a useful domain for evaluating language agents in interactive and partially observable environments (Nakano et al., 2021).

Other work covers additional gaps. As highlighted in studies on workflow-guided exploration and natural-language interfaces of web APIs, web interaction is not reducible to static page parsing (Su et al., 2017; Liu EZ et al., 2018). More recent systems such as SeeAct, AutoWebGLM, AgentOccam, and ScribeAgent have contributed methods for grounded prompting, bootstrap-and-reinforce training, strong minimalist baselines, and specialization to production workflows (Lai et al., 2024; Shen JH et al., 2024; Zheng et al., 2024; Yang K et al., 2025). Diagnostic analyses using the WebSuite and WorkArena++ benchmarks, along with studies on model

limitations, have shifted the focus from headline scores to the underlying reasons for failure, particularly during compositional, enterprise-style, or sequentially dependent tasks (Boisvert et al., 2024; Furuta et al., 2024; Li E and Waldo, 2024).

5.2 Hidden-state problem on the web

The core technical problem in web-agent research is the gap between the visible-page state and the system state that determines correctness. Although a webpage may appear submission-ready, the backend validations may remain unsatisfied. A button click may produce the correct visual feedback while the intended transaction silently fails. Factors such as authentication, permissions, quotas, server-side logic, inventory status, and asynchronous updates can govern success without being fully legible in the current observation. The web regime is often more benchmark-friendly than desktop or mobile environments, but clearly exposes the gap between visible interaction and true task completion. The same problem is exposed in checkout, ticket-booking, and enterprise-form workflows. Click of a submit button may be followed by progress or confirmation-like feedback on the page, but the actual success depends on payment authorization, inventory availability, account state, server-side validation, and creation of a durable order or request identifier. These tasks should therefore be verified through backend visible evidence such as a receipt, confirmation identifier (ID), saved record, or an independently checkable task state, not through the click event or transient visual feedback alone.

This gap has been directly addressed through platforms such as WebArena Verified, which has demonstrated that reliable assessment requires checking completion conditions post hoc; relying on superficial action traces or incomplete environment signals is insufficient (El Hattami et al., 2025). WorkArena similarly emphasizes outcome-centric evaluations in realistic work settings (Drouin et al., 2024). Although the existing benchmarks are not flawed in isolation, they are increasingly designed for transaction management evaluations—checking for the intended state changes—rather than for simple navigation proficiency evaluations.

5.3 Robustness and open-world web interaction

The web imposes concrete trustworthiness and robustness challenges. Because the agents operate over pages that may contain irrelevant, dynamic, misleading, or even adversarial content, the browser surface becomes part of the attack surface. WebInject has demonstrated that prompt injection can be embedded directly in web content to subvert agent behaviors (Wang XL et al., 2025). These problems are amplified in open-world and live-web settings, in which the page content, site layouts, and available information can change between runs.

Beyond accurate next-step prediction, Web agents must perform transaction-aware execution. The agent must determine whether the intended backend state change has occurred, check whether the evidence is sufficient to proceed, and decide when to trigger additional verification or recovery procedures. Concrete defenses include checking the postcondition after updates, confirming the existence of durable artifacts (receipt and order ID), and escalating to human review before irreversible or security-sensitive actions can be executed. Security-focused research using PopupAttack, AdvAgent, EIA, WIPI, along with analyses of refusal-trained LLMs subverted as browser agents, illustrates the diverse ways in which page content, browser context, and hybrid web–OS settings can undermine the agent's intent (Kumar et al., 2024; Wu FZ et al., 2024; Liao et al., 2025; Xu CJ et al., 2025; Zhang YZ et al., 2025). These findings show that web-agent robustness is a property of the interaction between the model's policy and a mutable and often untrusted environment, not merely a static prompt-safety problem.

Recent web-agent research has improved instruction-following in partially structured browser environments and visually grounded navigation, removing basic browsing fluency as the main bottleneck. More difficult problems—dependable transaction completion under hidden backend states, dynamic content, and adversarial surfaces—remain unsolved. Progress in this regime will likely depend on postcondition verification, uncertainty-aware recovery, and explicit defenses against environment-level attacks, rather than on improvement of local action prediction.

6 Mobile agents

Within mobile environments, GUI agent research has shifted from the browser to app-centric and device-level control, introducing a different set of challenges. The interface structure in mobile settings is less stable and more heavily depends on gesture-based interaction than the web; moreover, mobile interfaces are deeply shaped by user personalization, privacy constraints, and on-device resource limitations. Mobile agents are a key testbed for sequential decision-making under strong partial observability, where the visible screen is only a local snapshot of a complex, stateful, personalized device context (Rawles et al., 2023, 2025; Li YD et al., 2024; Wang JY et al., 2024; Wang LY et al., 2024; Wen et al., 2025; Zhang C et al., 2025).

6.1 Benchmarks and systems

Mobile agent benchmarks have progressed from static datasets to dynamic and executable environments. Android in the Wild is an influential large-scale resource that pairs screenshots with executable actions and instructions across diverse devices and software states (Rawles et al., 2023). MobileAgentBench curates tasks across open-source applications graded into difficulty levels, enabling advanced systematic comparison (Wang LY et al., 2024). AndroidWorld provides a functional Android emulator with programmatic task construction and automated success checking, marking an important step toward reproducible evaluation (Rawles et al., 2025). More recent resources such as AMEX, MobileWorld, and MobileWorldBench add multi-level annotations, enabling interactive evaluations that simulate real user behavior, along with semantic world models for planning-driven agents (Chai et al., 2025; Kong et al., 2025; Li SF et al., 2025).

Mobile-agent system design has developed along several directions. AppAgent uses a simplified action space,

autonomous exploration, and reusable interface knowledge bases (Zhang C et al., 2025). Its successor, AppAgent v2, provides a more flexible action space and structured knowledge retrieval (Li YD et al., 2024). Mobile-Agent has demonstrated the feasibility of vision-centric operation that minimizes reliance on platform-specific metadata such as accessibility trees (Wang JY et al., 2024). AutoDroid-V2 has shown that some mobile tasks are better handled by generating and executing procedural scripts than by executing incremental GUI interactions (Wen et al., 2025). As highlighted in DigiRL, reinforcement learning (RL) can assist the adaptation of control policies to the open-world variability of real devices (Bai H et al., 2024). AgentCPM-GUI is a recent example of compact, bilingual agents that combine grounding-aware pretraining with high-quality supervision and reinforcement fine-tuning (Zhang Z et al., 2025).

Additional aspects of data and control have also been investigated. Early works such as META-GUI foreshadowed conversational mobile agents (Sun LT et al., 2022). Subsequent studies have explored multi-agent collaboration (MobileExperts), learning from demonstrations (LearnAct), benchmark curation (AndroidControl-Curated), and interaction with complex UI elements (AndroidLens) (Zhang JY et al., 2024; Cao et al., 2025; Leung et al., 2025; Liu GY et al., 2025). Berkovitch et al. (2025) inferred goals from UI trajectories, suggesting that immediate screen targets are insufficient and that effective automation must reason the latent user intent.

6.2 Gesture semantics, personalization, and edge constraints

Mobile agent research is shaped by properties that distinguish mobile agents from web and desktop interactions. Action semantics are compressed into gestures whose meanings largely depend on context; for instance, a swipe can navigate, reveal a menu, dismiss a notification, or cause an unintended state transition. The correct interpretation is governed by local app conventions. Mobile devices are also inherently personalized, with notifications, logged-in accounts, privacy permissions, interaction histories, and cached data determining what appears on the screen and what constitutes a successful user-level outcome. Execution operates under stringent edge constraints, including privacy requirements, latency, battery life, and computational limitations. These factors are often abstracted away in browser-based benchmarking.

As demonstrated in mobile settings and notification-management tasks, the same instruction, such as disabling alerts for an application or changing a default account setting, may generate different screens depending on the permission prompts, operating system version, logged-in identity, notification state, and app-specific layout. A visually plausible tap sequence may change the wrong setting, dismiss a transient overlay, or fail silently if the required permission has not been granted. These cases show that the mobile hidden state is not limited to unseen pixels, but also includes device context, account state, privacy permissions, and personalized interface history.

These distinct properties create practical challenges in system design and evaluation. A benchmark may correctly specify a task objective without capturing the ambiguity introduced by real-world interruptions, personalized content, or device-specific UI variations. Conversely, an agent may perform well in a controlled lab setting but proves fragile when deployed in the unique and stateful device context of a user. Simple action–sequence matching is a poor proxy of mobile evaluation. Instead, the agent must efficiently, reliably, and safely achieve the intended user-level outcome under realistic state variability. Useful mitigations are re-checking the screen after each gesture, verifying the permission and account states before sensitive actions, supplementing screenshots with view-hierarchy signals when available, and requiring explicit confirmation of privacy-sensitive or irreversible steps.

7 Desktop and operating-system agents

Desktop and operating-system environments are among the most heterogeneous settings of GUI agents. In such environments, the agent must coordinate tasks across multiple, disparate applications, manage windows and file systems, switch between user accounts, and interface with external tools and scripts (Kapoor et al., 2024; Styles et al., 2024; Xie TB et al., 2024b; Agashe et al., 2025; Bonatti et al., 2025; Zhang CY et al., 2025b; Jia et al., 2026). Unlike web tasks, which are often confined to a single browser session, or mobile tasks, which typically follow app-centric patterns, desktop workflows routinely span different software ecosystems and generate costly or irreversible side effects. Desktop and OS agents are useful testbeds for dependable computer use because they require reliable multistep task management across applications, not mere navigation within a single interface.

7.1 Benchmarks and systems

Much of the recent work has been performed on executable desktop benchmarks. OSWorld provides an executable environment for evaluating multimodal agents on diverse and open-ended computer-use tasks (Xie TB et al., 2024b). WorkBench enables realistic workplace workflows and outcome-centric evaluations in knowledge-work settings (Styles et al., 2024). Windows Agent Arena is a dedicated benchmark for multimodal agents operating within the Windows OS (Bonatti et al., 2025). Collectively, these resources have secured desktop/OS control as a true benchmark category, not simply an extension of web or mobile interactions.

Various design approaches for systems operating in these environments have been proposed. UI-Focused Agent (UFO) is tailored for Windows interactions, emphasizing structured UI understanding (Zhang CY et al., 2025b). Agent S enables experience reuse and hierarchical planning for general computer-use tasks (Agashe et al., 2025). OmniACT treats certain tasks as suitable for direct program generation and execution, rather than step-by-step GUI manipulation (Kapoor et al., 2024). OSWorld-MCP evaluates integrations of external tools, demonstrating the increasing operation of effective desktop agents over a mixed-action space encompassing GUI actions and tool-mediated steps (Jia et al., 2026).

As demonstrated in grounding benchmarks such as ScreenSpot-Pro, WinSpot, and Phi-Ground, models are

persistently challenged by visual density and complex professional desktop interfaces (Hui et al., 2025; Li KX et al., 2025; Zhang MS et al., 2025). WorldGUI highlights the need for robustness to dynamic initial states and careful planning of desktop automation (Zhao et al., 2025).

Research is also expanding beyond pure GUI control. UI-Vision extends benchmarking toward desktop-centric perception and interaction tasks (Nayak et al., 2025). Proposals such as Turn Every Application into an Agent and Commands to Prompts allow the reframing of desktop automations around application APIs and semantic file systems, pointing toward hybrid GUI–tool control (Lu JT et al., 2024; Shi ZR et al., 2025). Systems such as AssistEditor and Magentic-UI show that practical workflows often require multi-agent coordination or human-in-the-loop oversight rather than full autonomy (Gao DF et al., 2024a; Mozannar et al., 2025).

7.2 Structural heterogeneity and costly side effects

The main challenge in desktop/OS agency is not task length but the distribution of correctness across multiple loosely coupled systems. A successful workflow may require locating a specific file, editing that file in one application, processing it through another application, and sharing it via a third application while maintaining the correct file version, user account context, and application window focus. Even when each individual step is locally correct, a mismatch in any of the distributed state components can cause failure of the overall task.

This problem is exposed in spreadsheet-to-email workflows. An agent may open a spreadsheet, edit a table, export a file, and compose an email. Nevertheless, the task will fail if the agent edits an outdated copy, saves it to the wrong directory, attaches the wrong version, or sends the message to the wrong recipient. The relevant hidden state includes the file paths, active window focus, version history, user account identity, attachment metadata, and recipient state. Verification therefore requires not merely the apparent success of local GUI actions, but outcome-level evidence such as file content, file metadata, attachment identity, target recipient, and application state.

Evaluation of such workflows is difficult because success cannot be judged by the accuracy of an isolated action. Multiple failures are detectable only at the final outcome level. WorkBench directly addresses this problem by focusing on outcome-centric evaluations (Styles et al., 2024). OSWorld-Human compares both the success rates and efficiencies of agents and humans, revealing that excessive or redundant interaction steps degrade the performance of agents (Abhyankar et al., 2025). Desktop-agent evaluation must account for both functional correctness and execution economics.

Desktop agents also face costly side effects. Irreversible or high-stake actions such as file deletions/overwrites, email- or message-sending, form submission, changing of system permissions, and invoking external tools are common in desktop and OS environments. Besides accurate grounding and planning, dependable desktop agents need mechanisms for confidence estimation, rollback-aware decision making, and intelligent tool arbitration. Practical safeguards are pre-execution checks for active file paths and window focus, checkpointing or versioning before destructive edits, sandboxed or dry-run execution of tool calls when possible, and explicit confirmations before email sending, file deletion, permission changes, or other irreversible operations. Human oversight is not merely a fallback for edge cases, but a component of a practical deployment strategy, as evidenced in outcome-checking and agent self-reflection studies (XLANG Lab, 2025).

8 Models, data, and training

Although this review is organized around platform-specific challenges rather than architecture alone, the model design is also important. GUI agent models differ in their interface representations, coordination of reasoning and memory, and use of training signals. These design choices interact with the operational regime: web agents benefit from structured observations, mobile agents are challenged by compact vision-first control, whereas desktop/OS agents frequently demand hybrid execution, tool routing, and longer-horizon state tracking (Wang S et al., 2024; Hu et al., 2025; Nguyen D et al., 2025). While earlier work focused on visual-only and structure-assisted systems, recent work has investigated agents with native reasoning and reflection capabilities (Liu YH et al., 2026), compact models for on-device deployment (Yang Z et al., 2026), and foundation-model families that jointly optimize the grounding, navigation, and tool use across multiple platforms (Liu ZY et al., 2025; Ye et al., 2025; Zhou HZ et al., 2025; Xu HY et al., 2026).

8.1 Cross-platform and generalist agents

A growing line of work has targeted cross-platform or generalist GUI agents with capabilities beyond those of single-environment systems. Browsers, mobile agents, and desktop interactions have been treated as a common problem family (Hu et al., 2025; Nguyen D et al., 2025; Xie JL et al., 2025; Sager et al., 2026). This line includes cross-platform grounding models such as Ferret-UI, Ferret-UI Lite, ShowUI, Aria-UI, R-VLM, and Phi-Ground (Li ZH et al., 2025; Lin KQ et al., 2025; Park J et al., 2025; Yang YH et al., 2025; Zhang MS et al., 2025; Yang Z et al., 2026), cross-environment benchmarks such as CRAB and GUI-360° (Mu et al., 2025; Xu TQ et al., 2025), and foundation-style agents such as OS-Atlas, Mobile-Agent-v3, Mobile-Agent-v3.5, and ScaleCUA (Liu ZY et al., 2025; Wu ZY et al., 2025; Ye et al., 2025; Xu HY et al., 2026). These systems do not eliminate cross-platform differences, but learn reusable capabilities (visual grounding, action abstraction, memory, and tool invocation), thus exposing the shared technical stack.

You Only Look at Screens, AutoGLM, and GUI-Course are related generalist efforts that target chain-of-action reasoning, foundation-style GUI agents, and curriculum-style transfer from general vision-language models to versatile GUI behavior, respectively (Liu X et al., 2024b; Zhang ZS and Zhang, 2024; Chen WT et al., 2025). OpenAgents and early studies on tool manipulation have placed GUI control within the wider context of open-ended and tool-using language agents (Xu QT et al., 2023; Xie TB et al., 2024a).

Very recently (2025–2026), it was shown that generality

requires not only larger backbones, but also improved training pipelines. Systems such as Explorer, AgentTrek, Synatra, UI-Venus, Chain-of-Agents, and PolySkill use replay buffers, trajectory synthesis, reinforcement fine-tuning, multi-agent distillation, or compositional skill abstraction for supervision scaling or skill reuse of GUI agents (Ou et al., 2024; Gu et al., 2025; Li WZ et al., 2025; Pahuja et al., 2025; Xu YH et al., 2025a; Yu et al., 2025). Agent backbones and data engines are becoming more reusable, although transfer across operational regimes remains imperfect.

8.2 Structure-assisted, visual, and hybrid representations

Agent families differ in their representations of the interface. One broad line of work focuses on structured interface representations such as DOM trees and accessibility hierarchies. In web settings, these methods exploit semantically rich fields before the model begins reasoning. A second family, in which screenshots are the primary input, includes CogAgent, SeeClick, ShowUI, and related screenshot-centric agents (Cheng KZ et al., 2024; Hong et al., 2024; Lin KQ et al., 2025). These methods are pushing multimodal models toward end-to-end visual grounding. A third (hybrid) family integrates multiple modalities, for example, screenshots and structured signals, to improve the robustness when one channel is missing or unreliable.

The choice of family influences both the system behavior and the interpretation of benchmark scores. Structure-assisted systems can be sample-efficient and interpretable in web tasks, but may fail when the structure is absent or poorly implemented. Pure visual systems may be broadly generalizable but are sensitive to tiny targets, dense layouts, and visual ambiguity. In practice, both families offer distinct advantages but impose their own limitations: structure-assisted methods can overstate robustness when the markup is unusually clean or privileged, whereas pure-vision methods are more portable but remain brittle on dense professional interfaces and visually similar affordances. Although hybrid systems offer a practical compromise, they introduce additional synchronization and engineering complexity.

8.3 Planning frameworks and memory mechanisms

Different GUI agents organize their sequential reasoning in different ways. Some systems primarily rely on prompting and general-purpose LLM inferences, often inspired by ReAct-style interleaving of action and thought (Yao et al., 2023). Other systems incorporate explicit reflection or verbal-feedback loops, similar to Reflexion (Shinn et al., 2023). External memory, retrieval, and reusable procedure libraries are essential in longer and more complex tasks where the relevant state may no longer be visible on screen, as clarified in AppAgent-style knowledge stores and Agent-S-style experience reuse (Li YD et al., 2024; Agashe et al., 2025; Zhang C et al., 2025). Effective GUI interaction is required to maintain a working belief of the task progress, not merely to solve a sequence of isolated local decisions.

8.4 Supervision and training signals

Training signals for GUI agents are heterogeneous, deriving from imitation learning of human demonstrations, supervised grounding data, synthetic action traces, preference-based optimization, RL, and mixed curricula that combine static data with interactive environments. Early research using WebShop and WebGPT showed the utility of interactive feedback in web-grounded agents (Nakano et al., 2021; Yao et al., 2022). More recently, the DigiRL approach has shown the importance of RL-style objectives for improving in-the-wild device control performance (Bai H et al., 2024). OS-Atlas and UI-TARS have shown how large-scale GUI-grounding corpora and action traces operate as foundation-style pretraining signals in generalist GUI control (Qin et al., 2025; Wu ZY et al., 2025). Mobile-Agent-v3 adds scalable environmental interaction RL to the design space (Ye et al., 2025). Meanwhile, multimodal pretraining over screenshots and UI corpora remains indispensable because many downstream failures originate in perception and grounding, not in high-level planning.

Complementary work has focused on scaling and curating the supervision itself. Projects such as GUICourse, LearnAct, AndroidControl-Curated, and User-Goal Mining from UI Trajectories aim to provide improved curricula, cleaner demonstrations, or easily interpretable task labels for downstream agents (Berkovitch et al., 2025; Chen WT et al., 2025; Leung et al., 2025; Liu GY et al., 2025). This work complements environment-based RL by improving what the agent is taught before its full interactive deployment.

Beyond algorithm choice, training must question which hidden-state assumptions will be learned by the model through the training signals. A system trained predominantly on static demonstration traces may become proficient at local pattern imitation while remaining weak at postcondition verification or error recovery. Rich environmental feedback during training may improve the robustness of a system, provided that the environment measures the appropriate notion of correctness. Evaluation fidelity and training-signal design are therefore closely linked.

9 Evaluation, robustness, and trustworthiness

Fig. 4 organizes the design space into three dimensions: interface representation, control strategy, and training signal. The dominant gaps lie between the front-end actions and backend state in web evaluations, between gesture completion and user-goal attainment in mobile evaluations, and between locally correct steps and globally correct outcomes in desktop/OS evaluations.

Benchmarks cannot be scored in terms of action-prediction accuracy alone. Reported results are jointly shaped by what the agent is permitted to observe, the tools invoked by the agent, how the environment is instrumented, how success is verified, and the tolerable execution cost. Although these factors are relevant on all platforms, they are particularly salient when comparing browser tasks, mobile-device control, and desktop/OS workflows under different benchmark assumptions. Figs. 5, 6, and 7 present benchmark-conditioned snapshots of frontier results. To avoid presenting these snapshots

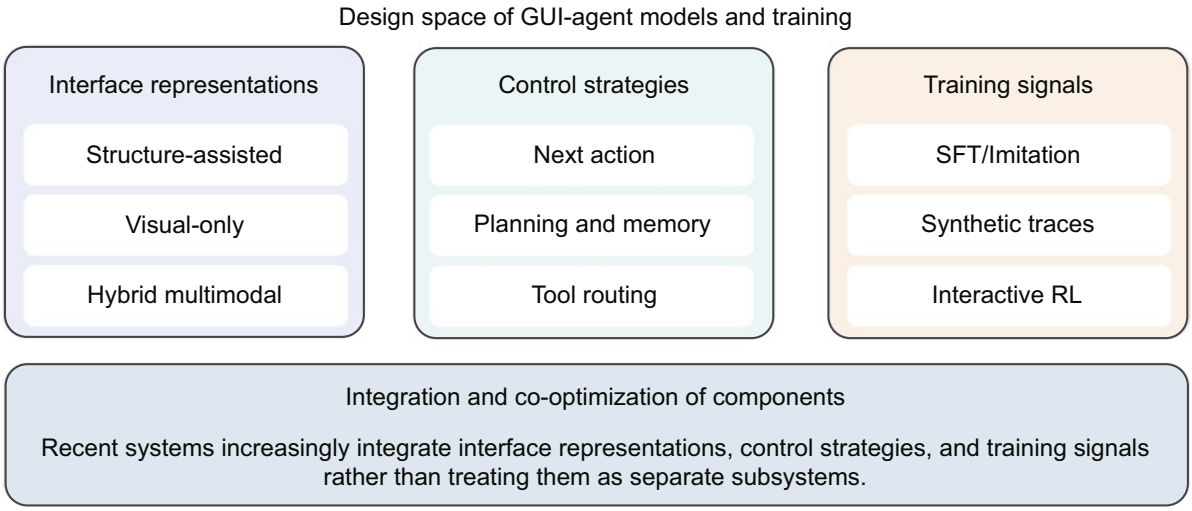

Fig. 4 Design space of GUI-agent models and training processes. The diagram groups recent systems by interface representations, control strategies, and training signals, and highlights the integration and co-optimization of these components

as a single leaderboard, Tables 4–6 record the provenance fields from which the plots are interpreted. In these tables, the reporting dimensions are standardized but the scores are not normalized across benchmarks because the observation contracts, retry budgets, tool access, and verifier design differ across sources.

9.1 Benchmark scores as system-level outcomes

Benchmark scores in GUI agent research are best interpreted as system-level outcomes, not as pure model-capability measures. For instance, the points in Figs. 5, 6, and 7 are reported by different organizations under different observation contracts, tool-access assumptions, and verification protocols. A reported success rate depends on the model capability, observation privilege, exposed structure, auxiliary tools, and evaluation fidelity. To illustrate this combined assessment, BrowserGym frames web-agent research as an ecosystem of tasks, interfaces, and evaluation settings, not as a single benchmark instance (de Chezelles et al., 2025). Similar considerations apply to non-web frameworks such as AndroidWorld and OSWorld, where evaluation fidelity also strongly depends on the initialization, instrumentation, and checking of the environment (Xie TB et al., 2024b; Rawles et al., 2025).

Structured browser observations, accessibility trees, and programmatic success signals can impose different levels of difficulty on similar surface-task descriptions. Improved results may reflect stronger reasoning, or may stem from more

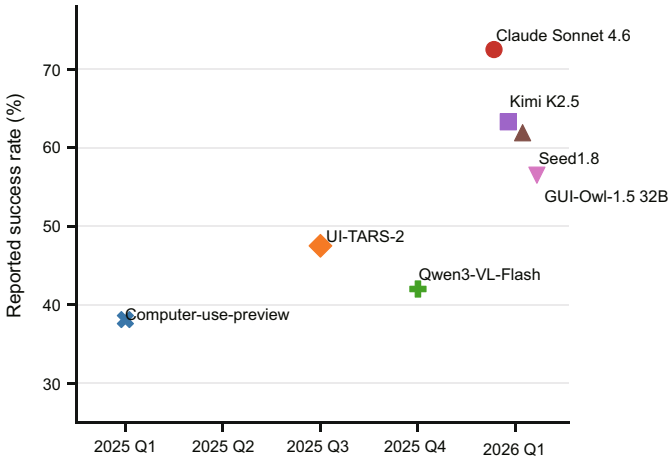

Fig. 5 Publicly reported results on OSWorld-family benchmarks, illustrating the interpretive challenge of cross-system comparison. The data points are sourced from official releases and represent different evaluation contracts. This plot is a snapshot of system-level outcomes under varied conditions, not a unified leaderboard of model capabilities. Detailed provenance of the plotted data points, including approximate readings when exact public values are not disclosed, is provided in Table 4

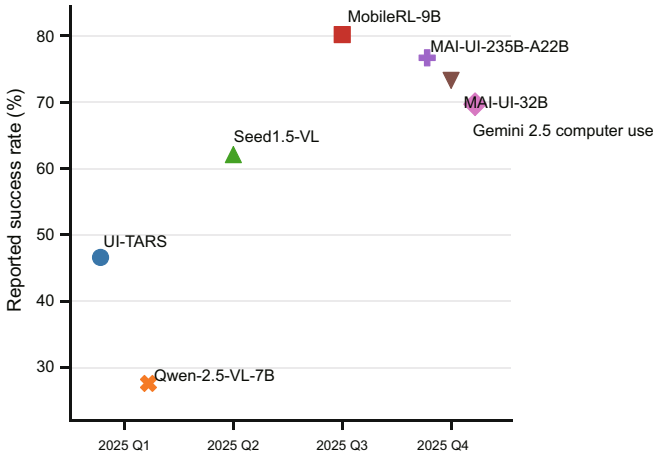

Fig. 6 Publicly reported performance trends on AndroidWorld-family benchmarks. This benchmark-conditioned snapshot shows the reported success rates of selected models over time. The upward trend suggests rapid progress in mobile GUI agent capabilities, while the spread of results within each quarter shows system-level differences (e.g., model scale, training paradigm, and access to privileged observations) that complicate direct comparisons. Like the data in Fig. 5, these data should be interpreted with reference to the evaluation conditions and system design choices. Detailed provenance of the plotted data points, including approximate readings when the exact public values are not disclosed, is provided in Table 5

Table 4 Data provenance for OSWorld-family trend points

Point	Score (%)	Benchmark/Metric	Observation/Action	Evaluation condition	Source
OpenAI CUA/computer-use-preview	38.1	OSWorld/Success	Screenshots; mouse–keyboard actions	Official evaluation; human intervention not reported	OpenAI CUA Team (2025)
Qwen3-VL-Flash	∼42	OSWorld-Verified/Success	Visual GUI observation	Public model report; exact contract not fully disclosed	Bai S et al. (2025b)
UI-TARS-2	47.5	OSWorld/Success	Screenshot-centric GUI actions	Report-specific evaluation; SDK variants separated	Wang HM et al. (2025)
GUI-Owl-1.5 32B	56.5	OSWorld-Verified/Success	Screenshots; native actions	Report-specific evaluation; MCP/tool variants separated	Xu HY et al. (2026)
Seed1.8	61.9	OSWorld/Success	Multimodal computer-use observation	Model-card report; plotted in 2026 Q1	ByteDance Seed (2026)
Kimi K2.5	63.3	OSWorld-Verified/Success	Visual observation; GUI actions	Report-specific evaluation; no human intervention reported	Bai TT et al. (2026)
Claude Sonnet 4.6	72.5	OSWorld-Verified/Success	Computer-use interface	System-card evaluation; no human intervention reported	Anthropic Safety Team (2026)

Values are public reports or approximate readings from the plotted trend when exact public values are not disclosed; scores are not normalized across evaluation contracts

Table 5 Data provenance for AndroidWorld-family trend points

Point	Score (%)	Benchmark/Metric	Observation/Action	Evaluation condition	Source
Qwen-2.5-VL-7B	27.6	AndroidWorld/Success	Mobile screenshots; touch actions	Baseline VLM execution reported in MobileRL of Table 1; Qwen2.5-VL documents to report the underlying model	Bai S et al. (2025a); Rawles et al. (2025); Xu YF et al. (2025b)
UI-TARS	∼46.6	AndroidWorld/Success	Screenshot-centric mobile actions	Reported agent evaluation; no human intervention reported	Qin et al. (2025)
Seed1.5-VL	62.1	AndroidWorld/Success	Mobile visual observation	Public VLM report; exact GUI-agent contract not fully disclosed	Guo D et al. (2025)
Gemini 2.5 computer use	69.7	AndroidWorld/Success	Computer-use API actions	Google evaluation/ browserbase-style reporting; AndroidWorld setup documented by Google	Google DeepMind Computer Use Team (2025)
MAI-UI-32B	73.3	AndroidWorld/Success	Screenshots; native GUI actions	Report-specific evaluation; no human intervention reported	Zhou HZ et al. (2025)
MAI-UI-235B-A22B	76.7	AndroidWorld/Success	Screenshots; zoom-in; MCP routing	Report-specific evaluation; no human intervention reported	Zhou HZ et al. (2025)
MobileRL-9B	80.2	AndroidWorld/Success	Mobile GUI observation; RL-enhanced execution	Public MobileRL report; AndroidWorld benchmark follows Rawles et al. (2025)	Rawles et al. (2025); Xu YF et al. (2025b)

AndroidWorld benchmark definition follows Rawles et al. (2025). Values are public reports, leaderboard entries, or approximate readings from the plotted trend when exact public values are not disclosed; scores are not normalized across evaluation contracts

favorable observation contracts or more reliable evaluator support. Evaluator audits such as WebArena Verified reveal the confounding nature of these outcomes (El Hattami et al., 2025).

Strong performance on AgentBench, GAIA, Shopping MMLU, MMT-Bench, ScienceBoard, or CUARewardBench may indicate important advances in general reasoning, shopping assistance, scientific workflow control, or reward-model quality, but does not clarify the contributions from GUI grounding, tool availability, or evaluator design (Jin et al., 2024; Liu X et al., 2024a; Mialon et al., 2024; Ying et al., 2024; Lin HJ et al., 2025; Sun QS et al., 2026). Evaluation frameworks such as GAL query not merely the success of an agent in contemporary tasks, but also the types of software interactions to which an agent can be generalized (Feng and Chen, 2026).

9.2 Efficiency, latency, and recovery burden

An agent that eventually completes a task after excessively many steps, incurs excessive latency, or repeatedly explores avoidable failure states is impractical. Highlighting this problem, OSWorld-Human compares the task completion, efficiency, and interaction overhead of agents with human performance (Abhyankar et al., 2025). Similar concerns

are increasingly relevant in mobile settings, where usability is shaped by latency and resource limitations, and in web settings, where repeated retries can mask the fragility behind nominal completion. Action economy, time cost, and recovery burden should be included in the evaluation rather than treated as secondary engineering details.

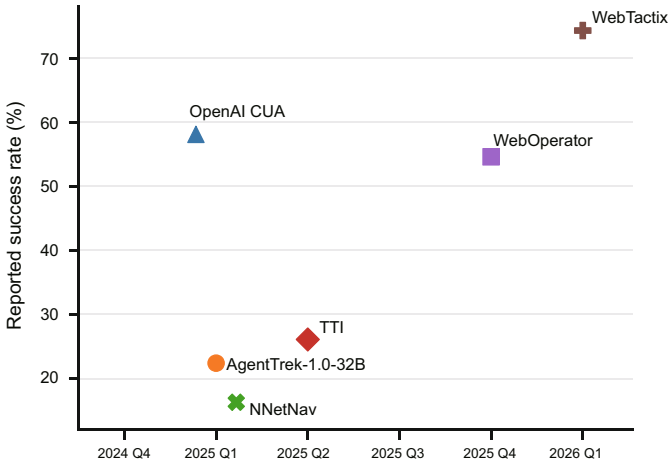

Fig. 7 Publicly reported performance trends on WebArena-family benchmarks. This benchmark-conditioned snapshot tracks the reported success rates of selected web agents over time. The upward trend suggests rapid capability improvement in the web-agent domain. The results of different models are separated, showing that raw scores combine advances in base models, specialized training, observation privilege, and evaluation protocol. Like the data in Figs. 5 and 6, these trends should be interpreted with explicit reference to the underlying benchmark conditions and system design choices. Detailed provenance of the plotted data points is provided in Table 6

9.3 Security, adversarial content, and human oversight

Trustworthiness is not judged merely by a benchmark score, but by the safety, robustness, and predictability of an agent during deployment. Recent survey work has indicated that GUI agents should be assessed for security, reliability, transparency, and governance along with task performance (Shi YC et al., 2025). One problem is adversarial or untrusted interface content. On the web, prompt injection embedded directly on the page content can subvert agent behavior (Wang XL et al., 2025). Multimodal robustness studies suggest that environment-provided content can systematically distort action selection or derail multistep execution (Wu CH et al., 2025). The agent must separate user intent, system policy, and environment content, which would otherwise be treated as an undifferentiated input stream.

A second challenge is the irreversibility of actions in many real-world tasks, such as sending a message, submitting a form, overwriting a file, changing permissions, or invoking an external tool with side effects. In these settings, uninterrupted autonomy should be replaced with explicit policies for escalation, rollback, user confirmation, and evidence display that ensure dependable deployment (Cheng PZ et al., 2025). Trustworthiness cannot be appended after the fact, as it shapes the core design of what the agent does and when it pauses.

Recent benchmark and red-teaming studies have reported similar risks across substrates. PopupAttack, AdvAgent, EIA, WIPI, RefusalBrowser, ST-WebAgentBench, Active Environmental Injection, and RedTeamCUA evaluate browser or

Table 6 Data provenance for WebArena-family trend points

Point	Score (%)	Benchmark/Metric	Observation/Action	Evaluation condition	Source
NNetNav	16.3	WebArena/Success	Browser observations from demonstrations	Benchmark evaluator; score also listed in the public WebArena leaderboard sheet	Murty et al. (2024); X-WebArena Leaderboard Maintainers (2026)
AgentTrek-1.0-32B	22.4	WebArena/Success	Web observations; tutorial-guided trajectories	Reported evaluation; score also listed in the public WebArena leaderboard sheet	Xu YH et al. (2025a); X-WebArena Leaderboard Maintainers (2026)
TTI	26.1	WebArena/Success	Browser observation; test-time interaction scaling	RL/Interaction scaling evaluation; score also listed in the public WebArena leaderboard sheet	Shen JH et al. (2025); X-WebArena Leaderboard Maintainers (2026)
WebOperator	54.6	WebArena/Success	Browser observation; tree search	Search/Backtracking allowed; score also listed in the public WebArena leaderboard sheet	Dihan et al. (2025); X-WebArena Leaderboard Maintainers (2026)
OpenAI CUA	58.1	WebArena/Success	Browser pixels; mouse–keyboard actions	Official evaluation; score also listed in the public WebArena leaderboard sheet	OpenAI CUA Team (2025); X-WebArena Leaderboard Maintainers (2026)
WebTactix	74.3	WebArena/Success	Web observation; multi-agent planning	Public leaderboard entry; exact evaluation contract follows leaderboard metadata	Steel.dev (2026)

Values are publicly reported success rates and are not normalized across evaluation contracts

hybrid web–OS attacks; MobileSafetyBench and the Security Matrix line extend the security concern to device-control settings (Kumar et al., 2024; Wu FZ et al., 2024; Yang YL et al., 2024; Chen YR et al., 2025; Liao et al., 2025; Xu CJ et al., 2025; Zhang YZ et al., 2025; Lee et al., 2026; Levy et al., 2026; Liao et al., 2026). Environmental distraction studies have highlighted the security problem from a wider multimodal perspective (Ma et al., 2025).

10 Applications, deployment, and industrial trajectories

GUI agents are appealing because they can automate a wide range of tasks without requiring a clean API exposure from every service. In the near term, GUI agents will most likely be deployed in routine browser workflows, enterprise knowledge work, lightweight mobile assistance, and desktop tasks that are sufficiently frequent to justify semi-automation yet sufficiently structured to support oversight. These applications are foreshadowed by the existing benchmark suites.

For instance, browser environments capture shopping, information retrieval, and portal workflows, WorkBench captures workplace tasks, and AndroidWorld and AppAgent-style systems capture mobile-use cases such as navigation through consumer apps or settings panels (Yao et al., 2022; Drouin et al., 2024; Styles et al., 2024; Rawles et al., 2025; Zhang C et al., 2025).

However, deployment cannot be assessed solely in terms of raw success rates. Organizations must understand the observation scope of an agent, the data stored by the agent, when the agent requests confirmation, evidence of completion by the agent, and how the agent responds to failure. This knowledge is especially important in regulated domains such as finance, healthcare, legal work, and enterprise software, where the GUI alone may visualize a stateful and governed process. In such settings, a human-compatible audit trail and explicit postcondition are often as important as task success.

A more realistic near-term deployment scenario is staged autonomy, in which agents can freely act on low-risk subtasks but must seek confirmation before irreversible operations, provide surface evidence when completion remains ambiguous, and invoke tools when GUI-only interaction is unnecessarily fragile. Recent systems have increasingly advanced from GUI mimicry toward dependable execution. Candidate deployment domains already span browser-based commerce and customer-service portals, spreadsheet and document editing, software testing, accessibility support, mobile-settings navigation, and cross-application workplace routines (Deng et al., 2023; Drouin et al., 2024; Gao DF et al., 2024b; Xie TB et al., 2024b; Zhang CY et al., 2025b). Resources such as macOSWorld reflect the growing interest in multilingual and internationalized settings, suggesting an increasing need for localization and language variation alongside application heterogeneity in evaluations (Yang P et al., 2025).

The application, deployment, and industrial strands of the literature are most informative when read together. Most relevant are the usage scopes of GUI agents and the deployment envelope, risk tolerance, and product pathway of each class of system.

10.1 Deployment archetypes and enterprise pathways

Current systems can be grouped into different deployment archetypes. Browser-bounded assistants typically operate within a website or a controlled enterprise portal with clean structural exposure and rollback conditions. Mobile personal assistants must handle user-specific context, privacy, and device-side constraints. Open-computer-use agents traverse browsers, office tools, file systems, communication tools, and external utilities. These archetypes can be imperfectly but usefully mapped onto the web, mobile, and desktop/OS regimes.

Some technically impressive systems are not easily deployed. A model that excels on screenshot-centric desktop benchmarks may incur excessive latency, lack rollback discipline, or be unable to decide when a task should be handed to a script or API. Conversely, a system with moderate benchmark performance may reliably operate within a narrow, well-defined envelope such as expense filing, browser-based record lookup, or guided customer-service workflows. Deployment readiness is not equivalent to benchmark leadership; rather, it depends on whether the operating regime, tool substrate, and risk profile match the intended use case.

Enterprise pathways also shape the evaluation design. Controlled browser portals, kiosk-like interfaces, and internal line-of-business systems often exhibit more stable layouts and stronger structural affordances than open consumer software. These environments may be suitable for semistructured hybrid agents that combine GUI actions with APIs, scripts, or rule-based checks. By contrast, consumer-facing and cross-application personal-assistant scenarios must strongly resist interruptions, UI drift, and ambiguity. The distinction between these two scenarios is increasingly reflected in the literature: benchmark ecosystems now coexist with investigations on tool invocation, mixed execution, and confidence-aware escalation, suggesting that practical adoption will not be mediated by a general-purpose agent, but will proceed through staged deployment envelopes (Abhyankar et al., 2025; Bonatti et al., 2025; Cheng PZ et al., 2025; Jia et al., 2026).

This staged deployment picture is reinforced by tool-oriented and application-facing systems. As shown in Turn Every Application into an Agent, Commands to Prompts, AssistEditor, and Magentic-UI, GUI control can be mixed with APIs, file abstractions, specialized agents, or explicit human participation, providing more useful automation than end-to-end autonomy alone (Gao DF et al., 2024a; Lu JT et al., 2024; Mozannar et al., 2025; Shi ZR et al., 2025).

10.2 Industrial systems and product trajectories

Fig. 8 summarizes these industrial trajectories, focusing on staged autonomy, explicit confirmation before irreversible actions, and mixed GUI–tool execution. To make the classification criteria explicit, the figure groups systems by their role in GUI-agent deployment: browser or computer-use products, evaluation and deployment infrastructure, GUI-specific or cross-platform open models, and safety or oversight mechanisms. General-purpose foundation models are treated as substrates only when public documentation directly connects them to computer-use, GUI interaction, tool orchestration, or

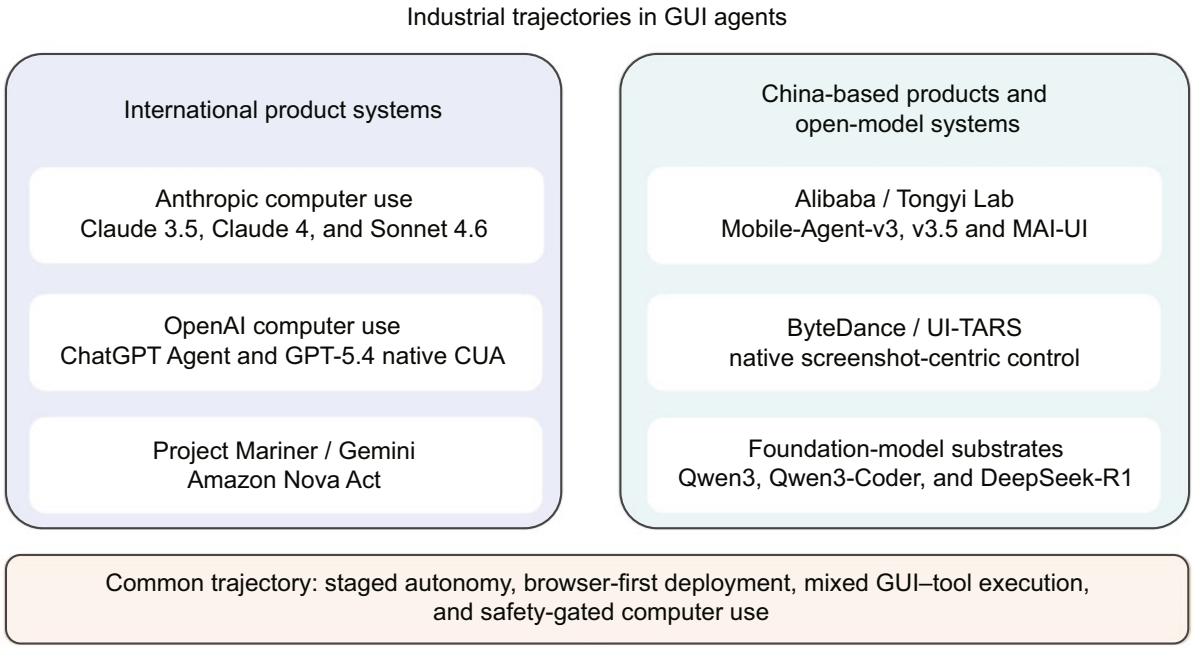

Fig. 8 Industrial trajectories of GUI agents, distinguishing among browser/computer-use products, evaluation and deployment infrastructures, GUI-specific or cross-platform open models, and safety or oversight mechanisms. Consistent with product-facing documentation and system cards, this figure emphasizes the importance of staged autonomy, explicit confirmation before irreversible actions, and mixed GUI–tool execution over unconstrained end-to-end control. General-purpose foundation models are included only when directly tied to public computer systems

agentic workflow deployment.

Industrial work complements the literature by showing which parts of the GUI agent stack are ready for real-user exposure and which are still gated by safety or reliability concerns. International product work has recently converged on browser and computer-use agents. Anthropic introduced "computer use" with Claude 3.5 Sonnet in late 2024, describing it as screen perception plus cursor-and-keyboard actuation under human oversight. An accompanying research note described computer use as both a capability milestone and a new safety problem (Anthropic Computer Use Team, 2024; Anthropic Research Team, 2024). OpenAI followed with Operator in early 2025, exposing a browser agent powered by its computer-using agent (CUA) model and documenting the release through both product materials and a dedicated system card (OpenAI CUA Team, 2025; OpenAI Operator System Card Team, 2025; OpenAI Operator Team, 2025). By mid-2025, however, OpenAI had shifted the product surface from standalone Operator toward the wider ChatGPT Agent experience, so staged integration became part of the product strategy (OpenAI ChatGPT Agent Team, 2025; OpenAI Operator Team, 2025). Google positioned browser-use agents within the Gemini 2.0 "agentic era" roadmap, highlighting them as one component of a wider family of action-taking systems (Google Gemini 2.0 Team, 2024). Subsequently, Nova Act was productized for browser-based applications, explicitly combining UI interaction with code and tool steps under supervisory controls (Amazon Web Services, 2025). These systems indicate a shared industrial preference for browser and browser-adjacent automation as early deployment targets, partly because the execution substrate is constrained and human confirmation is more easily inserted at appropriate process nodes.

Research organizations at large companies have also launched evaluation infrastructures and deployable open models. For instance, Windows Agent Arena by Microsoft translates the OSWorld spirit into a scalable Windows-specific benchmark, demonstrating that evaluation speed, reproducibility, and environment control are industrial bottlenecks (Bonatti et al., 2025). Apple's Ferret-UI line (Ferret-UI 2 and Ferret-UI Lite) emphasizes cross-platform UI understanding and compact on-device GUI agents, reflecting a deployment perspective in which latency, privacy, and device efficiency must be considered alongside benchmark success (Li ZH et al., 2025; Yang Z et al., 2026). Google's ScreenAI highlights the value of UI-specialized perception backbones that treat documents, screens, and visual-text interfaces as a shared multimodal parsing problem (Baechler et al., 2024). In summary, several major laboratories are investing in end-user agents, measurement, grounding, and compact-model infrastructure for dependable and scalable deployment.

Public China-based industrial releases have emphasized different priorities. Alibaba-affiliated efforts have pushed progress toward foundation-style GUI agents with large-scale data generation, online RL, and device–cloud collaboration. MobileAgent-v3 frames a general-purpose GUI agent framework, and its associated GUI-Owl model line as a scalable automation infrastructure spanning desktop and mobile benchmarks (Ye et al., 2025). Mobile-Agent-v3.5 has extended this infrastructure to multi-platform execution, grounding, tool calling, and memory-oriented tasks (Xu HY et al., 2026). MAI-UI combines native agent–user interaction, MCP-style tool calls, online RL, and device–cloud routing into a single technical report, mainly focused on the deployment architecture (Zhou HZ et al., 2025). Complementarily, UI-TARS and UI-TARS-2 of ByteDance are native, screenshot-centric GUI agents with shared action modeling, strong system-2 reasoning,

and multi-turn RL (Qin et al., 2025; Wang HM et al., 2025). Publicly visible China-based efforts have been particularly active in GUI-agent-relevant open-model releases, pure-vision interfaces, scalable data pipelines, and mobile-first or cross-platform automation.

China-based organizations have developed product-adjacent materials beyond benchmarks. AutoGLM kickstarted the trajectory of autonomous foundation agents for GUIs (Liu X et al., 2024b), whereas GLM-4.5V and GLM-4.1V-Thinking clarified the positioning of multimodal reasoning models as substrates for future agentic computer use (Hong et al., 2025).

Product-facing computer-use systems are also being updated through both system-card revisions and traditional benchmarks. A representative example is OpenAI's addendum for the o3 Operator (OpenAI o3 Operator Team, 2025).

Another pattern is the fast upgrade of frontier multimodal models serving as substrates of GUI agents. OpenAI's public trajectory from GPT-5 to GPT-5.4 frames computer use, tool orchestration, and long-context professional work as model capabilities (OpenAI Developer Platform Team, 2025; OpenAI GPT-5 Team, 2025; OpenAI GPT-5.4 Team, 2026). GPT-5.4 is particularly relevant to GUI-agent evaluations because it can be explicitly connected to computers and deployment-oriented multimodal workflows through OpenAI's public materials. Subsequent GPT-5.4 mini- and nano-releases have extended this positioning to lighter deployment tiers (OpenAI GPT-5.4 Mini and Nano Team, 2026). Similarly, Google has progressed from Gemini 2.5 Pro to Gemini 3.1 Pro and Gemini 3.1 (Google Gemini 2.5 Team, 2025; Google Gemini 3 Team, 2025). Flash-Lite explicitly spans both deep reasoning and scaled deployment (Google Gemini 3.1 Flash-Lite Team, 2026; Google Gemini 3.1 Pro Team, 2026). The public model line of Anthropic has advanced from Claude 3.7 Sonnet to Claude 4 and Claude Sonnet 4.6. These models deliver hybrid reasoning, tool use during extended thinking, and long-running agent workflows as core model capabilities (Anthropic Claude 3.7 Team, 2025; Anthropic Claude 4 Team, 2025; Anthropic Claude Sonnet 4.6 Team, 2026; Anthropic Documentation Team, 2026). Qwen3 and Qwen3-Coder, two publicly released China-based model lines, show how open-weight ecosystems have aligned frontier reasoning with agentic coding and browser/tool use (Qwen3 Team, 2025; Qwen3-Coder Team, 2025), whereas Qwen3Guard and DeepSeek-R1 indicate the integration of safety instrumentation and reinforcement-trained reasoning into the same frontier stack (Guo DY et al., 2025; Qwen3Guard Team, 2025).

Improved base models have raised the ceiling of screenshot understanding, grounding, long-horizon tool use, and software-environment reasoning. Accordingly, downstream GUI-agent gains are increasingly tied to the capabilities of the underlying foundation model. Product lines such as GPT-5.4, Gemini 3.1 Pro, Claude Sonnet 4.6, and Qwen3-Coder provide tools, coding, and agent workflows, suggesting that future GUI agents will not be limited to GUI clicking alone, but will combine multimodal reasoning, browser/desktop control, tool connectors, and execution sandboxes. Evaluations have also been affected by the pace of industrial release: as major laboratories are updating their base models every few months, benchmark papers that treat the model as fixed may underestimate the rate at which the deployment frontier is shifting. For this reason, the current survey of GUI agents examines benchmark papers, open-source systems, and product and safety documents released by major model providers.

In these public industrial technical reports, deployment priorities are more sharply differentiated than those sometimes presented in the literature. International product efforts have focused on browser assistance, staged autonomy, enterprise workflow automation, and safety gating during real-user deployment. Company-affiliated research groups in China and elsewhere have prioritized scalable evaluation, pure vision grounding, compact on-device agents, and RL-based generalization. Industrial work is converging not on monolithic systems that can be deployed on any computer, but on deployment pathways with different regimes, risk profiles, and product-surface technical bottlenecks.

11 Cross-cutting challenges

All three operational regimes face common challenges. The first challenge is data mismatch. Although static traces, synthetic trajectories, and highly instrumented environments are useful, they only weakly approximate the combination of hidden states, user variations, long horizons, and recovery requirements in real digital work. This problem is especially prominent when training data and evaluation uses are collected under different observation contracts. A model trained on structure-rich traces may underperform in screenshot-only environments, whereas a screenshot-trained model with strong visual transfer may remain weak at postcondition reasoning or tool use.

The second bottleneck is benchmark infrastructure. The field increasingly depends on evaluators, simulators, wrappers, and tool adapters that are themselves complex software systems. As demonstrated in BrowserGym, WebArena Verified, AndroidWorld, and OSWorld, evaluation infrastructure is not passive background machinery, but directly shapes the meanings of scores (Xie TB et al., 2024b; de Chezelles et al., 2025; El Hattami et al., 2025; Rawles et al., 2025). If the evaluator checks only the visible UI state, models may be overfitted to superficial completion. If the environment exposes a privileged structure or success signals, scores cannot be easily compared with those of screenshot-only systems. Progress requires a clearer separation of what is measured, what is exposed, and what is presumed. Operationally, benchmark reports should disclose observation contracts, evaluator privileges, tool access, retry budgets, and postcondition checks alongside the headline scores.

The third problem is interface heterogeneity. The custom widgets, legacy software, browser-specific components, variable-quality accessibility layers, and application-specific shortcuts of GUI agents are not cleanly generalizable, especially in desktop and enterprise settings but also in mobile applications with local gesture semantics and inconsistent app conventions. This problem is not necessarily resolvable by model improvement, but may also require cleaner agent–computer interfaces, stronger tool abstractions, and interface layers designed for machine and human use.

11.1 Data and supervision gaps

Current supervision signals remain uneven across platforms. The web hosts comparatively rich task data, structured observations, and language-grounding benchmarks with a long history. Mobile settings benefit from large demonstration datasets but lack standardized multiapp and privacy-aware evaluations. Desktop and OS settings offer the most realistic picture of computer use but are harder to scale than web and mobile settings because successful labels, recovery events, and tool decisions are expensive to annotate or simulate. This imbalance shapes the easily optimizable capabilities. Data for action prediction are far more numerous than data for verification, recovery, escalation, or rollback.

One consequence of this imbalance is a commonly inherited mismatch between what agents are trained to imitate and what they must execute during deployment. For example, demonstrations can show how a human clicks through a workflow, but may not reveal the hidden states that are mentally checked by the human before continuing to the next step. Similarly, trajectory datasets can expose local action patterns while obscuring the evidence chain that justifies those actions. Richer supervision for verification, uncertainty estimation, and recovery is a priority.

11.2 Toward agent-ready interfaces

Some tasks are rendered difficult not by a complex underlying operation but by inefficiency of the GUI substrate. This problem is well clarified in desktop and enterprise settings, where a task may be equally well expressed as a sequence of clicks, a script, an application-specific macro, or a structured tool call. OSWorld-MCP and other hybrid systems provide agent-ready interface layers that remain human-usable but expose a more reliable interaction contract for autonomous systems (Hu et al., 2025; Jia et al., 2026; Sager et al., 2026).

Toward an agent-ready interface, the GUI perspective should not be abandoned but GUI control should be treated as part of a larger interaction stack in which the agent may choose among multiple substrates. In many cases, the most important decision is not clicking on the exact coordinate, but whether clicking is the appropriate substrate. Research on agent-ready interfaces, tool routing, and structured confirmation policies may be as important as further improvements in multimodal perception.

11.3 Human factors and deployment realism

GUI agent research often neglects the human factors in deployment. Real users are concerned with latency, confidence, transparency, undoability, and privacy. Users perceive the system not as a benchmark score but as a process that may or may not feel predictable and controllable. For this reason, human oversight should not be framed as a fallback in low-accuracy cases, but as part of the user–agent interface contract in practical settings.

Deployment design should focus not only on enhancing the accuracy of the agent, but on the staging of autonomy. In some workflows, the agent may act freely until it reaches a risky step; in others, it may be better used as a verification assistant, a workflow suggester, or a tool-routing layer rather than as a fully autonomous executor. Careful study of these human factors would help with connecting benchmark-driven progress to real deployment quality. Interface-quality and critiquing resources such as UICrit have suggested that agents may eventually need to explain, compare, or critique interface choices for users, broadening the human-factor agenda beyond accuracy and latency alone (Duan et al., 2024).

12 Staged research agenda

Early progress in GUI agents was driven by multimodal perception and the need for stronger action generation. Those advances remain important, but the main deployment bottlenecks have shifted. Failures across web, mobile, and desktop/OS settings remain common because the agents (even when generating plausible local actions) cannot reliably maintain the state, verify outcomes, manage execution cost, or recover from errors under realistic uncertainty. As an operational agenda, Table 7 summarizes the main bottlenecks, candidate methods, and required evidence of progress assessments.

12.1 State-aware execution

Next-action prediction is a useful starting point, but is insufficient for dependable automation. Most prior work has evaluated the plausibility of the next click, tap, swipe, or keystroke under the current observation. In realistic settings, what matters is not only whether the next action appears plausible, but whether it produces the intended state change. For this purpose, agents must maintain a working belief of the latent state, update that belief after execution, and seek additional evidence when visible interface changes cannot establish correctness.

Along this direction, WebArena Verified strengthens web-agent evaluation through explicit post hoc completion checks (El Hattami et al., 2025). WorkBench emphasizes outcome-centric evaluation in realistic workplace tasks (Styles et al., 2024). OS-Kairos studies adaptive interaction and confidence-aware intervention under uncertainty (Cheng PZ et al., 2025). In these studies, postcondition verification and uncertainty

Table 7 A staged research agenda for dependable GUI agents

Open problem	Main bottleneck	Candidate method	Evidence of progress
State-aware execution	Hidden state and false local success	Belief tracking; postcondition checks; runtime monitoring; recovery	Outcome evidence; calibrated uncertainty; lower false-success rate
Adaptive sensing	Costly or incomplete observations	Selective zooming; structure requests; tool queries; evidence thresholds	Fewer observations; lower latency; stable success rate
Hybrid automation	Fragile GUI-only execution and side effects	GUI–tool arbitration; APIs/scripts; sandboxing; rollback; confirmation	Fewer irreversible errors; clearer audit trails; safer recovery

management are treated not as external checks but as part of the agent loop.

12.2 Adaptive sensing

Agents in many environments should not process every available signal with the same intensity at every step. High-resolution screenshots, accessibility trees, browser structure, tool outputs, and historical context all incur costs. Recent grounding works on ScreenSpot-Pro (Li KX et al., 2025), Ferret-UI 2 (Li ZH et al., 2025), and ShowUI (Lin KQ et al., 2025) have shown that perception quality remains a bottleneck in dense or professional interfaces. Research should probe the accuracy to which agents perceive and allocate sensing efforts, including when to zoom in, when to request structural signals, when to switch the execution substrate, and when visible evidence is insufficient to justify a further action.

Adaptive sensing is important because it connects capability to efficiency. A system that succeeds only by repeatedly querying large contexts or reflecting upon every step may score well in a benchmark, but may remain unusable in practice. Improving sensing policies, tightening evidence thresholds, and ensuring deliberate information acquisition will likely become as important as increasing the size of base models.

12.3 Hybrid automation

A large body of GUI-agent research has investigated direct interface manipulation through clicks, gestures, typing, and scrolling. Although such research is a natural starting point, many practical tasks can be handled in multiple ways, for instance, as a sequence of GUI actions or as a more structured combination of interface operations, scripts, tool calls, and application-specific procedures. Accordingly, OmniACT explores executable program generation for multimodal agents (Kapoor et al., 2024). AutoDroid-V2 reformulates parts of a mobile automation through code generation (Wen et al., 2025). OSWorld-MCP evaluates external tool invocation within computer-use agents (Jia et al., 2026). Future GUI agents will need to arbitrate among multiple execution substrates instead of relying on GUI manipulation alone.

To meet this need, the agent must decide when to use direct GUI interaction or invoke a tool or script. Other pertinent questions are: What interface information (beyond pixels) should be exposed in the environment? How should the agent backtrack, re-plan, or escalate when multiple execution substrates are available and one substrate fails? These questions are especially important in desktop and OS settings, where multiapplication workflows and irreversible side effects are common and tool choice becomes part of the core planning problem.

13 Representative systems and literature synthesis

A map of representative systems and milestones will better benefit the literature than a summary of concepts. Tables 8 and 9 list the representative GUI-agent milestones in

Table 8 Representative GUI-agent milestones (2017–2024)

Work	Regime	Historical role
World of Bits (Shi TL et al., 2017)	Web	Sequential web-control platform
AndroidEnv (Toyama et al., 2021)	Mobile	Android control as an RL environment
Pix2Act (Shaw et al., 2023)	Cross-platform	Pixel-to-action GUI formulation
Mind2Web (Deng et al., 2023)	Web	Real-website data for web agents
AppAgent (Zhang C et al., 2025)	Mobile	Exploration-driven smartphone agent
CogAgent (Hong et al., 2024)	Cross-platform	VLM backbone for GUI understanding
WebVoyager (He et al., 2024)	Web	Screenshot-centric browser agent
SeeClick (Cheng KZ et al., 2024)	Cross-platform	GUI grounding as a core subproblem
AppAgent v2 (Li YD et al., 2024)	Mobile	Reusable mobile interaction knowledge
Mobile-Agent (Wang JY et al., 2024)	Mobile	Vision-first mobile control
OmniACT (Kapoor et al., 2024)	Desktop/Web	GUI acting plus program execution

Table 9 Representative GUI-agent milestones (2025–2026)

Work	Regime	Historical role
UFO (Zhang CY et al., 2025b)	Desktop/OS	Executable Windows agent
Agent S (Agashe et al., 2025)	Desktop/OS	Long-horizon computer use with experience reuse
ShowUI (Lin KQ et al., 2025)	Cross-platform	GUI vision-language-action backbone
Ferret-UI 2 (Li ZH et al., 2025)	Cross-platform	Cross-platform UI representation
UI-TARS (Qin et al., 2025)	Desktop/Mobile	Native GUI action modeling
Mobile-Agent-v3 (Ye et al., 2025)	Mobile/Desktop	Data generation plus environment RL
OS-Kairos (Cheng PZ et al., 2025)	Desktop/OS	Confidence-aware recovery timing
Claude computer use (Anthropic Computer Use Team, 2024; Anthropic Claude Sonnet 4.6 Team, 2026)	Browser/Desktop	Bounded screen and cursor-keyboard control
ChatGPT Agent/CUA (OpenAI CUA Team, 2025; OpenAI ChatGPT Agent Team, 2025)	Browser-first	Browser action, tools, code, and computer use
Google browser-use system (Google Gemini 2.0 Team, 2024; Google Gemini 3 Team, 2025)	Browser-first	Browser-use trajectory in Gemini roadmap
Nova Act (Amazon Web Services, 2025)	Enterprise browser	Productivity-workflow automation

chronological order, not by their current popularity. The lists span formative precursors, the first generalist visual-control wave, current cross-platform agents, and productized frontier systems. The label "GUI agent" now covers very different technical objects—browser environments, mobile-control agents, desktop assistants, grounding backbones, and staged industrial products—and the action substrate has similarly diversified. The current literature has moved from improving click points to changing what is observable, what action channels are available, and what counts as verifiable completion.

Tables 8 and 9 show the main historical shifts. Early work on environments and low-level control abstractions was followed by insights into realistic traces, screenshot understanding, and wider multimodal perception. Current work has moved to cross-platform grounding, generalist computer-use agents, stronger evaluator design, mixed GUI–tool execution, and industrial deployment pathways that explicitly constrain autonomy. The literature corpus is both expanding and moving from local interface fluency to evidence-backed execution over heterogeneous digital workflows.

The literature can also be grouped by the type of hidden state in each setting. Web papers are most informative when they clarify the gap between the visible surface and backend state using WebArena, WebArena Verified, BrowserGym, and AgentOccam. Mobile papers are most informative when they expose gesture ambiguity, personalization, and dynamic device context using Android in the Wild, AndroidWorld, MobileWorld, and MobileAgent-v3. Desktop and OS papers are most informative when they expose cross-window coordination, tool arbitration, and costly side effects, as in studies based on OSWorld, Windows Agent Arena, WorkBench, and OSWorld-Human. This view is more informative than a flat benchmark list because it explains why transfer across benchmarks is often weaker than suggested by raw scores.

A layered reading of the literature separates four types of contributions: survey and taxonomy papers that define terminology and scope, benchmark papers that clarify what is measured in different environments, grounding papers focused on cross-platform perception and UI understanding, and framework papers that integrate memory, planning, tool use, and verification into longer-horizon systems. This separation clarifies the distinction among benchmark innovation, perception innovation, and system-design innovation.

14 Conclusions

GUI agents have grown from a niche automation topic into a test domain for general-purpose digital intelligence. Recent progress in multimodal models, benchmark construction, and agent frameworks has expanded what can be measured and automated across web, mobile, and desktop/OS environments. These settings are not superficial interface variants of a single uniform problem.

GUI agents share a similar technical stack spanning perception, grounding, planning, execution, and verification, but impose correctness problems. The main problems are mismatches between the visible pages and backend transactional state in web environments, gesture-driven interactions under personalization, privacy, and resource constraints in mobile environments, and heterogeneous multiapplication workflows, high-resolution grounding, and costly side effects in desktop and OS environments. Benchmark scores should not be read as pure model numbers; rather, they are system-level outcomes shaped by observation privilege, tool access, evaluator fidelity, execution fidelity, and recovery cost.

Future GUI-agent research will not be defined by next-action prediction alone. Future systems must determine whether the world behind the interface has changed in the intended way, whether evidence is sufficient for safe continuation of the execution, and whether GUI interaction can be combined with tools and human oversight when direct manipulation alone is not dependable. As frontier multimodal and reasoning models, including GPT-5 and GPT-5.4, Gemini 3.1 Pro, Claude Sonnet 4.6, Qwen3, and DeepSeek-R1, are being released at rapid rates, GUI agent gains are increasingly tied to the integration, constraint, and evaluation of base models inside platform-specific control loops. The goal is not the imitation of visible interface behavior but the reliable and safe completion of real tasks when the relevant system state is only partially observable.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62372341).

Author contributions

Yu WU and Yi YANG contributed to the conception, organization, writing, and revision of the paper.

Conflict of interest

Yi YANG is an editorial board member and director of the junior editorial board of *ENGINEERING Information Technology & Electronic Engineering*; he was not involved with the peer review process of this paper. Both authors declare that they have no conflict of interest.

Declaration on the use of generative AI tools

During the preparation of this work, the authors used ChatGPT 5.5 to improve language. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the published article.

References

Abhyankar R, Qi Q, Zhang YY, 2025. OSWorld-Human: benchmarking the efficiency of computer-use agents. https://doi.org/10.48550/arXiv.2506.16042

Agashe S, Han JZ, Gan SY, et al., 2025. Agent S: an open agentic framework that uses computers like a human. Proc 13th Int Conf on Learning Representations, p.1-23.

Amazon Web Services, 2025. Build agents to automate production UI workflows with Amazon Nova Act (GA). https://aws.amazon.com/about-aws/whats-new/2025/12/build-automate-production-ui-workflows-nova-act [Accessed on Apr. 1, 2026].

Anthropic Claude 3.7 Team, 2025. Claude 3.7 Sonnet and Claude Code. https://www.anthropic.com/news/claude-3-7-sonnet [Accessed on Apr. 1, 2026].

Anthropic Claude 4 Team, 2025. Introducing Claude 4. https://www.anthropic.com/news/claude-4 [Accessed on Apr. 1, 2026].

Anthropic Claude Sonnet 4.6 Team, 2026. Introducing Claude Sonnet 4.6. https://www.anthropic.com/news/claude-sonnet-4-6 [Accessed on Apr. 1, 2026].

Anthropic Computer Use Team, 2024. Introducing computer use, a new Claude 3.5 Sonnet, and Claude 3.5 Haiku. https://www.anthropic.com/news/3-5-models-and-computer-use [Accessed on Apr. 1, 2026].

Anthropic Documentation Team, 2026. Models overview. https://docs.anthropic.com/en/docs/about-claude/models/overview [Accessed on Apr. 1, 2026].

Anthropic Research Team, 2024. Developing a computer use model. https://www.anthropic.com/news/developing-computer-use [Accessed on Apr. 1, 2026].

Anthropic Safety Team, 2026. Claude Sonnet 4.6 system card. https://www.anthropic.com/claude-sonnet-4-6-system-card [Accessed on Apr. 1, 2026].

Anupam S, Brown D, Li S, et al., 2025. BrowserArena: evaluating LLM agents on real-world web navigation tasks. Workshop on Multi-Turn Interactions in Large Language Models at NeurIPS, p.1-19.

Azam R, Abuelsaad T, Vempaty A, et al., 2024. Multimodal auto validation for self-refinement in web agents. Proc 38th Conf on Neural Information Processing Systems, Workshop on Open-World Agents, p.1-10.

Baechler G, Sunkara S, Wang M, et al., 2024. ScreenAI: a vision-language model for UI and infographics understanding. Proc 33rd Int Joint Conf on Artificial Intelligence, p.3058-3068. https://doi.org/10.24963/ijcai.2024/339

Bai H, Zhou YF, Cemri M, et al., 2024. DigiRL: training in-the-wild device-control agents with autonomous reinforcement learning. Proc 38th Int Conf on Neural Information Processing Systems, Article 397.

Bai S, Chen KQ, Liu XJ, et al., 2025a. Qwen2.5-VL technical report. https://doi.org/10.48550/arXiv.2502.13923

Bai S, Cai YX, Chen RZ, et al., 2025b. Qwen3-VL technical report. https://doi.org/10.48550/arXiv.2511.21631

Bai TT, Bai YF, Bao YP, et al., 2026. Kimi K2.5: visual agentic intelligence. https://doi.org/10.48550/arXiv.2602.02276

Berkovitch O, Caduri S, Kahlon N, et al., 2025. Identifying user goals from UI trajectories. Proc ACM on Web Conf, p.2381-2390. https://doi.org/10.1145/3701716.3717525.

Boisvert L, Thakkar M, Gasse M, et al., 2024. WorkArena++: towards compositional planning and reasoning-based common knowledge work tasks. Proc 38th Conf on Neural Information Processing Systems, Article 195.

Bommasani R, Hudson DA, Adeli E, et al., 2021. On the opportunities and risks of foundation models. https://arxiv.org/abs/2108.07258v1

Bonatti R, Zhao D, Bonacci F, et al., 2025. Windows Agent Arena: evaluating multi-modal OS agents at scale. Proc 42nd Int Conf on Machine Learning, p.4874-4910.

Brown TB, Mann B, Ryder N, et al., 2020. Language models are few-shot learners. Proc 34th Int Conf on Neural Information Processing Systems, Article 159.

Burns A, Arsan D, Agrawal S, et al., 2022. A dataset for interactive vision-language navigation with unknown command feasibility. Proc 17th European Conf on Computer Vision, p.312-328. https://doi.org/10.1007/978-3-031-20074-8_18

ByteDance Seed, 2026. Seed1.8 model card: towards generalized real-world agency. https://doi.org/10.48550/arXiv.2603.20633

Cao Y, Wang YY, Bu P, et al., 2025. AndroidLens: long-latency evaluation with nested sub-targets for Android GUI agents. https://doi.org/10.48550/arXiv.2512.21302

Chai YX, Huang SY, Niu YZ, et al., 2025. AMEX: Android multi-annotation expo dataset for mobile GUI agents. Findings of the Association for Computational Linguistics: ACL, p.2138-2156. https://doi.org/10.18653/v1/2025.findings-acl.110

Chen DP, Huang Y, Wu SY, et al., 2025. GUI-World: a video benchmark and dataset for multimodal GUI-oriented understanding. Proc 13th Int Conf on Learning Representations, p.1-53.

Chen Q, Pitawela D, Zhao CY, et al., 2024. WebVLN: vision-and-language navigation on websites. Proc 38th AAAI Conf on Artificial Intelligence, p.1165-1173. https://doi.org/10.1609/aaai.v38i2.27878

Chen WT, Cui JB, Hu JY, et al., 2025. GUICourse: from general vision language model to versatile GUI agent. Proc 63rd Annual Meeting of the Association for Computational Linguistics, p.21936-21959. https://doi.org/10.18653/v1/2025.acl-long.1065

Chen XT, Zhao ZH, Chen L, et al., 2021. WebSRC: a dataset for web-based structural reading comprehension. Proc Conf on Empirical Methods in Natural Language Processing, p.4173-4185. https://doi.org/10.18653/v1/2021.emnlp-main.343

Chen XT, Li HC, Liang JQ, et al., 2024. EDGE: enhanced grounded GUI understanding with enriched multi-granularity synthetic data. https://doi.org/10.48550/arXiv.2410.19461

Chen YR, Hu XY, Yin KT, et al., 2025. Evaluating the robustness of multimodal agents against active environmental injection attacks. Proc 33rd ACM Int Conf on Multimedia, p.11648-11656. https://doi.org/10.1145/3746027.3755646

Cheng KZ, Sun QS, Chu YG, et al., 2024. SeeClick: harnessing GUI grounding for advanced visual GUI agents. Proc 62nd Annual Meeting of the Association for Computational Linguistics, p.9313-9332. https://doi.org/10.18653/v1/2024.acl-long.505

Cheng PZ, Wu Z, Wu ZR, et al., 2025. OS-Kairos: adaptive interaction for MLLM-powered GUI agents. Findings of the Association for Computational Linguistics: ACL, p.6701-6725. https://doi.org/10.18653/v1/2025.findings-acl.348

de Chezelles TLS, Gasse M, Drouin A, et al., 2025. The BrowserGym ecosystem for web agent research. https://arxiv.org/abs/2412.05467

Deka B, Huang ZF, Franzen C, et al., 2017. Rico: a mobile app dataset for building data-driven design applications. Proc 30th Annual ACM Symp on User Interface Software and Technology, p.845-854. https://doi.org/10.1145/3126594.3126651

Deng X, Gu Y, Zheng BY, et al., 2023. Mind2Web: towards a generalist agent for the web. Proc 37th Int Conf on Neural Information Processing Systems, Article 1220.

Dihan ML, Hashem T, Ali ME, et al., 2025. WebOperator: action-aware tree search for autonomous agents in web environment. https://arxiv.org/abs/2512.12692

Dong LZ, Zhou ZQ, Yang SB, et al., 2025. Say one thing, do another? Diagnosing reasoning–execution gaps in VLM-powered mobile-use agents. https://arxiv.org/abs/2510.02204

Drouin A, Gasse M, Caccia M, et al., 2024. WorkArena: how capable are web agents at solving common knowledge work tasks? Proc 41st Int Conf on Machine Learning, p.11642-11662.

Du YQ, Konyushkova K, Denil M, et al., 2023. Vision-language models as success detectors. Proc 2nd Conf on Lifelong Learning Agents, p.120-136.

Duan PT, Cheng CY, Li G, et al., 2024. UICrit: enhancing automated design evaluation with a UI critique dataset. Proc 37th Annual ACM Symp on User Interface Software and Technology, Article 46. https://doi.org/10.1145/3654777.3676381

El Hattami A, Thakkar M, Chapados N, et al., 2025. WebArena Verified. Proc 39th Conf on Neural Information Processing Systems Workshop, p.1-28.

Fan Y, Ding L, Kuo CC, et al., 2024. Read anywhere pointed: layout-aware GUI screen reading with tree-of-lens grounding. Proc Conf on Empirical Methods in Natural Language Processing, p.9503-9522. https://doi.org/10.18653/v1/2024.emnlp-main.533

Feng SD, Chen CY, 2026. How smart is your GUI agent? A framework for the future of software interaction. https://doi.org/10.48550/arXiv.2602.11514

Furuta H, Matsuo Y, Faust A, et al., 2024. Exposing limitations of language model agents in sequential-task compositions on the web. https://arxiv.org/abs/2311.18751

Gao DF, Hu SY, Bai ZC, et al., 2024a. AssistEditor: multi-agent collaboration for GUI workflow automation in video creation. Proc 32nd ACM Int Conf on Multimedia, p.11255-11257. https://doi.org/10.1145/3664647.3684998

Gao DF, Ji L, Bai ZC, et al., 2024b. AssistGUI: task-oriented PC graphical user interface automation. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.13289-13298. https://doi.org/10.1109/CVPR52733.2024.01262

Gao LX, Zhang L, Wang SH, et al., 2024. MobileViews: a large-scale mobile GUI dataset. https://doi.org/10.48550/arXiv.2409.14337

Google DeepMind Computer Use Team, 2025. Introducing the Gemini 2.5 computer use model. https://blog.google/innovation-and-ai/models-and-research/google-deepmind/gemini-computer-use-model [Accessed on Apr. 1, 2026].

Google Gemini 2.0 Team, 2024. Introducing Gemini 2.0: our new AI model for the agentic era. https://blog.google/innovation-and-ai/models-and-research/google-deepmind/google-gemini-ai-update-december-2024 [Accessed on Apr. 1, 2026].

Google Gemini 2.5 Team, 2025. Gemini 2.5: our most intelligent AI model. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025 [Accessed on Apr. 1, 2026].

Google Gemini 3 Team, 2025. A new era of intelligence with Gemini 3. https://blog.google/products-and-platforms/products/gemini/gemini-3 [Accessed on Apr. 1, 2026].

Google Gemini 3.1 Flash-Lite Team, 2026. Gemini 3.1 flash-lite: built for intelligence at scale. https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-flash-lite [Accessed on Apr. 1, 2026].

Google Gemini 3.1 Pro Team, 2026. Gemini 3.1 Pro: a smarter model for your most complex tasks. https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-pro [Accessed on Apr. 1, 2026].

Gou BY, Wang RH, Zheng BY, et al., 2025. Navigating the digital world as humans do: universal visual grounding for GUI agents. Proc 13th Int Conf on Learning Representations, p.1-33.

Gu ZX, Zeng ZW, Xu ZY, et al., 2025. UI-Venus technical report: building high-performance UI agents with RFT. https://doi.org/10.48550/arXiv.2508.10833

Guo D, Wu FM, Zhu FD, et al., 2025. Seed1.5-VL technical report. https://doi.org/10.48550/arXiv.2505.07062

Guo DY, Yang DJ, Zhang HW, et al., 2025. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633-638. https://doi.org/10.1038/s41586-025-09422-z

Gur I, Nachum O, Miao YJ, et al., 2023. Understanding HTML with large language models. Findings of the Association for Computational Linguistics: EMNLP, p.2803-2821. https://doi.org/10.18653/v1/2023.findings-emnlp.185

He HL, Yao WL, Ma KX, et al., 2024. WebVoyager: building an end-to-end web agent with large multimodal models. Proc 62nd Annual Meeting of the Association for Computational Linguistics, p.6864-6890. https://doi.org/10.18653/v1/2024.acl-long.371

Hong WY, Wang WH, Lv QS, et al., 2024. CogAgent: a visual language model for GUI agents. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.14281-14290. https://doi.org/10.1109/CVPR52733.2024.01354

Hong WY, Yu WM, Gu XT, et al. 2025. GLM-4.1V-Thinking: towards versatile multimodal reasoning with scalable reinforcement learning. https://arxiv.org/abs/2507.01006v1

Hsiao YC, Zubach F, Baechler G, et al., 2025. ScreenQA: large-scale question–answer pairs over mobile app screenshots. Proc Conf of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, p.9427-9452. https://doi.org/10.18653/v1/2025.naacl-long.477

Hu XY, Xiong T, Yi B, et al., 2025. OS agents: a survey on MLLM-based agents for computer, phone and browser use. Proc 63rd Annual Meeting of the Association for Computational Linguistics, p.7436-7465. https://doi.org/10.18653/v1/2025.acl-long.369

Huang J, Zeng ZX, Han WK, et al., 2025. ScaleTrack: scaling and back-tracking automated GUI agents. https://doi.org/10.48550/arXiv.2505.00416

Huang KH, Qiu HY, Dai YT, et al., 2025. GUI-KV: efficient GUI agents via KV cache with spatio-temporal awareness. https://arxiv.org/abs/2510.00536

Hui Z, Li YH, Zhao D, et al., 2025. WinSpot: GUI grounding benchmark with multimodal large language models. Proc 63rd Annual Meeting of the Association for Computational Linguistics, p.1086-1096. https://doi.org/10.18653/v1/2025.acl-short.85

Jang LK, Li YH, Zhao D, et al., 2025. VideoWebArena: evaluating long context multimodal agents with video understanding web tasks. Proc 13th Int Conf on Learning Representations, p.1-25.

Jia HR, Liao JT, Zhang X, et al., 2026. OSWorld-MCP: benchmarking MCP tool invocation in computer-use agents. Proc 14th Int Conf on Learning Representations, p.1-35.

Jin YL, Li Z, Zhang CW, et al., 2024. Shopping MMLU: a massive multi-task online shopping benchmark for large language models. Proc 38th Conf on Neural Information Processing Systems, p.1-28.

Kapoor R, Butala YP, Russak M, et al., 2024. OmniACT: a dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web. Proc 18th European Conf on Computer Vision, p.161-178. https://doi.org/10.1007/978-3-031-73113-6_10

Koh JY, Lo R, Jang L, et al., 2024. VisualWebArena: evaluating multimodal agents on realistic visual web tasks. Proc 62nd Annual Meeting of the Association for Computational Linguistics, p.881-905. https://doi.org/10.18653/v1/2024.acl-long.50

Kong QY, Zhang X, Yang ZY, et al., 2025. MobileWorld: benchmarking autonomous mobile agents in agent–user interactive, and MCP-augmented environments. https://doi.org/10.48550/arXiv.2512.19432

Kumar P, Lau E, Vijayakumar S, et al., 2024. Refusal-trained LLMs are easily jailbroken as browser agents. https://doi.org/10.48550/arXiv.2410.13886

Lai HY, Liu X, Iong IL, et al., 2024. AutoWebGLM: a large language model-based web navigating agent. Proc 30th ACM SIGKDD Conf on Knowledge Discovery and Data Mining, p.5295-5306. https://doi.org/10.1145/3637528.3671620

Lai HY, Gao JJ, Liu X, et al., 2025. AndroidGen: building an Android language agent under data scarcity. Proc 63rd Annual Meeting of the Association for Computational Linguistics, p.2727-2749. https://doi.org/10.18653/v1/2025.acl-long.138

Lee J, Hahm D, Choi JS, et al., 2026. MobileSafetyBench: evaluating safety of autonomous agents in mobile device control. Proc 40th AAAI Conf on Artificial Intelligence, p.37565-37573. https://doi.org/10.1609/aaai.v40i44.41090

Lei B, Xu N, Payani A, et al., 2025. GUI-SPOTLIGHT: adaptive iterative focus refinement for enhanced GUI visual grounding. https://arxiv.org/abs/2510.04039

Leung HF, Xi XY, Zuo F, 2025. AndroidControl-Curated: revealing the true potential of GUI agents through benchmark purification. https://doi.org/10.48550/arXiv.2510.18488

Levy I, Wiesel B, Marreed S, et al., 2026. ST-WebAgentBench: a benchmark for evaluating safety and trustworthiness in web agents. Proc 14th Int Conf on Learning Representations, p.1-43.

Li E, Waldo J, 2024. WebSuite: systematically evaluating why web agents fail. https://doi.org/10.48550/arXiv.2406.01623

Li KX, Meng ZY, Lin HZ, et al., 2025. ScreenSpot-Pro: GUI grounding for professional high-resolution computer use. Proc 33rd ACM International Conference on Multimedia, p.8778-8786. https://doi.org/10.1145/3746027.3755688

Li SF, Kallidromitis K, Gokul A, et al., 2025. MobileWorldBench: towards semantic world modeling for mobile agents. https://doi.org/10.48550/arXiv.2512.14014

Li SL, Bu XY, Wang WJ, et al., 2025. MM-BrowseComp: a comprehensive benchmark for multimodal browsing agents. https://arxiv.org/abs/2508.13186

Li T, Li G, Zheng JJ, et al., 2024. MUG: interactive multimodal grounding on user interfaces. Findings of the Association for Computational Linguistics: EACL, p.231-251. https://doi.org/10.18653/v1/2024.findings-eacl.17

Li TJJ, Popowski L, Mitchell TM, et al., 2021. Screen2Vec: semantic embedding of GUI screens and GUI components. Proc CHI Conf on Human Factors in Computing Systems, Article 578. https://doi.org/10.1145/3411764.3445049

Li WZ, Lin JB, Jiang ZS, et al., 2025. Chain-of-Agents: end-to-end agent foundation models via multi-agent distillation and agentic RL. https://doi.org/10.48550/arXiv.2508.13167

Li YD, Zhang C, Jiang WJ, et al., 2024. AppAgent v2: advanced agent for flexible mobile interactions. https://doi.org/10.48550/arXiv.2408.11824

Li ZH, You KE, Zhang HT, et al., 2025. Ferret-UI 2: mastering universal user interface understanding across platforms. Proc 13th Int Conf on Learning Representations, p.1-24.

Liao ZY, Mo LB, Xu CJ, et al., 2025. EIA: environmental injection attack on generalist web agents for privacy leakage. Proc 13th Int Conf on Learning Representations, p.1-32.

Liao ZY, Jones J, Jiang LX, et al., 2026. RedTeamCUA: realistic adversarial testing of computer-use agents in hybrid web-OS environments. Proc 14th Int Conf on Learning Representations, p.1-46.

Lin HJ, Tan XY, Qin YL, et al., 2025. CUARewardBench: a benchmark for evaluating reward models on computer-using agent. https://doi.org/10.48550/arXiv.2510.18596

Lin KQ, Li LJ, Gao DF, et al., 2024. VideoGUI: a benchmark for GUI automation from instructional videos. Proc 38th Conf on Advances in Neural Information Processing Systems, p.1-32.

Lin KQ, Li LJ, Gao DF, et al., 2025. ShowUI: one vision-language-action model for GUI visual agent. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.19498-19508. https://doi.org/10.1109/cvpr52734.2025.01816

Liu EZ, Guu K, Pasupat P, et al., 2018. Reinforcement learning on web interfaces using workflow-guided exploration. Proc 6th Int Conf on Learning Representations, p.1-15.

Liu GY, Zhao PX, Liu L, et al., 2025. LearnAct: few-shot mobile GUI agent with a unified demonstration benchmark. https://doi.org/10.48550/arXiv.2504.13805

Liu GY, Zhao PX, Liang YZ, et al., 2026. MemGUI-Bench: benchmarking memory of mobile GUI agents in dynamic environments. https://arxiv.org/abs/2602.06075

Liu JP, Ou TY, Song YF, et al., 2025. Harnessing webpage UIs for text-rich visual understanding. Proc 13th Int Conf on Learning Representations, p.1-35.

Liu SY, Liu MH, Zhou HC, et al., 2025. VeriWeb: verifiable long-chain GUI dataset. https://arxiv.org/abs/2508.04026v1

Liu X, Yu H, Zhang HC, et al., 2024a. AgentBench: evaluating LLMs as agents. Proc 12th Int Conf on Learning Representations, p.1-58.

Liu X, Qin B, Liang DZ, et al., 2024b. AutoGLM: autonomous foundation agents for GUIs. https://doi.org/10.48550/arXiv.2411.00820

Liu YH, Li PX, Wei ZS, et al., 2026. InfiGUIAgent: a multimodal generalist GUI agent with native reasoning and reflection. Proc 19th Conf of the European Chapter of the Association for Computational Linguistics, p.1035-1051. https://doi.org/10.18653/v1/2026.eacl-long.45

Liu ZY, Xie JJ, Ding ZC, et al., 2025. ScaleCUA: scaling open-source computer use agents with cross-platform data. https://doi.org/10.48550/arXiv.2509.15221

Lu JT, Zhang ZY, Yang FK, et al., 2024. Turn every application into an agent: towards efficient human–agent–computer interaction with API-first LLM-based agents. https://arxiv.org/abs/2409.17140v1

Lu QF, Shao WQ, Liu ZT, et al., 2025. GUIOdyssey: a comprehensive dataset for cross-app GUI navigation on mobile devices. Proc IEEE/CVF Int Conf on Computer Vision, p.22404-22414. https://doi.org/10.1109/iccv51701.2025.02080

Lu YD, Yang JW, Shen YL, et al., 2024. OmniParser for pure vision based GUI agent. https://doi.org/10.48550/arXiv.2408.00203

Lù XH, Kasner Z, Reddy S, 2024. WebLINX: real-world website navigation with multi-turn dialogue. Proc 41st Int Conf on Machine Learning, p.33007-33056.

Luo TG, Logeswaran L, Johnson J, et al., 2025. Visual test-time scaling for GUI agent grounding. Proc IEEE/CVF Int Conf on Computer Vision, p.19989-19998. https://doi.org/10.1109/iccv51701.2025.01859

Ma XB, Wang YT, Yao Y, et al., 2025. Caution for the environment: multimodal LLM agents are susceptible to environmental distractions. Proc 63rd Annual Meeting of the Association for Computational Linguistics, p.22324-22339. https://doi.org/10.18653/v1/2025.acl-long.1087

Mialon G, Fourrier C, Swift C, et al., 2024. GAIA: a benchmark for general AI assistants. Proc 12th Int Conf on Learning Representations, p.1-25.

Mozannar H, Bansal G, Tan C, et al., 2025. Magentic-UI: towards human-in-the-loop agentic systems. https://doi.org/10.48550/arXiv.2507.22358

Mu J, Zhang CY, Ni CM, et al., 2025. GUI-360°: a comprehensive dataset and benchmark for computer-using agents. https://doi.org/10.48550/arXiv.2511.04307

Murty S, Zhu H, Bahdanau D, et al., 2024. NNetNav: unsupervised learning of browser agents through environment interaction in the wild. https://doi.org/10.48550/arXiv.2410.02907

Nakano R, Hilton J, Balaji S, et al., 2021. WebGPT: browser-assisted question-answering with human feedback. https://doi.org/10.48550/arXiv.2112.09332

Nayak S, Jian XR, Lin KQ, et al., 2025. UI-Vision: a desktop-centric GUI benchmark for visual perception and interaction. Proc 42nd Int Conf on Machine Learning, p.45817-45851.

Nguyen A, 2024. Improved GUI grounding via iterative narrowing. https://doi.org/10.48550/arXiv.2411.13591

Nguyen D, Chen J, Wang Y, et al., 2025. GUI agents: a survey. Findings of the Association for Computational Linguistics: ACL, p.22522-22538. https://doi.org/10.18653/v1/2025.findings-acl.1158

Nong SQ, Zhu JL, Wu R, et al., 2024. MobileFlow: a multimodal LLM for mobile GUI agent. https://doi.org/10.48550/arXiv.2407.04346

OpenAI ChatGPT Agent Team, 2025. Introducing ChatGPT agent: bridging research and action. https://openai.com/index/introducing-chatgpt-agent [Accessed on Apr. 1, 2026].

OpenAI CUA Team, 2025. Computer-using agent. https://openai.com/index/computer-using-agent [Accessed on Apr. 1, 2026].

OpenAI Developer Platform Team, 2025. Introducing GPT-5 for developers. https://openai.com/index/introducing-gpt-5-for-developers [Accessed on Apr. 1, 2026].

OpenAI GPT-5 Team, 2025. Introducing GPT-5. https://openai.com/index/introducing-gpt-5 [Accessed on Apr. 1, 2026].

OpenAI GPT-5.4 Mini and Nano Team, 2026. Introducing GPT-5.4 mini and nano. https://openai.com/index/introducing-gpt-5-4-mini-and-nano [Accessed on Apr. 1, 2026].

OpenAI GPT-5.4 Team, 2026. Introducing GPT-5.4. https://openai.com/index/introducing-gpt-5-4 [Accessed on Apr. 1, 2026].

OpenAI o3 Operator Team, 2025. Addendum to OpenAI o3 and o4-mini system card: OpenAI o3 Operator. https://openai.com/index/o3-o4-mini-system-card-addendum-operator-o3 [Accessed on Apr. 1, 2026].

OpenAI Operator System Card Team, 2025. Operator system card. https://openai.com/index/operator-system-card [Accessed on Apr. 1, 2026].

OpenAI Operator Team, 2025. Introducing operator. https://openai.com/index/introducing-operator [Accessed on Apr. 1, 2026].

OpenGVLab, 2025. WebArena-Lite-v2 benchmark evaluation guide. https://github.com/OpenGVLab/ScaleCUA/tree/main/evaluation/WebArenaLiteV2 [Accessed on Apr. 1, 2026].

Ou TY, Xu FF, Madaan A, et al., 2024. Synatra: turning indirect knowledge into direct demonstrations for digital agents at scale. Proc 38th Int Conf on Advances in Neural Information Processing Systems, Article 2908.

Pahuja V, Lu YD, Rosset C, et al., 2025. Explorer: scaling exploration-driven web trajectory synthesis for multimodal web agents. Findings of the Association for Computational Linguistics: ACL, p.6300-6323. https://doi.org/10.18653/v1/2025.findings-acl.326

Pan LH, Wang BW, Yu C, et al., 2023. AutoTask: executing arbitrary voice commands by exploring and learning from mobile GUI. https://doi.org/10.48550/arXiv.2312.16062

Pan YC, Kong DH, Zhou SD, et al., 2024. WebCanvas: benchmarking web agents in online environments. Proc ICML Workshop on Agentic Markets, p.1-26.

Park J, Tang P, Das S, et al., 2025. R-VLM: region-aware vision language model for precise GUI grounding. Findings of the Association for Computational Linguistics: ACL, p.9669-9685. https://doi.org/10.18653/v1/2025.findings-acl.501

Park JS, O'Brien JC, Cai CJ, et al., 2023. Generative agents: interactive simulacra of human behavior. Proc 36th Annual ACM Symp on User Interface Software and Technology, Article 2. https://doi.org/10.1145/3586183.3606763

Pawlowski P, Zawistowski K, Lapacz W, et al., 2025. TinyClick: single-turn agent for empowering GUI automation. Proc 26th Annual Conf of the Int Speech Communication Association, p.3035-3039. https://doi.org/10.21437/Interspeech.2025-176

Qin YJ, Ye YN, Fang JJ, et al., 2025. UI-TARS: pioneering automated GUI interaction with native agents. https://doi.org/10.48550/arXiv.2501.12326

Qwen3 Team, 2025. Qwen3: think deeper, act faster. https://qwenlm.github.io/blog/qwen3 [Accessed on Apr. 1, 2026].

Qwen3-Coder Team, 2025. Qwen3-coder: agentic coding in the world. https://qwenlm.github.io/blog/qwen3-coder [Accessed on Apr. 1, 2026].

Qwen3Guard Team, 2025. Qwen3Guard: real-time safety for your token stream. https://qwenlm.github.io/blog/qwen3guard [Accessed on Apr. 1, 2026].

Rawles C, Li A, Rodriguez D, et al., 2023. Android in the Wild: a large-scale dataset for Android device control. Proc 37th Conf on Neural Information Processing Systems, Datasets and Benchmarks Track, p.1-21.

Rawles C, Clinckemaillie S, Chang YF, et al., 2025. AndroidWorld: a dynamic benchmarking environment for autonomous agents. Proc 13th Int Conf on Learning Representations, p.1-36.

Rodriguez JA, Jian XR, Panigrahi SS, et al., 2025. BigDocs: an open and permissively-licensed dataset for training multimodal models on document and code tasks. Proc 13th Int Conf on Learning Representations, p.1-57.

Sager PJ, Meyer B, Yan P, et al., 2026. A comprehensive survey of agents for computer use: foundations, challenges, and future directions. *J Artif Intell Res*, 85:34. https://doi.org/10.1613/jair.1.19490

Schick T, Dwivedi-Yu J, Dessi R, et al., 2023. Toolformer: language models can teach themselves to use tools. Proc 37th Int Conf on Advances in Neural Information Processing Systems, Article 2997.

Shaw P, Joshi M, Cohan J, et al., 2023. From pixels to UI actions: learning to follow instructions via graphical user interfaces. Proc 37th Int Conf on Neural Information Processing Systems, Article 1490.

Shen HW, Liu C, Li GL, et al., 2024. Falcon-UI: understanding GUI before following user instructions. https://doi.org/10.48550/arXiv.2412.09362

Shen JH, Jain A, Xiao ZD, et al., 2024. ScribeAgent: towards specialized web agents using production-scale workflow data. https://doi.org/10.48550/arXiv.2411.15004

Shen JH, Bai H, Zhang LJ, et al., 2025. Thinking vs. doing: agents that reason by scaling test-time interaction. https://doi.org/10.48550/arXiv.2506.07976

Shi TL, Karpathy A, Fan LX, et al., 2017. World of Bits: an open-domain platform for web-based agents. Proc 34th Int Conf on Machine Learning, p.3135-3144.

Shi YC, Yu WH, Huang JY, et al., 2025. Towards trustworthy GUI agents: a survey. https://doi.org/10.48550/arXiv.2503.23434

Shi ZR, Mei K, Jin MY, et al., 2025. From commands to prompts: LLM-based semantic file system for AIOS. Proc 13th Int Conf on Learning Representations, p.1-24.

Shinn N, Cassano F, Gopinath A, et al., 2023. Reflexion: language agents with verbal reinforcement learning. Proc 37th Int Conf on Neural Information Processing Systems, Article 377.

Song YP, Bian YH, Tang YT, et al., 2024. VisionTasker: mobile task automation using vision based UI understanding and LLM task planning. Proc 37th Annual ACM Symp on User Interface Software and Technology, Article 49. https://doi.org/10.1145/3654777.3676386

Styles O, Miller S, Cerda-Mardini P, et al., 2024. WorkBench: a benchmark dataset for agents in a realistic workplace setting. Proc 1st Conf on Language Modeling, p.1-39.

Steel.dev, 2026. WebArena leaderboard. https://leaderboard.steel.dev/leaderboards/webarena/ [Accessed on Apr. 1, 2026].

Su Y, Awadallah AH, Khabsa M, et al., 2017. Building natural language interfaces to web APIs. Proc 26th ACM Int Conf on Information and Knowledge Management, p.177-186. https://doi.org/10.1145/3132847.3133009

Sun LT, Chen XY, Chen L, et al., 2022. META-GUI: towards multi-modal conversational agents on mobile GUI. Proc Conf on Empirical Methods in Natural Language Processing, p.6699-6712. https://doi.org/10.18653/v1/2022.emnlp-main.449

Sun QS, Liu ZMZ, Ma C, et al., 2026. ScienceBoard: evaluating multimodal autonomous agents in realistic scientific workflows. Proc 14th Int Conf on Learning Representations, p.1-38.

Toyama D, Hamel P, Gergely A, et al., 2021. AndroidEnv: a reinforcement learning platform for Android. https://doi.org/10.48550/arXiv.2105.13231

Tur AD, Meade N, Lù XH, et al., 2025. SafeArena: evaluating the safety of autonomous web agents. Proc 42nd Int Conf on Machine Learning, p.60404-60441.

Vaswani A, Shazeer N, Parmar N, et al., 2017. Attention is all you need. Proc 31st Int Conf on Neural Information Processing Systems, p.6000-6010.

Venkatesh SG, Talukdar P, Narayanan S, 2022. UGIF: UI grounded instruction following. https://doi.org/10.48550/arXiv.2211.07615

Vu MD, Wang H, Chen JS, et al., 2024. GPTVoiceTasker: advancing multi-step mobile task efficiency through dynamic interface exploration and learning. Proc 37th Annual ACM Symp on User Interface Software and Technology, Article 48. https://doi.org/10.1145/3654777.3676356

Wang B, Li G, Zhou X, et al., 2021. Screen2Words: automatic mobile UI summarization with multimodal learning. Proc 34th Annual ACM Symp on User Interface Software and Technology, p.498-510. https://doi.org/10.1145/3472749.3474765

Wang HM, Zou HY, Song HT, et al., 2025. UI-TARS-2 technical report: advancing GUI agent with multi-turn reinforcement learning. https://doi.org/10.48550/arXiv.2509.02544

Wang JY, Xu HY, Ye JB, et al., 2024. Mobile-Agent: autonomous multi-modal mobile device agent with visual perception. https://doi.org/10.48550/arXiv.2401.16158

Wang K, Xia TY, Gu ZX, et al., 2024. E-ANT: a large-scale dataset for efficient automatic GUI navigation. https://doi.org/10.48550/arXiv.2406.14250

Wang LY, Deng YY, Zha YW, et al., 2024. MobileAgentBench: an efficient and user-friendly benchmark for mobile LLM agents. https://doi.org/10.48550/arXiv.2406.08184

Wang P, Tao RH, Chen QG, et al., 2025. X-WebAgentBench: a multilingual interactive web benchmark for evaluating global agentic system. Findings of the Association for Computational Linguistics: ACL, p.19320-19335. https://doi.org/10.18653/v1/2025.findings-acl.988

Wang S, Liu WW, Chen JX, et al., 2024. GUI agents with foundation models: a comprehensive survey. https://doi.org/10.48550/arXiv.2411.04890

Wang XL, Bloch J, Shao ZD, et al., 2025. WebInject: prompt injection attack to web agents. Proc Conf on Empirical Methods in Natural Language Processing, p.2010-2030. https://doi.org/10.18653/v1/2025.emnlp-main.104

Wang YQ, Zhang HJ, Tian JQ, et al., 2025. Ponder & Press: advancing visual GUI agent towards general computer control. Findings of the Association for Computational Linguistics: ACL, p.1461-1473. https://doi.org/10.18653/v1/2025.findings-acl.76

Wen H, Wang HM, Liu JX, et al., 2023. DroidBot-GPT: GPT-powered UI automation for Android. https://doi.org/10.48550/arXiv.2304.07061

Wen H, Tian SZ, Pavlov B, et al., 2025. AutoDroid-V2: boosting SLM-based GUI agents via code generation. Proc 23rd Annual Int Conf on Mobile Systems, Applications and Services, p.223-235. https://doi.org/10.1145/3711875.3729134

Wu CH, Shah RR, Koh JY, et al., 2025. Dissecting adversarial robustness of multimodal LM agents. Proc 13th Int Conf on Learning Representations, p.1-22.

Wu FZ, Wu ST, Cao YL, et al., 2024. WIPI: a new web threat for LLM-driven web agents. https://doi.org/10.48550/arXiv.2402.16965

Wu QZ, Xu WK, Liu W, et al., 2024. MobileVLM: a vision-language model for better intra- and inter-UI understanding. Findings of the Association for Computational Linguistics: EMNLP, p.10231-10251. https://doi.org/10.18653/v1/2024.findings-emnlp.599

Wu QZ, Gao PZ, Liu W, et al., 2025. BacktrackAgent: enhancing GUI agent with error detection and backtracking mechanism. Proc Conf on Empirical Methods in Natural Language Processing, p.4250-4272. https://doi.org/10.18653/v1/2025.emnlp-main.212

Wu ZY, Wu ZY, Xu FZ, et al., 2025. OS-Atlas: a foundation action model for generalist GUI agents. Proc 13th Int Conf on Learning Representations, p.1-19.

Xie B, Shao R, Chen GW, et al., 2025. GUI-Explorer: autonomous exploration and mining of transition-aware knowledge for GUI agent. Proc 63rd Annual Meeting of the Association for Computational Linguistics, p.5650-5667.

Xie JL, Chen ZH, Zhang RF, et al., 2025. Large multimodal agents: a survey. *Vis Intell*, 3(1):24. https://doi.org/10.1007/s44267-025-00093-y

Xie TB, Zhou F, Cheng ZJ, et al., 2024a. OpenAgents: an open platform for language agents in the wild. Proc 1st Conf on Language Modeling, p.1-36.

Xie TB, Zhang DY, Chen JX, et al., 2024b. OSWorld: benchmarking multimodal agents for open-ended tasks in real computer environments. Proc 38th Int Conf on Neural Information Processing Systems, Article 1650.

Xie TB, Deng JQ, Li XC, et al., 2025. Scaling computer-use grounding via user interface decomposition and synthesis. Proc 39th Conf on Neural Information Processing Systems, p.1-58.

XLANG Lab, 2025. OSWorld-Verified. https://xlang.ai/blog/osworld-verified [Accessed on Apr. 1, 2026].

Xu CJ, Kang MT, Zhang JW, et al., 2025. AdvAgent: controllable blackbox red-teaming on web agents. Proc 42nd Int Conf on Machine Learning, p.69318-69330.

Xu HY, Zhang X, Liu HW, et al., 2026. Mobile-Agent-v3.5: multi-platform fundamental GUI agents. https://doi.org/10.48550/arXiv.2602.16855

Xu QT, Hong FL, Li B, et al., 2023. On the tool manipulation capability of open-source large language models. https://doi.org/10.48550/arXiv.2305.16504

Xu TQ, Chen LY, Wu DJ, et al., 2025. CRAB: cross-environment agent benchmark for multimodal language model agents. Findings of the Association for Computational Linguistics: ACL, p.21607-21647. https://doi.org/10.18653/v1/2025.findings-acl.1113

Xu YF, Liu X, Sun XQ, et al., 2025a. AndroidLab: training and systematic benchmarking of Android autonomous agents. Proc 63rd Annual Meeting of the Association for Computational Linguistics, p.2144-2166. https://doi.org/10.18653/v1/2025.acl-long.107

Xu YF, Liu X, Liu XH, et al., 2025b. MobileRL: online agentic reinforcement learning for mobile GUI agents. https://arxiv.org/abs/2509.18119

Xu YH, Lu DJ, Shen ZN, et al., 2025a. AgentTrek: agent trajectory synthesis via guiding replay with web tutorials. Proc 13th Int Conf on Learning Representations, p.1-22.

Xu YH, Wang ZK, Wang JL, et al., 2025b. AGUVIS: unified pure vision agents for autonomous GUI interaction. Proc 42nd Int Conf on Machine Learning, p.69772-69805.

X-WebArena Leaderboard Maintainers, 2026. X-WebArena-Leaderboard: public WebArena results spreadsheet. https://docs.google.com/spreadsheets/d/1M801lEpBbKSNwP-vDBkC_pF7LdyGU1f_ufZb_NWNBZQ [Accessed on Apr. 1, 2026].

Yang JW, Zhang H, Li F, et al., 2023. Set-of-Mark prompting unleashes extraordinary visual grounding in GPT-4V. https://doi.org/10.48550/arXiv.2310.11441

Yang K, Liu Y, Chaudhary S, et al., 2025. AgentOccam: a simple yet strong baseline for LLM-based web agents. Proc 13th Int Conf on Learning Representations, p.1-33.

Yang P, Ci H, Shou MZ, 2025. macOSWorld: a multilingual interactive benchmark for GUI agents. https://doi.org/10.48550/arXiv.2506.04135

Yang Y, Li DX, Dai YT, et al., 2026. GTA1: GUI test-time scaling agent. Proc 14th Int Conf on Learning Representations, p.1-29.

Yang YH, Wang Y, Li DX, et al., 2025. Aria-UI: visual grounding for GUI instructions. Findings of the Association for Computational Linguistics: ACL, p.22418-22433. https://doi.org/10.18653/v1/2025.findings-acl.1152

Yang YL, Yang XS, Li SD, et al., 2024. Security matrix for multimodal agents on mobile devices: a systematic and proof of concept study. https://arxiv.org/abs/2407.09295v1

Yang Z, Dou ZY, Feng D, et al., 2026. Ferret-UI Lite: lessons from building small on-device GUI agents. https://machinelearning.apple.com/research/ferret-ui [Accessed on Apr. 1, 2026].

Yao SY, Chen H, Yang J, et al., 2022. WebShop: towards scalable real-world web interaction with grounded language agents. Proc 36th Int Conf on Advances in Neural Information Processing Systems, Article 1508.

Yao SY, Zhao J, Yu D, et al., 2023. ReAct: synergizing reasoning and acting in language models. Proc 11th Int Conf on Learning Representations, p.1-33.

Ye JB, Zhang X, Xu HY, et al., 2025. Mobile-Agent-v3: fundamental agents for GUI automation. https://doi.org/10.48550/arXiv.2508.15144

Yin XR, Luo X, Wu H, et al., 2025. Unlocking smarter device control: foresighted planning with a world model-driven code execution approach. Findings of the Association for Computational Linguistics: EMNLP, p.3982-4005. https://doi.org/10.18653/v1/2025.findings-emnlp.213

Ying KN, Meng FQ, Wang J, et al., 2024. MMT-Bench: a comprehensive multimodal benchmark for evaluating large vision-language models towards multitask AGI. Proc 41st Int Conf on Machine Learning, p.57116-57198.

Yu S, Li G, Shi WY, et al., 2025. PolySkill: learning generalizable skills through polymorphic abstraction. https://doi.org/10.48550/arXiv.2510.15863

Zhang BL, Xiong SY, Sui DB, et al., 2024. RealWeb: a benchmark for universal instruction following in realistic web services navigation. Proc IEEE Int Conf on Web Services, p.342-351. https://doi.org/10.1109/icws62655.2024.00056

Zhang C, Yang Z, Liu JX, et al., 2025. AppAgent: multimodal agents as smartphone users. Proc CHI Conf on Human Factors in Computing Systems, Article 70. https://doi.org/10.1145/3706598.3713600

Zhang CY, He SL, Qian JX, et al., 2025a. Large language model-brained GUI agents: a survey. https://arxiv.org/abs/2411.18279

Zhang CY, Li LQ, He SL, et al., 2025b. UFO: a UI-Focused Agent for Windows OS interaction. Proc Conf of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, p.597-622. https://doi.org/10.18653/v1/2025.naacl-long.26

Zhang DY, Chen L, Yu K, 2023. Mobile-Env: a universal platform for training and evaluation of mobile interaction. https://arxiv.org/abs/2305.08144v1

Zhang DY, Zhang ST, Yang ZY, et al., 2025. ProgRM: build better GUI agents with progress rewards. https://doi.org/10.48550/arXiv.2505.18121

Zhang JW, Yu YQ, Liao MH, et al., 2025. UI-Hawk: unleashing the screen stream understanding for mobile GUI agents. Proc Conf on Empirical Methods in Natural Language Processing, p.18217-18236. https://doi.org/10.18653/v1/2025.emnlp-main.920

Zhang JY, Zhao C, Zhao YH, et al., 2024. MobileExperts: a dynamic tool-enabled agent team in mobile devices. https://doi.org/10.48550/arXiv.2407.03913

Zhang MS, Xu ZQ, Zhu JL, et al., 2025. Phi-Ground Tech Report: advancing perception in GUI grounding. Technical Report No. MSR-TR-2025-63, Microsoft Research, Redmond, WA.

Zhang SQ, Zhang ZS, Chen KH, et al., 2024. Dynamic planning for LLM-based graphical user interface automation. Findings of the Association for Computational Linguistics: EMNLP, p.1304-1320. https://doi.org/10.18653/v1/2024.findings-emnlp.70

Zhang YZ, Yu T, Yang DY, 2025. Attacking vision-language computer agents via pop-ups. Proc 63rd Annual Meeting of the Association for Computational Linguistics, p.8387-8401. https://doi.org/10.18653/v1/2025.acl-long.411

Zhang Z, Lu YX, Fu YK, et al., 2025. AgentCPM-GUI: building mobile-use agents with reinforcement fine-tuning. Proc Conf on Empirical Methods in Natural Language Processing: System Demonstrations, p.155-180. https://doi.org/10.18653/v1/2025.emnlp-demos.12

Zhang ZS, Zhang A, 2024. You Only Look at Screens: multimodal chain-of-action agents. Findings of the Association for Computational Linguistics: ACL, p.3132-3149. https://doi.org/10.18653/v1/2024.findings-acl.186

Zhang ZY, Liu XY, Zhang XY, et al., 2025. UI-Evol: automatic knowledge evolving for computer use agents. https://doi.org/10.48550/arXiv.2505.21964

Zhao HH, Gao DF, Shou MZ, 2025. WorldGUI: an interactive benchmark for desktop GUI automation from any starting point. https://doi.org/10.48550/arXiv.2502.08047

Zheng BY, Gou BY, Kil J, et al., 2024. GPT-4V(ision) is a generalist web agent, if grounded. Proc 41st Int Conf on Machine Learning, p.61349-61385.

Zhou HZ, Zhang X, Tong PR, et al., 2025. MAI-UI technical report: real-world centric foundation GUI agents. https://doi.org/10.48550/arXiv.2512.22047

Zhou SY, Xu FF, Zhu H, et al., 2024. WebArena: a realistic web environment for building autonomous agents. Proc 12th Int Conf on Learning Representations, p.1-22.

Zhu ZC, Tang H, Li YS, et al., 2024. MobA: multifaceted memory-enhanced adaptive planning for efficient mobile task automation. https://doi.org/10.48550/arXiv.2410.13757