GUI Agents with Foundation Models: A Comprehensive Survey

Anonymous ACL submission

Abstract

Recent advances in foundation models, particularly Large Language Models (LLMs) and Multimodal Large Language Models (MLLMs), have facilitated the development of intelligent agents capable of performing complex tasks. By leveraging the ability of (M)LLMs to process and interpret Graphical User Interfaces (GUIs), these agents can autonomously execute user instructions, simulating human-like interactions such as clicking and typing. This survey consolidates recent research on (M)LLM-based GUI agents, highlighting key innovations in data resources, frameworks, and applications. We begin by reviewing representative datasets and benchmarks, followed by an overview of a generalized, unified framework that encapsulates the essential components of prior studies, supported by a detailed taxonomy. Additionally, we explore relevant commercial applications. Drawing insights from existing work, we identify key challenges and propose future research directions. We hope this survey will inspire further advancements in the field of (M)LLM-based GUI agents.

1 Introduction

Graphical User Interfaces (GUIs) are the primary medium through which humans interact with digital devices. From mobile phones to websites, people engage with GUIs daily, and well-designed GUI agents can significantly enhance the user experience. Thus, research on GUI agents has been extensive. However, traditional methods struggle with coomplex tasks requiring human-like interactions (Liu et al., 2018a; Toyama et al., 2021), limiting the applicability of GUI agents.

Recent advancements in Large Language Models (LLMs) and Multimodal Large Language Models (MLLMs) have significantly enhanced their capabilities in language understanding and cognitive processing (Achiam et al., 2024; Touvron et al., 2023; Yang et al., 2024a). With improved natural language comprehension and enhanced reasoning abilities, (M)LLM-based agents can now effectively interpret and utilize human language, formulate detailed plans, and execute complex tasks. These breakthroughs provide new opportunities for researchers to address challenges previously considered highly difficult, such as automating tasks within GUIs.

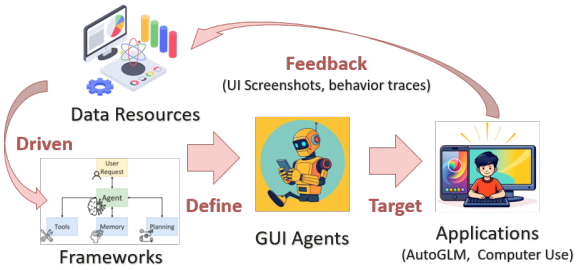

Figure 1: The foundational aspects and goals of GUI agents.

As shown in Figure 2, recent studies on GUI agents illustrate a shift from simple Transformer-based models to (M)LLM-based agentic frameworks. Their capabilities have expanded from single-modality interactions to multimodal processing, making them increasingly relevant to commercial applications. Given these advancements, we believe it is timely to systematically analyze the development trends of GUI agents, particularly from an application perspective.

This paper aims to provide a structured overview of the latest and influential work in the field of GUI agents. As depicted in Figure 1, we focus on the foundational aspects and goals of GUI agents. Data resources, such as user instructions, User Interface (UI) screenshots, and behavior traces, drive the design of GUI agents (Rawles et al., 2023; Lu et al., 2024a). Frameworks define the underlying algorithms and models that enable intelligent decision-making (Li et al., 2024b; Wang et al., 2024a; Zhu

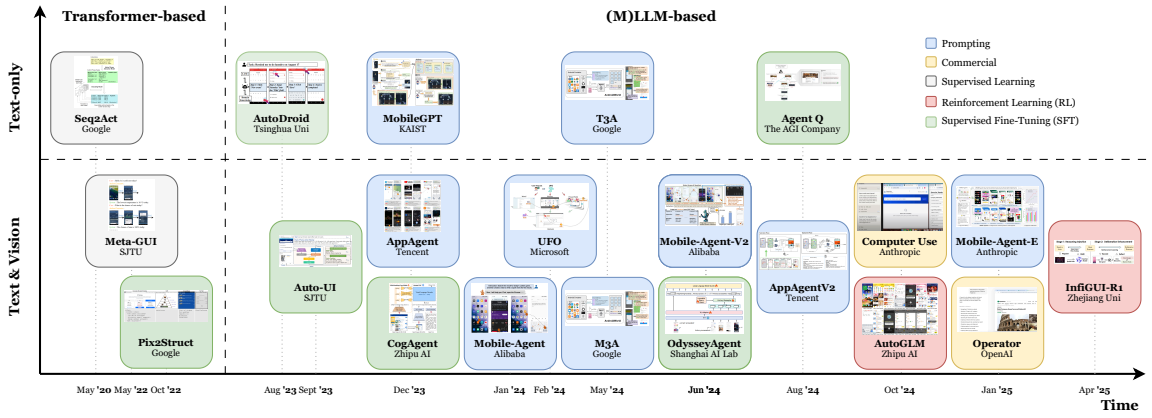

Figure 2: Illustration of the growth trend in the field of GUI agents with foundation models.

et al., 2024). Applications represent the optimized and practical goals (Lai et al., 2024; Liu et al., 2024). The current state of these aspects reflects the maturity of the field and highlights future research priorities.

To this end, we organize this survey around three key areas: **Data Resources**, **Frameworks**, and **Applications**. The main contributions of this paper are: 1) a comprehensive summary of existing research and a detailed review of current data sources, providing a useful guide for newcomers to the field; 2) a unified and generalized GUI agent framework with clearly defined and categorized functional components to facilitate a structured review; 3) an analysis of trends in both research and commercial applications of GUI agents.

2 GUI Agent Data Resources

Recent research has focused on developing datasets and benchmarks to train and evaluate the capabilities of (M)LLM-based GUI agents. A variety of datasets are available for training GUI agents. These agents employ different approaches to interact with environments. Additionally, multiple methods have been proposed for evaluation.

Dataset: Common datasets for training GUI agents typically contain natural language instructions that describe task goals, along with demonstration trajectories that include screenshots and action pairs. A pioneering work in this area is PIXELHELP (Li et al., 2020), which introduces a new class of problems focused on translating natural language instructions into actions on mobile user interfaces. In recent years, Android in the Wild (Rawles et al., 2023) has created a dataset featuring a variety of single-step and multi-step tasks. Aimed at advancing GUI navigation agent research, Android-In-The-Zoo (Zhang et al., 2024b) introduces a benchmark dataset with chained action reasoning annotations.

Insight-UI (Shen et al., 2024) automatically constructs a GUI pre-training dataset that simulates multiple platforms across 312,000 domains. To assess model performance both within and beyond the scope of training data, AndroidControl (Li et al., 2024a) includes demonstrations of daily tasks along with both high- and low-level human-generated instructions. The scope of mobile control datasets is further extended from single-application to cross-application scenarios by GUI-Odyssey (Lu et al., 2024a).

Most of the aforementioned datasets are primarily limited to English and image-based tasks. However, UGIF Dataset (Venkatesh et al., 2024) covers eight languages, Mobile3M (Wu et al., 2024) focuses on Chinese, and GUI-WORLD (Chen et al., 2024a) includes video annotations, expanding the dataset landscape for broader multilingual and multimodal research.

Environment: GUI agents require environments for task execution, which can be broadly categorized into three types. The first category is static environments, where the environment remains fixed as it was when developed. Agents in this category operate within predefined datasets without the ability to create new states.

In contrast, the second and third categories involve dynamic environments, where new outcomes can emerge during agent execution. The key distinction between these categories lies in whether the dynamic environment is simulated or realistic. Simulations of real-world environments require ad-

ditional implementation but are often cleaner and free of distractions, such as pop-ups and advertisements. WebArena (Zhou et al., 2023) implements a versatile website covering e-commerce, social forums, collaborative software development, and content management. Similarly, GUI Testing Arena (Zhao et al., 2024) provides a standardized environment for testing GUI agents, including defect injection.

For realistic environments, agents interact directly with web or mobile platforms as human users do, better reflecting real-world conditions. SPA-Bench (Chen et al., 2024b) encompasses tasks that involve both system and third-party mobile applications, supporting single-app and cross-app scenarios in both English and Chinese.

Evaluation: Another critical component of GUI agent datasets is the evaluation of agent performance. The most common and important metric is success rate, which measures how effectively an agent completes tasks. Additional metrics, such as efficiency, are sometimes considered as well.

Evaluation methods are often closely tied to the environment type. In static environments, action matching is a widely used method that compares an agent's executed action sequence with a human we may demonstration (e.g., Rawles et al. (2023), Li et al. (2024a)). However, a major limitation of action matching is its inability to account for multiple successful execution paths, leading to false negatives when evaluating agent performance.

Evaluating dynamic environments, whether simulated or realistic, presents additional challenges due to their uncertain conditions. Evaluation methods can range from fully human-dependent to semi-automated and fully automated approaches. Human evaluations require manual verification, making them non-reusable. In AppAgent (Li et al., 2024b) and MobileAgent (Ding, 2024), human evaluators assess whether each agent-executed task was successful. Semi-automated evaluations involve human-developed validation logic that can be reused for different execution trajectories of the same task. For example, WebArena (Zhou et al., 2023) and AndroidWorld (Rawles et al., 2024) incorporate handcrafted validation functions for task completion. Fully automated evaluations eliminate human involvement by relying on models for success detection. SPA-Bench (Chen et al., 2024b), for instance, employs MLLMs for evaluating task completion. Although reducing human labor is crucial for large-scale evaluation, balancing efficiency with accuracy remains a key research challenge.

3 (M)LLM-based GUI Agent

With the human-like capabilities of (M)LLMs, GUI agents aim to handle various tasks to meet users' needs. Organizing the frameworks of GUI agents and designing methods to optimize their performance is crucial to unlocking the full potential of (M)LLMs. As shown in Figure 4, we summarize a generalized **Framework** and discuss its components in relation to existing works in Section 3.1. Building on this foundation, we then review recent influential **Methods** for constructing and optimizing GUI agents, categorizing them with an exhaustive taxonomy in Section 3.2.

3.1 (M)LLM-based GUI Agent Framework

The goal of GUI agents is to automatically control a device to complete tasks defined by the user. Typically, GUI agents take a user's query and the device's UI status as inputs and generate a series of human-like actions to achieve the tasks.

As shown in Figure 3, we present a generalized (M)LLM-based GUI agent framework, consisting of five components: GUI Perceiver, Task Planner, Decision Maker, Memory Retriever, and Executor. Many variations of this framework exist. For instance, Wang et al. (2024a) proposes a multi-agent GUI control framework comprising a planning agent, a decision agent, and a reflection agent to tackle navigation challenges in mobile device operations. This approach shares functional similarities with our proposed framework. A follow-up study (Wang et al., 2025) further disentangles high-level planning from low-level actions by employing dedicated agents and introduces memory-based self-evolution to enhance performance.

GUI Perceiver: To effectively complete a device task, a GUI agent should accurately interpret user input and detect changes in the device's UI. Although language models excel in understanding user intent (Touvron et al., 2023; Achiam et al., 2024), navigating device UIs requires a reliable visual perception model to understand GUIs.

A GUI Perceiver appears explicitly or implicitly in GUI agent frameworks. For agents based on single-modal LLMs (Wen et al., 2023a,b; Li et al., 2020), a GUI Perceiver is usually an explicit module of the frameworks. However, for agents with multi-modal LLMs (Hong et al., 2024; Zhang et al.,

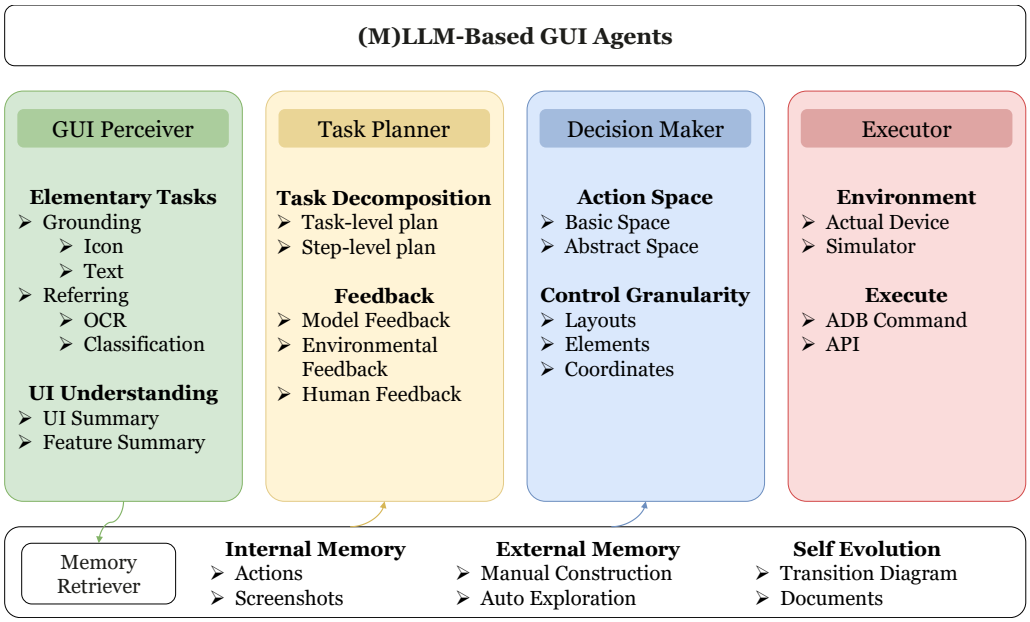

Figure 3: (M)LLM-based GUI agents: the generalized framework and key technologies.

2023; Wang et al., 2024b), UI perception is seen as a capability of the model itself.

UI perception is an important problem in GUI agents research. Hence, some research (You et al., 2024; Zhang et al., 2021; Lu et al., 2024b) focuses on processing UI, rather than building the agent. For example, Pix2struct (Lee et al., 2023a) employs a ViT-based image-encoder-text-decoder architecture, which pre-trains on Screenshot-HTML data pairs and fine-tunes for downstream tasks. Screen2words (Wang et al., 2021)encapsulates a UI screen into a coherent language representation, which is based on a transformer encoder-decoder architecture to process UIs and generate the representation. To address the defects of purely vision-based screen parsing methods, Ge et al. (2024) introduces Iris, a visual agent for GUI understanding, addressing challenges related to architectural limitations for heterogeneous GUI information and annotation bias in GUI training.

Task Planner: The GUI agent should effectively decompose complex tasks, often employing a Chain-of-Thought (CoT) approach. Due to the complexity of tasks, recent studies (Zhang et al., 2024a; Wang et al., 2024a) introduce an additional module to support more detailed planning.

In GUI agents, plan should adapt dynamically based on decision feedback, typically achieved through a ReAct-style. For instance, Zhang et al. (2023) uses on-screen observations to enhance the CoT for improved decision-making, while Wang et al. (2024a) develops a reflection agent that provides feedback to refine plans.

Decision Maker: A Decision Maker provides the next operation(s) to control a device. Most studies (Lu et al., 2024a; Zhang et al., 2024a; Wen et al., 2024) define a set of UI-related actions—such as click, text, and scroll—as a basic action space. In a more complicated case, Ding (2024) encapsulates a sequence of actions to create Standard Operating Procedures(SOPs) to guide further operations.

As the power of GUI agents improves, the granularity of operations becomes more refined. Recent work has progressed from element-level operations (Zhang et al., 2023; Wang et al., 2024b) to coordinate-level controls (Wang et al., 2024a; Hong et al., 2024).

Executor: An Executor maps outputs to the relevant environments. While most studies use Android Debug Bridge (ADB) to control real devices (Li et al., 2024b; Wang et al., 2024a), Rawles et al. (2024) develops a simulator to access additional UI-related information.

Memory Retriever: A Memory Retriever is designed as an additional source of information to help agents perform tasks more effectively (Wang et al., 2024c).

GUI agents' memory is typically divided into internal and external categories. Internal memory (Lu et al., 2024a) consists of prior actions, screenshots, and system states during execution, while external memory (Zhang et al., 2023; Ding, 2024) includes knowledge and rules related to the UI or task, providing additional inputs for the agent.

3.2 (M)LLM-based GUI Agent Taxonomy

Consequently, this paper classifies existing work with the difference of input modality and learning mode in Figure 4.

3.2.1 GUI Agents with Different Input modality

LLM-based GUI Agents: With the limited multimodal capability, earlier GUI agents (Lee et al., 2023b; Li et al., 2020; Ma et al., 2023; Lai et al., 2024; Putta et al., 2024; Nakano et al., 2021) often require a GUI perceiver to convert GUI screens into text-based inputs.

So, parsing and grounding the GUI screens is the first step. For instance, Li et al. (2020) transforms the screen into a series of object descriptions and applies a transformer-based action mapping. The problem definitions and datasets have spurred further research. You et al. (2024) proposes a series of referring and grounding tasks, which provide valuable insights into the pre-training of GUIs. Lu et al. (2024b) proposes a screen parsing framework incorporating the local semantics of functionality with interactable region detection for better UI understanding and element grounding.

Afterward, LLMs are used as the brains of agents. Wen et al. (2024) further converts GUI screenshots into a simplified HTML representation for compatibility with the LLMs. By combining GUI representation with app-specific knowledge, they build Auto-Droid, a GUI agent based on online GPT and on-device Vicuna. In the field of web automation, LASER (Ma et al., 2023) navigates web environments purely through text, treating web navigation as state-space exploration to enable flexible state transitions and error recovery. Similarly, AutoWebGLM (Lai et al., 2024) processes HTML text data without visual inputs, refining webpage structures to preserve key information for ChatGLM3-6B. Agent Q (Putta et al., 2024) further extends this paradigm by relying solely on HTML DOM text for reasoning and decision-making, emphasizing language models for planning and action execution.

MLLM-based GUI Agents: Recent studies (Wang et al., 2024a; Bai et al., 2021; Zhang et al., 2023; Kim et al., 2023) utilize the multimodal capabilities of advanced (M)LLMs to improve GUI comprehension and task execution.

Leveraging the visual understanding capabilities of MLLMs, recent studies (Wang et al., 2024a; Li and Li, 2023; Bai et al., 2021; Zhu et al., 2024; Qin et al., 2025) explore end-to-end frameworks for GUI device control. For example, Spotlight (Li and Li, 2023) proposes a Vision-Language model framework, pre-trained on web/mobile data and fine-tuned for UI tasks. UIbert is a transformer-based joint image-text model, which is pre-trained in large-scale unlabeled GUI data to learn the feature representation of UI elements. Zhu et al. (2024) presents a two-level agent structure for executing complex and dynamic GUI tasks. Moba's Global Agent handles high-level planning, while the Local Agent selects actions for sub-tasks, streamlining the decision-making process with improved efficiency. UI-TARS (Qin et al., 2025) navigates interfaces through screenshots, enabling human-like interactions via keyboard and mouse. Leveraging a large-scale GUI dataset, it achieves context-aware UI understanding and precise captioning.

To enhance performance, some studies (Zhang et al., 2023; Rawles et al., 2024) utilize additional invisible metadata. For instance, AndroidWorld (Rawles et al., 2024) establishes a fully functional Android environment with real-world tasks, serving as a benchmark for evaluating GUI agents. They propose M3A, a zero-shot prompting agent that uses Set-of-Marks as input. Experiments with M3A variants assess how different input modalities—text, screenshots, and accessibility trees—affect GUI agent performance. Yang et al. (2024b) proposes a framework incorporating dynamic action history with both textual and interleaved text-image formats, which allows it to ground elements more effectively for dynamic, multi-step scenarios.

3.2.2 GUI Agents with Different Learning Mode

Prompting-based GUI Agents: Prompting is an effective approach to building agents with minimal extra computational overhead. Given the diversity of GUIs and tasks, numerous studies (Zhang et al., 2023; Li et al., 2024b; Wang et al., 2024a; Wen et al., 2023b; Xie et al., 2024; Zhang et al., 2024a; He et al., 2024a) use prompting to create GUI agents, adopting CoT or ReAct styles.

Recent studies use prompting to simulate the functions of GUI agent components. For example, Yan et al. (2023) introduces MM-Navigator, which utilizes GPT-4V for zero-shot GUI understanding and navigation. Additionally, Wen et al. (2023b) presents DroidBot-GPT, which summarizes the app's status, past actions, and tasks into a prompt, using ChatGPT to choose the next action.

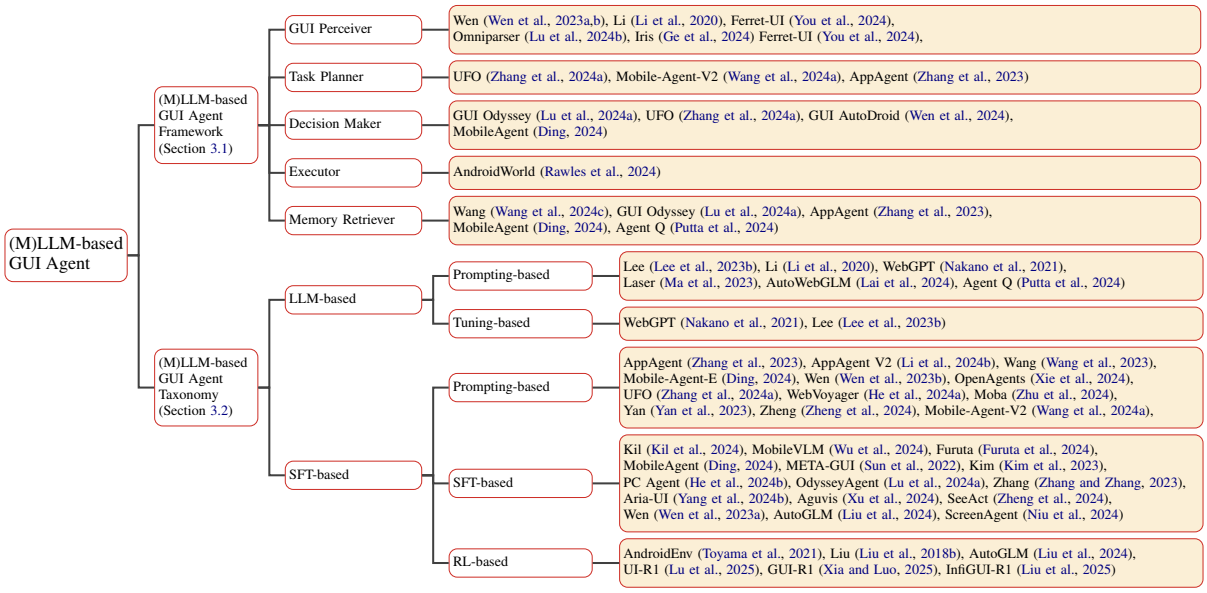

Figure 4: A comprehensive taxonomy of (M)LLM-based GUI Agents: frameworks, modality, and learning paradigms.

Beyond mobile applications, prompting-based approaches have also been widely adopted in web-based GUI agents. Zheng et al. (2024) proposes SeeAct, a GPT-4V-based generalist web agent. With screenshots as input, SeeAct generates action descriptions and converts them into executable actions with designed action grounding techniques. Similarly, WebVoyager (He et al., 2024a) integrates visual and textual information from screenshots and web pages, using prompts to interpret UI elements and execute interactions like clicking and typing. UFO (Zhang et al., 2024a) dynamically generates task plans and executes actions through prompting, allowing it to generalize across diverse web tasks without requiring task-specific adaptations.

Some research enhances GUI agent with external knowledge through prompting to complete tasks.

AppAgent (Zhang et al., 2023) proposes a multi-modal agent framework to simulate human-like mobile phone operations. The framework is divided into two phases: Exploration, where agents explore applications and document their operations, and Deployment, where these documents guide the agent in observing, thinking, acting, and summarizing tasks. This is the first work to claim human-like GUI automation capabilities. AppAgent V2 (Li et al., 2024b) further improves GUI parsing, document generation, and prompt integration by incorporating optical character recognition (OCR) and detection tools, moving beyond the limitations of off-the-shelf parsers for UI element identification. Wang et al. (2023) uses a pure in-context learning method to implement interaction between LLMs and mobile UIs. The method divides the conversations between agents and users into four categories from the originator and designs a series of structural CoT prompting to adapt an LLM to execute mobile UI tasks. MobileGPT (Lee et al., 2023b) emulates the cognitive processes of human use of applications to enhance the LLM-based agent with a human-like app memory. MobileGPT uses a random explorer to explore and generate screen-related subtasks on many apps and save them as app memory. During the execution, the related memory is recalled to complete tasks.

SFT-based GUI Agents: Supervised fine-tuning (SFT) allows (M)LLMs to adapt to specific domains and perform customized tasks. Recent studies on GUI agents (Wen et al., 2023a; Furuta et al., 2024; Niu et al., 2024; He et al., 2024b; Kil et al., 2024) demonstrate the benefits of SFT for GUI agents to process multi-modal inputs, learn specific procedures or execute specialized tasks.

For instance, Furuta et al. (2024) proposes WebGUM for web navigation. WebGUM is jointly fine-tuned with an instruction-optimized language model and a vision encoder, incorporating temporal and local perceptual capabilities. Zhang and Zhang (2023) introduces Auto-UI, a multimodal solution combining an image-language encoder-decoder architecture with a Chain of Actions policy, fine-tuned on the AitW dataset. This Chain of Actions

captures intermediate previous action histories and future action plans. Xu et al. (2024) introduces a two-stage training paradigm for AGUVIS. In the first stage, the agent learns visual representations of GUI components through self-supervised learning. In the second stage, it fine-tunes interactive tasks using reinforcement learning, enabling efficient autonomous GUI interaction. PC-Agent (He et al., 2024b) employs a multi-agent architecture, fine-tuning a planning agent on cognitive trajectories collected via PC Tracker, enabling it to model human cognitive patterns.

RL-based GUI Agents:

Early efforts (Liu et al., 2018a; Toyama et al., 2021) used reinforcement learning to enhance models for controling UI interfaces, but with limited success. To overcome the lack of offline data, AutoGLM adopted a self-evolving online curriculum reinforcement learning approach, and showed great ability for task exploration. More recently, the effectiveness of Group Relative Policy Optimization(GRPO) (Shao et al., 2024) has led to the development of several GRPO-based GUI Agents (Lu et al., 2025; Xia and Luo, 2025; Liu et al., 2025), which have been applied to grounding and reasoning tasks, however, lack GUI domain special algorithm improvement.

In summary, we provide a systematic overview of recent influential research on (M)LLM-based GUI agents. We address their goal formulations, input perceptions, and learning paradigms, as shown in Figure 4 in appendix.

4 GUI Agents Performance Analysis

Recent GUI agents have achieved marked advances in GUI understanding, grounding and execution. This paper analyses their results on the broadly recognized ScreenSpot and AndroidWorld benchmarks to provide an overview of the performance of GUI Agents with various technical designs.

Vanilla commercial and open-source multimodal foundation models perform poorly on UI understanding and grounding tasks. In contrast, models reported adding GUI-specific knowledge perform much better, even outperform early work SFT on GUI data. Among the latest SFT-based methods, exemplified by UI-Tars, outperform leading commercial models with arge-scale UI corpus construction. Recently, with GRPO becoming widely accepted, RL-based methods show a great advantage on mitigating decision-data sparsity. InfiGUI-R1 has achieved comparable performance to SOTA methods with only 3B parameters. The performance of representative methods is shown in Table 3 in Appendix.

Table 1: Android World Performance Comparison. "-" indicates missing values due to unavailable results in the original paper, unreleased model checkpoints, and unreleased inference code.

Model Name	Size	Screen Format	Success(%)
V-Droid (Llama8B)	8B	A11y tree	59.5
Agent S2 (Agashe et al., 2025)	-	Screenshot	54.3
UI-TARS (Qin et al., 2025)	72B	Screenshot	46.6
GPT-4o + Aria-UI	-	Screenshot	44.8
GPT-4o + UGround	-	Screenshot	44.0
GPT-4o	-	Screenshot	34.5
GPT-4 Turbo	-	A11y tree	30.6
InfiGUIAgent	2B	Screenshot	9.00
ShowUI-2B	2B	Screenshot	7.00
Qwen2-VL-2B	2B	Screenshot	0.00
Human	-	-	80.0

Compared to UI understanding and grounding, end-to-end task execution remains significantly more challenging for GUI agents. As shown in Table.1, even the latest research on the AndroidWorld benchmark still lags behind human performance. Among methods with provided models or APIs, UI-TARS, an example of the single-model approaches, achieves substantially better performance than previous work, however, smaller open-source SFT models still perform poorly. On the other hand, Multi-agent frameworks continue to dominate. Notably, AgentS2 achieves state-of-the-art performance through a combination of strategies such as enhanced visual perception and task planning. Furthermore, GPT-4o-based multi-agent frameworks substantially outperform their single-agent GPT-4o counterparts, highlighting the benefits of the Multi-agent frameworks design.

5 Industrial Applications of (M)LLM-Based GUI Agents

GUI agents have been widely used in industrial settings, such as mobile assistants and search agents, demonstrating significant potential.

Google Assistant for Android: By saying "Hey Google, start a run on Example App" users can use Google Assistant for Android to launch apps, perform tasks, and access content. App Actions, powered by built-in intents (BIIs), enhance app functionality by integrating with Google Assistant. This enables users to navigate apps and access features through voice queries, which the Assistant interprets to display the desired screen or widget.

Apple Intelligence: Apple Intelligence is the suite of AI-powered features and services developed by Apple. This includes technologies such as machine learning, natural language processing, and computer vision that power features like Siri, facial recognition, and photo organization. Apple also integrates AI into its hardware and software ecosystem to improve device performance and user experience. Their focus on privacy means that much of this AI processing happens on-device, ensuring that user data remains secure.

Anthropic Computer Use: Anthropic's "Computer Use" feature enables Claude to interact with tools and manipulate a desktop environment. By understanding and executing commands, Computer-Using Agent can perform the necessary actions to complete tasks, much like a human.

OpenAI Operator: OpenAI recently introduced Operator, an AI agent capable of autonomously performing tasks using its own browser. This agent leverages the CUA model, which combines GPT-4o's vision capabilities with advanced reasoning through reinforcement learning. Operator can interpret screenshots and interact with GUIs just as humans do.

AutoGLM: AutoGLM (Liu et al., 2024) is designed for autonomous mission completion via controlling GUIs on platforms like phones and the web. Its Android capability allows it to understand user instructions autonomously without manual input, enabling it to handle complex tasks such as ordering takeout, editing comments, shopping, and summarizing articles.

MagicOS 9.0 YOYO: An advanced assistant with four features: natural language and vision processing, user behavior learning, intent recognition and decision-making, and seamless app integration. It understands user habits to autonomously fulfill requests, such as ordering coffee through voice commands, by navigating apps and services.

6 Challenges

Due to the rapid development of this field, we summarize several key research questions that require urgent attention:

Personalized GUI Agents: Due to the personal nature of user devices, GUI agents inherently interact with personalized information. As an example, users may commute from home to work during weekdays, while walking to their favorite restaurants and cafes on weekends. The integration of personalized information would clearly enhance the user experience with GUI agents. As the capabilities of (M)LLMs continue to improve, personalized GUI agents have become a priority. Effectively collecting and utilizing personal information to deliver a more intelligent experience for users is an essential topic for future research and applications.

Security of GUI Agents: GUI devices play a crucial role in modern life, making the idea of allowing GUI agents to take control a significant concern for users. For instance, improper operations in financial apps could lead to substantial financial losses, while inappropriate comments on social media apps could damage one's reputation and privacy. Ensuring that GUI agents are not only highly efficient and capable of generalizing but also uphold user-specific security and provide transparency about their actions is an urgent research challenge. This is a critical issue, as it directly impacts the viability of applying GUI agents in real-world scenarios.

Inference Efficiency: Humans are highly sensitive to GUI response time, which significantly impacts the user experience. Current (M)LLM-based GUI agents still face notable drawbacks with inference latency. Additionally, communication delay is also an important consideration in real-world applications. As a result, efficient device-cloud collaboration strategies and effective device-side (M)LLM research will become critical areas of focus in the future.

7 Conclusion

In this paper, we provide a comprehensive review of the rapidly evolving field of (M)LLM-based GUI Agents. The review is organized into three main perspectives: Data Resources, Frameworks, and Applications. Additionally, we present a detailed taxonomy that connects existing research and highlights key techniques. We also discuss several challenges and propose potential future directions for GUI Agents that leverage foundation models.

Limitations

This paper provides a survey of GUI agents based on (M)LLMs. Several limitations should be noted: First, the review focuses on recent (M)LLM-based approaches for GUI interaction and does not cover

earlier methods based on traditional machine learning. Second, due to varying development speeds across areas, the included works are mostly centered on mobile and web applications, with fewer for PC. Finally, only some representative works in every topics are selected but not all relevant GUI agent studies are included.

References

Josh Achiam, Steven Adler, and 1 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Saaket Agashe, Kyle Wong, Vincent Tu, Jiachen Yang, Ang Li, and Xin Eric Wang. 2025. Agent s2: A compositional generalist-specialist framework for computer use agents. *arXiv preprint arXiv:2504.00906*.

Anthropic. 2024. Developing a computer use model. https://www.anthropic.com/news/developing-computer-use. Accessed: 2025-04-12.

Chen Bai, Xiaoyu Zang, Yan Xu, Srinivas Sunkara, Abhinav Rastogi, and Jieshan Chen. 2021. Uibert: Learning generic multimodal representations for ui understanding.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and 1 others. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Dongping Chen, Yue Huang, Siyuan Wu, and 1 others. 2024a. Gui-world: A dataset for gui-oriented multimodal llm-based agents. *arXiv preprint arXiv:2406.10819*.

Jingxuan Chen, Derek Yuen, Bin Xie, and 1 others. 2024b. Spa-bench: A comprehensive benchmark for smartphone agent evaluation. In *NeurIPS 2024 Workshop on Open-World Agents*.

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

Google DeepMind. 2024. Gemini-2.0 (project mariner). https://deepmind.google/technologies/project-mariner. Accessed: 2025-04-12.

Tinghe Ding. 2024. Mobileagent: enhancing mobile control via human-machine interaction and sop integration. *arXiv preprint arXiv:2401.04124*.

Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, and 1 others. 2024. Multimodal web navigation with instruction-finetuned foundation models. In *ICLR*.

Zhiqi Ge, Juncheng Li, Xinglei Pang, and 1 others. 2024. Iris: Breaking gui complexity with adaptive focus and self-refining. *arXiv preprint arXiv:2412.10342*.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. 2025. Navigating the digital world as humans do: Universal visual grounding for gui agents. In *13th International Conference on Learning Representations, ICLR 2025*.

Hongliang He, Wenlin Yao, Kaixin Ma, and 1 others. 2024a. Webvoyager: Building an end-to-end web agent with large multimodal models. *arXiv preprint arXiv:2401.13919*.

Yanheng He, Jiahe Jin, Shijie Xia, and 1 others. 2024b. Pc agent: While you sleep, ai works–a cognitive journey into digital world. *arXiv preprint arXiv:2412.17589*.

Wenyi Hong, Weihan Wang, Qingsong Lv, and 1 others. 2024. Cogagent: A visual language model for gui agents. In *CVPR*, pages 14281–14290.

Jihyung Kil, Chan Hee Song, Boyuan Zheng, Xiang Deng, Yu Su, and Wei-Lun Chao. 2024. Dual-view visual contextualization for web navigation. In *CVPR*, pages 14445–14454.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. 2023. Language models can solve computer tasks. In *NIPS*, pages 39648–39677.

Hanyu Lai, Xiao Liu, Iat Long Iong, and 1 others. 2024. Autowebglm: A large language model-based web navigating agent. In *SIGKDD*, pages 5295–5306.

Kenton Lee, Mandar Joshi, Iulia Raluca Turc, and 1 others. 2023a. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *ICML*, pages 18893–18912.

Sunjae Lee, Junyoung Choi, Jungjae Lee, and 1 others. 2023b. Explore, select, derive, and recall: Augmenting llm with human-like memory for mobile task automation. *arXiv preprint arXiv:2312.03003*.

Gang Li and Yang Li. 2023. Spotlight: Mobile ui understanding using vision-language models with a focus. In *ICLR*.

Wei Li, William Bishop, Alice Li, and 1 others. 2024a. On the effects of data scale on computer control agents. *arXiv preprint arXiv:2406.03679*.

Yanda Li, Chi Zhang, Wanqi Yang, and 1 others. 2024b. Appagent v2: Advanced agent for flexible mobile interactions. *arXiv preprint arXiv:2408.11824*.

Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, and Jason Baldridge. 2020. Mapping natural language instructions to mobile ui action sequences. In *ACL*, pages 8198–8210.

Yang Li, Gang Li, Xin Zhou, Mostafa Dehghani, and Alexey Gritsenko. 2021. Vut: Versatile ui transformer for multi-modal multi-task user interface modeling. *arXiv preprint arXiv:2112.05692*.

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

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. 2018a. Reinforcement learning on web interfaces using workflow-guided exploration. In *ICLR*.

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. 2018b. Reinforcement learning on web interfaces using workflow-guided exploration. *arXiv preprint arXiv:1802.08802*.

Xiao Liu, Bo Qin, Dongzhu Liang, and 1 others. 2024. Autoglm: Autonomous foundation agents for guis. *arXiv preprint arXiv:2411.00820*.

Yuhang Liu, Pengxiang Li, Congkai Xie, Xavier Hu, Xiaotian Han, Shengyu Zhang, Hongxia Yang, and Fei Wu. 2025. Infigui-r1: Advancing multimodal gui agents from reactive actors to deliberative reasoners. *arXiv preprint arXiv:2504.14239*.

Quanfeng Lu, Wenqi Shao, Zitao Liu, and 1 others. 2024a. Gui odyssey: A comprehensive dataset for cross-app gui navigation on mobile devices. *arXiv preprint arXiv:2406.08451*.

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

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

Kaixin Ma, Hongming Zhang, Hongwei Wang, Xiaoman Pan, and Dong Yu. 2023. Laser: Llm agent with state-space exploration for web navigation. In *NeurIPS Workshop*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, and 1 others. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Runliang Niu, Jindong Li, Shiqi Wang, and 1 others. 2024. Screenagent: A vision language model-driven computer control agent. *arXiv preprint arXiv:2402.07945*.

OpenAI. 2024. Gpt-4o. Accessed: 2025-01-03.

Lihang Pan, Bowen Wang, Chun Yu, Yuxuan Chen, Xiangyu Zhang, and Yuanchun Shi. 2023. Autotask: Executing arbitrary voice commands by exploring and learning from mobile gui. *arXiv preprint arXiv:2312.16062*.

Pranav Putta, Edmund Mills, Naman Garg, and 1 others. 2024. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*.

Yujia Qin, Yining Ye, Junjie Fang, and 1 others. 2025. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, and 1 others. 2024. Androidworld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy P Lillicrap. 2023. Androidinthewild: A large-scale dataset for android device control. In *NIPS Datasets and Benchmarks Track*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Huawen Shen, Chang Liu, Gengluo Li, and 1 others. 2024. Falcon-ui: Understanding gui before following user instructions. *arXiv preprint arXiv:2412.09362*.

Liangtai Sun, Xingyu Chen, Lu Chen, Tianle Dai, Zichen Zhu, and Kai Yu. 2022. Meta-gui: Towards multi-modal conversational agents on mobile gui. In *EMNLP*, pages 6699–6712.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, and 1 others. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Daniel Toyama, Philippe Hamel, Anita Gergely, and 1 others. 2021. Androidenv: A reinforcement learning platform for android. *arXiv preprint arXiv:2105.13231*.

Sagar Gubbi Venkatesh, Partha Talukdar, and Srini Narayanan. 2024. Ugif-dataset: A new dataset for cross-lingual, cross-modal sequential actions on the ui. In *Findings of NAACL*, pages 1390–1399.

Bryan Wang, Gang Li, and Yang Li. 2023. Enabling conversational interaction with mobile ui using large language models. In *CHI*, pages 1–17.

Bryan Wang, Gang Li, Xin Zhou, Zhourong Chen, Tovi Grossman, and Yang Li. 2021. Screen2words: Automatic mobile ui summarization with multimodal learning. In *UIST*, pages 498–510.

Junyang Wang, Haiyang Xu, Haitao Jia, and 1 others. 2024a. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. In *NIPS*.

Junyang Wang, Haiyang Xu, Jiabo Ye, and 1 others. 2024b. Mobile-agent: Autonomous multi-modal mobile device agent with visual perception. *arXiv preprint arXiv:2401.16158*.

Lei Wang, Chen Ma, Xueyang Feng, and 1 others. 2024c. A survey on large language model based autonomous agents. *FCS*, 18(6):186345.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, and 1 others. 2024d. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Zhenhailong Wang, Haiyang Xu, Junyang Wang, and 1 others. 2025. Mobile-agent-e: Self-evolving mobile assistant for complex tasks. *arXiv preprint arXiv:2501.11733*.

Hao Wen, Yuanchun Li, Guohong Liu, and 1 others. 2023a. Empowering LLM to use Smartphone for Intelligent Task Automation. *arXiv preprint arXiv:2308.15272*.

Hao Wen, Yuanchun Li, Guohong Liu, and 1 others. 2024. Autodroid: Llm-powered task automation in android. In *MobiCom*, pages 543–557.

Hao Wen, Hongming Wang, Jiaxuan Liu, and Yuanchun Li. 2023b. Droidbot-gpt: Gpt-powered ui automation for android. *arXiv preprint arXiv:2304.07061*.

Qinzhuo Wu, Weikai Xu, Wei Liu, and 1 others. 2024. Mobilevlm: A vision-language model for better intra- and inter-ui understanding. In *EMNLP*, pages 10231–10251.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and 1 others. 2025. Os-atlas: A foundation action model for generalist gui agents. In *13th International Conference on Learning Representations, ICLR 2025*.

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

Tianbao Xie, Fan Zhou, and 1 others. 2024. Openagents: An open platform for language agents in the wild. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Yiheng Xu, Zekun Wang, Junli Wang, and 1 others. 2024. Aguvis: Unified pure vision agents for autonomous gui interaction. *arXiv preprint arXiv:2412.04454*.

An Yan, Zhengyuan Yang, Wanrong Zhu, and 1 others. 2023. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation. *arXiv preprint arXiv:2311.07562*.

An Yang, Baosong Yang, and 1 others. 2024a. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Yuhao Yang, Yue Wang, Dongxu Li, and 1 others. 2024b. Aria-ui: Visual grounding for gui instructions. *arXiv preprint arXiv:2412.16256*.

Keen You, Haotian Zhang, Eldon Schoop, and 1 others. 2024. Ferret-ui: Grounded mobile ui understanding with multimodal llms. In *ECCV*, pages 240–255.

Chaoyun Zhang, Liqun Li, Shilin He, and 1 others. 2024a. Ufo: A ui-focused agent for windows os interaction. *arXiv preprint arXiv:2402.07939*.

Chi Zhang, Zhao Yang, Jiaxuan Liu, and 1 others. 2023. AppAgent: Multimodal Agents as Smartphone Users. *arXiv preprint arXiv:2312.13771*.

Jiwen Zhang, Jihao Wu, Yihua Teng, and 1 others. 2024b. Android in the zoo: Chain-of-action-thought for gui agents. *arXiv preprint arXiv:2403.02713*.

Xiaoyi Zhang, Lilian De Greef, Amanda Swearngin, and 1 others. 2021. Screen recognition: Creating accessibility metadata for mobile applications from pixels. In *CHI*, pages 1–15.

Zhuosheng Zhang and Aston Zhang. 2023. You only look at screens: Multimodal chain-of-action agents. *arXiv preprint arXiv:2309.11436*.

Kangjia Zhao, Jiahui Song, Leigang Sha, and 1 others. 2024. Gui testing arena: A unified benchmark for advancing autonomous gui testing agent. *arXiv preprint arXiv:2412.18426*.

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

Shuyan Zhou, Frank F Xu, Hao Zhu, and 1 others. 2023. Webarena: A realistic web environment for building autonomous agents. In *ICLR*.

Zichen Zhu, Hao Tang, Yansi Li, and 1 others. 2024. Moba: A two-level agent system for efficient mobile task automation. *arXiv preprint arXiv:2410.13757*.

Table 2: Overview of (M)LLM-Based GUI Agents.

Model Name	Category	GUI Perceiver	Learning Method	Base Model	Scenarios
Prompting-based					
PaLM (Wang et al., 2023)	Single Step	HTML	Few-shot prompting	PaLM	Mobile
MM-Navigator (Yan et al., 2023)	Single Step	Screenshot	Zero-shot prompting	GPT-4V	Mobile
MemoDroid (Lee et al., 2023b)	End-to-End	HTML	Few-shot prompting	ChatGPT/GPT-4V	Mobile/Desktop
AutoTask (Pan et al., 2023)	End-to-End	Screenshot/API	Zero-shot prompting	GPT-4V	Mobile
AppAgent (Zhang et al., 2023)	End-to-End	Screenshot	Exploration-based/In-context learning	GPT4V	Mobile
DroidBot-GPT (Wen et al., 2023b)	End-to-End	Screenshot	Zero-shot prompting	ChatGPT	Mobile
Mobile-Agent-V2 (Wang et al., 2024a)	End-to-End	Screenshot	Zero-shot prompting	GPT4V	Mobile
SeeAct (Zheng et al., 2024)	End-to-End	Screenshot/HTML	Few-shot prompting	GPT-4V	Web
Mobile-Agent-E (Wang et al., 2025)	End-to-End	Screenshot	Zero-shot prompting	GPT-4o/Claude-3.5-Sonnet/Gemini-1.5-pro	Mobile
Learning-based					
Spotlight (Li and Li, 2023)	UI modeling	Screenshot	Pretrain/SFT	ViT	Mobile/Web
Pix2Struct (Lee et al., 2023a)	UI modeling	Screenshot	Pretrain/SFT	ViT	Web
VUT (Li et al., 2021)	UI modeling	Screenshot	SFT	Transformer	Mobile/Web
Screen Recognition (Zhang et al., 2021)	UI modeling	Screenshot	SFT	Faster R-CNN	Mobile
Screen2Words (Wang et al., 2021)	UI modeling	Screenshot	SFT	Transformer	Mobile
Aria-UI (Yang et al., 2024b)	UI modeling	Screenshot	Pretrain/SFT	Aria	Mobile/Web/Desktop
Ferret-UI (You et al., 2024)	UI modeling	Screenshot	Pretrain/SFT	Ferret	Mobile
AutoDroid (Wen et al., 2024)	End-to-End	HTML	Exploration-based/SFT	Vicuna-7B	Mobile
Seq2Act (Li et al., 2020)	End-to-End	Texts	Supervised learning	Transformer	Mobile
Meta-GUI (Sun et al., 2022)	End-to-End	Screenshot/XML	Supervised learning	Transformer	Mobile
Agent Q (Putta et al., 2024)	End-to-End	Screenshot/DOM	RL/BC Training	Transformer	Web
WebGUM (Furuta et al., 2024)	End-to-End	Screenshot/HTML	SFT	Flan-T5	Web
CogAgent (Hong et al., 2024)	End-to-End	Screenshot	SFT	CogVLM	Mobile/Desktop
MobileVLM (Wu et al., 2024)	End-to-End	XML/Screenshot	Pretrain/SFT	Qwen-VL-Chat	Mobile
WebGPT (Nakano et al., 2021)	End-to-End	Texts	SFT	GPT-3	Web
AutoGLM (Liu et al., 2024)	End-to-End	Screenshot/HTML	Pretrain/SFT/RL	ChatGLM	Mobile/Web
OdysseyAgent (Lu et al., 2024a)	End-to-End	Screenshot	SFT	Qwen-VL	Mobile

Table 3: Performance on ScreenSpot across Mobile, Desktop, and Web. "-" indicates missing values due to unavailable results in the original paper, unreleased model checkpoints, and unreleased inference code.

Model Name	**Accuracy (%)**						**Avg.**
	Mobile		Desktop		Web		
	Text	Icon	Text	Icon	Text	Icon	
Proprietary Models							
GPT-4o (OpenAI, 2024)	30.5	23.2	20.6	19.4	11.1	7.8	18.8
Claude Computer Use (Anthropic, 2024)	-	-	-	-	-	-	83.0
Gemini 2.0 (Project Mariner) (DeepMind, 2024)	-	-	-	-	-	-	84.0
General Open-source Models							
Qwen2-VL-7B (Wang et al., 2024d)	61.3	39.3	52.0	45.0	33.0	21.8	42.9
Qwen2.5-VL-3B (Bai et al., 2025)	-	-	-	-	-	-	55.5
Qwen2.5-VL-7B (Bai et al., 2025)	-	-	-	-	-	-	84.7
GUI-specific Models (SFT)							
CogAgent-18B (Hong et al., 2024)	67.0	24.0	74.2	20.0	70.4	28.6	47.4
SeeClick-9.6B (Cheng et al., 2024)	78.0	52.0	72.2	30.0	55.7	32.5	53.4
UGround-7B (Gou et al., 2025)	82.8	60.3	82.5	63.6	80.4	70.4	73.3
OS-Atlas-7B (Wu et al., 2025)	93.0	72.9	91.8	62.9	90.9	74.3	82.5
ShowUI-2B (Lin et al., 2024)	92.3	75.5	76.3	61.1	81.7	63.6	75.1
Aguvis-7B (Xu et al., 2024)	95.6	77.7	93.8	67.1	88.3	75.2	84.4
UI-TARS-7B (Qin et al., 2025)	94.5	89.2	95.9	85.7	90.0	83.5	89.5
UI-TARS-72B (Qin et al., 2025)	94.9	82.5	89.7	88.6	88.7	85.0	88.4
GUI-specific Models (RL)							
UI-R1-3B (Lu et al., 2025)	-	-	90.2	59.3	85.2	73.3	-
GUI-R1-3B (Xia and Luo, 2025)	-	-	93.8	64.8	89.6	72.1	-
GUI-R1-7B (Xia and Luo, 2025)	-	-	91.8	73.6	91.3	75.7	-
InfiGUI-R1-3B (Liu et al., 2025)	<u>97.1</u>	<u>81.2</u>	<u>94.3</u>	<u>77.1</u>	**91.7**	<u>77.6</u>	<u>87.5</u>