Reinforcement Learning in GUI Agents: A Survey Toward Digital Inhabitants

Anonymous authors
Paper under double-blind review

Abstract

We present the first comprehensive overview of the intersection of Reinforcement Learning (RL) and GUI agents—intelligent systems that perceive graphical interfaces visually and execute tasks through simulated human inputs—and examine how this line of research may evolve toward digital inhabitants. While Multimodal Large Language Models provide the perceptual backbone, supervised fine-tuning alone cannot address the core difficulties of GUI automation: long-horizon credit assignment under sparse, delayed rewards, distribution shift across continuously evolving interfaces, and safe exploration in environments where erroneous actions carry irreversible consequences. We propose a principled taxonomy organizing existing methods into *Offline RL*, *Online RL*, and *Hybrid Strategies*, complemented by analyses of reward engineering, data efficiency, and technical innovations. Our analysis highlights critical ongoing shifts: the tension between reliability and scalability is driving the adoption of composite multi-tier reward architectures, while GUI I/O latency bottlenecks are catalyzing a move toward world-model-based training that achieves substantial performance gains. Additionally, the spontaneous emergence of System-2-style deliberation suggests that explicit reasoning supervision may be unnecessary given sufficiently rich reward signals. We distill these findings into a roadmap addressing process rewards, continual RL, cognitive architectures, and safe deployment to guide the next generation of robust GUI automation and the agent-native infrastructure it may ultimately require.

1 Introduction

GUI agents—intelligent systems that perceive graphical user interfaces visually and execute tasks through simulated human inputs such as mouse clicks, typing, and scrolling—sit at the frontier of a paradigm shift in AI: from passive information processing to active task execution in real digital environments. Unlike traditional automation tools such as Selenium (Selenium Contributors, 2023) or Robotic Process Automation (RPA) (Van der Aalst et al., 2018), which depend on brittle DOM selectors or rigid coordinate scripts, modern GUI agents leverage visual perception and semantic reasoning to generalize across heterogeneous operating systems, dynamic web pages, and previously unseen applications. Training such agents to operate robustly in these complex, stochastic environments requires **Reinforcement Learning (RL)**.

Why RL? A formal argument. Consider a GUI task formalized as a Partially Observable Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$. The agent observes a screenshot $s_t \in \mathcal{S}$, executes an action $a_t \in \mathcal{A}$ (e.g., $\texttt{click}(x, y)$, $\texttt{type}(\text{“query”})$), and receives a reward $r_t = \mathcal{R}(s_t, a_t)$. In realistic GUI environments, three properties make supervised fine-tuning (SFT) alone insufficient and RL necessary:

1. **Sparse, delayed rewards.** The reward signal is typically binary—$r_T = +1$ upon task completion, $r_t = 0$ for all intermediate steps $t < T$—with trajectory lengths T ranging from 50 to 200 steps. No intermediate supervision is available for sub-step correctness. This creates a severe credit assignment problem: $\nabla_\theta J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t | s_t, \mathcal{H}_t) \cdot \hat{A}_t \right]$, where the advantage $\hat{A}_t$ must propagate reward information across dozens of steps with non-Markovian observations.

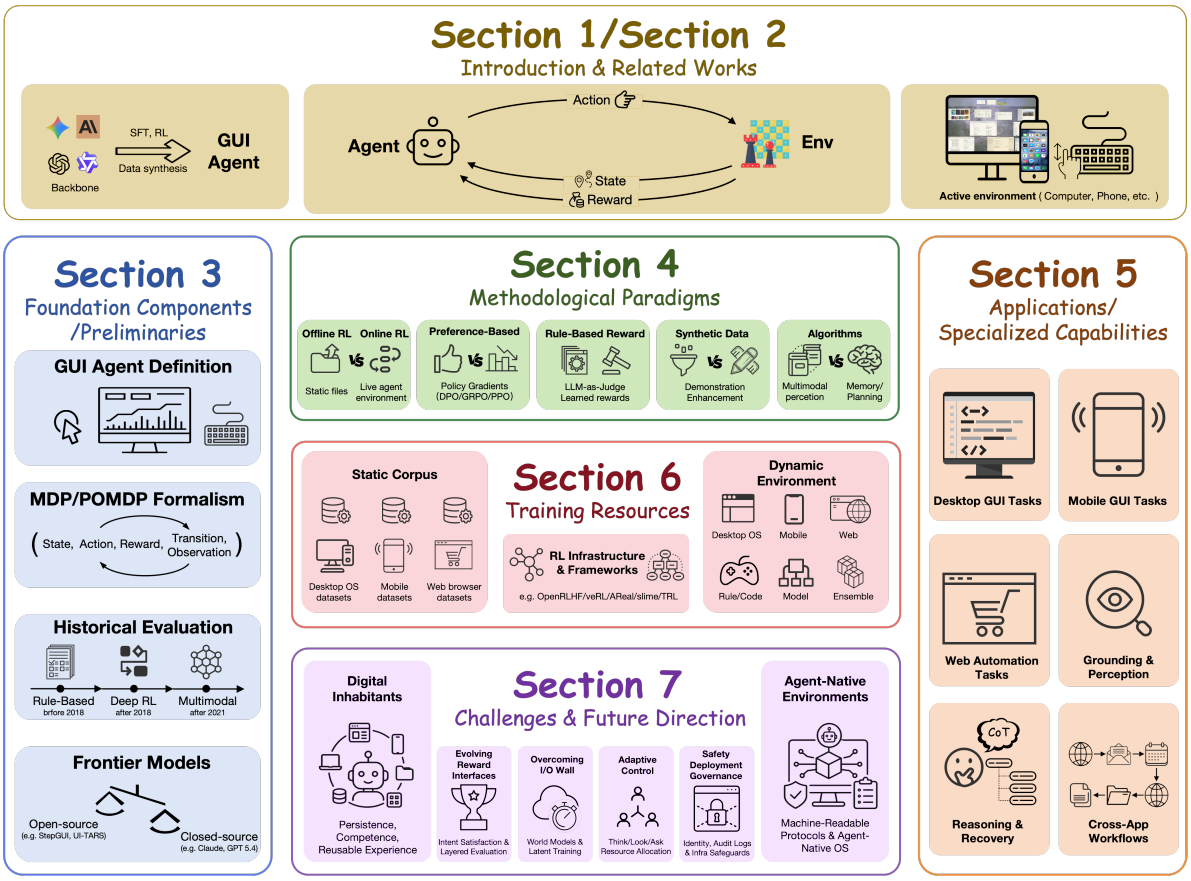

Figure 1: Overview of the survey structure. Sections 1 and 2 introduce the background of GUI agent and related works; Section 3 presents the preliminaries; Section 4 reviews RL methodologies; Section 5 summarizes key dimensions and applications; Section 6 surveys training resources; and Section 7 discusses challenges and future directions.

2. **Distribution shift and environment non-stationarity.** Static demonstration datasets capture specific UI versions, but real interfaces evolve continuously through updates, A/B testing, and redesigns. Behavioral cloning from such data suffers from compounding covariate shift: a single out-of-distribution action at step t cascades into unrecoverable failure states, with error probability growing as $\mathcal{O}(\epsilon T)$ where ϵ is the per-step imitation error.

3. **Optimization beyond human imitation.** RL enables agents to discover execution paths more efficient than human demonstrations. While imitation learning is bounded by $J(\pi_{\text{IL}}) \leq J(\pi_{\text{expert}})$, RL with well-designed rewards can discover policies π^* satisfying $J(\pi^*) > J(\pi_{\text{expert}})$, as demonstrated by GUI agents finding shorter task completion paths through self-play.

Why now? GUI as the ultimate RL laboratory. Beyond addressing the aforementioned structural challenges, the current acceleration of RL in GUI agents is driven by a fundamental property distinguishing them from pure text LLMs: **verifiability**. In plain text generation, defining an "absolute correct" response is notoriously elusive, rendering reward modeling subjective and prone to hallucination or reward hacking. In striking contrast, GUI environments yield objective, measurable ground truths—a successfully navigated URL, a predictably altered DOM tree, or a confirmed database transaction. This structured verifiability transforms the GUI into the perfect testbed for Reinforcement Learning with Verifiable Rewards (RLVR). Consequently, applying RL here transcends being a mere "patch solution" for imitation learning shortcomings; it establishes a closed-loop evolutionary path toward general intelligent behavior, where

agents can autonomously interact, receive absolute environmental feedback, and continuously self-improve. More broadly, GUI agents may serve as the transitional substrate through which AI systems evolve from task-specific operators of human software into persistent digital inhabitants, while revealing what future agent-native infrastructure should look like.

RL's proven track record. The case for RL in GUI automation builds on a series of validated breakthroughs. AlphaGo (Silver et al., 2016) and AlphaZero (Silver et al., 2017) demonstrated that RL with binary win/loss rewards could surpass human experts through self-play—precisely the regime of sparse terminal feedback that GUI tasks inhabit. RLHF (Ouyang et al., 2022) and DPO (Rafailov et al., 2023) showed that RL can optimize for objectives too nuanced for explicit labels, enabling GPT-4 (Achiam et al., 2023) and Claude (Bai et al., 2022) to align with complex human preferences. Most recently, reasoning-focused models like OpenAI o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025a) employed RL with Verifiable Rewards (RLVR) to achieve breakthrough performance on mathematical and multi-step reasoning tasks. GUI agents share all the characteristics that made RL successful in these domains: sequential decision-making under uncertainty, sparse delayed feedback, and the need to generalize beyond static training distributions.

GUI agents vs. CLI agents: The necessity of visual interaction. While command-line interface (CLI) agents excel in efficiency and determinism for expert-oriented workflows—where structured text streams and explicit exit codes provide unambiguous feedback (Yang et al., 2023; Gandhi et al., 2026)—they face fundamental coverage limitations. CLI agents operate exclusively within systems exposing programmatic APIs or terminal interfaces, typically constraining them to software engineering, system administration, and database operations (Bechard et al., 2026; Li et al., 2026). In contrast, the vast majority of real-world digital systems remain inaccessible via APIs: legacy enterprise software in finance and healthcare, proprietary industrial control systems, and countless closed-source applications built atop graphical interfaces without programmatic access (Van der Aalst et al., 2018).

GUI agents address this **long-tail coverage problem** through visual perception. By reasoning over pixel-level observations or accessibility trees, they can interact with *any* system a human can operate—regardless of whether backend APIs exist. Modern unified frameworks like ClawGUI (Tang et al., 2026) demonstrate this universality across diverse platforms and applications. However, this generality comes at a cost: GUI environments exhibit higher observational complexity (high-dimensional pixel arrays vs. structured text), lower feedback determinism (visual confirmation vs. binary exit codes), and substantially greater computational overhead (Hong et al., 2024; Zhang et al., 2025c). Despite these trade-offs being unavoidable when automating the estimated 60–80% of enterprise workflows locked behind graphical interfaces (Van der Aalst et al., 2018), GUI agents capture richer contextual signals—color-coded warnings, disabled buttons, spatial layout—that enable semantic error recovery impossible in text-only environments. As we demonstrate throughout this survey, RL provides the principled framework to navigate this complexity, transforming visual perception into robust task execution despite sparse rewards and dynamic interface evolution.

Scope and contributions. This survey provides the comprehensive analysis of RL techniques within the GUI agent domain, our contributions are:

- **A principled RL taxonomy for GUI agents.** We categorize methods into three paradigms—offline RL, online RL, and hybrid strategies—and identify the dominant algorithmic families (DPO for offline RFT, GRPO for online/hybrid) together with their theoretical motivations and practical trade-offs.

- **Cross-cutting dimensional analysis.** We analyze three critical dimensions that cut across all paradigms: reward engineering (a three-tier pyramid from rule-based to LLM-as-Judge to learned rewards), data efficiency (world models, demonstration enhancement, self-improvement), and technical innovations in perception, memory, and multi-turn optimization.

- **Actionable insights and roadmap.** We distill three key findings—the reliability–scalability tension in reward design, the I/O latency wall driving world-model adoption, and the emergence of

reasoning from structured action spaces—into a concrete research roadmap for the field, culminating in a broader perspective on digital inhabitants and the agent-native infrastructure they may ultimately require.

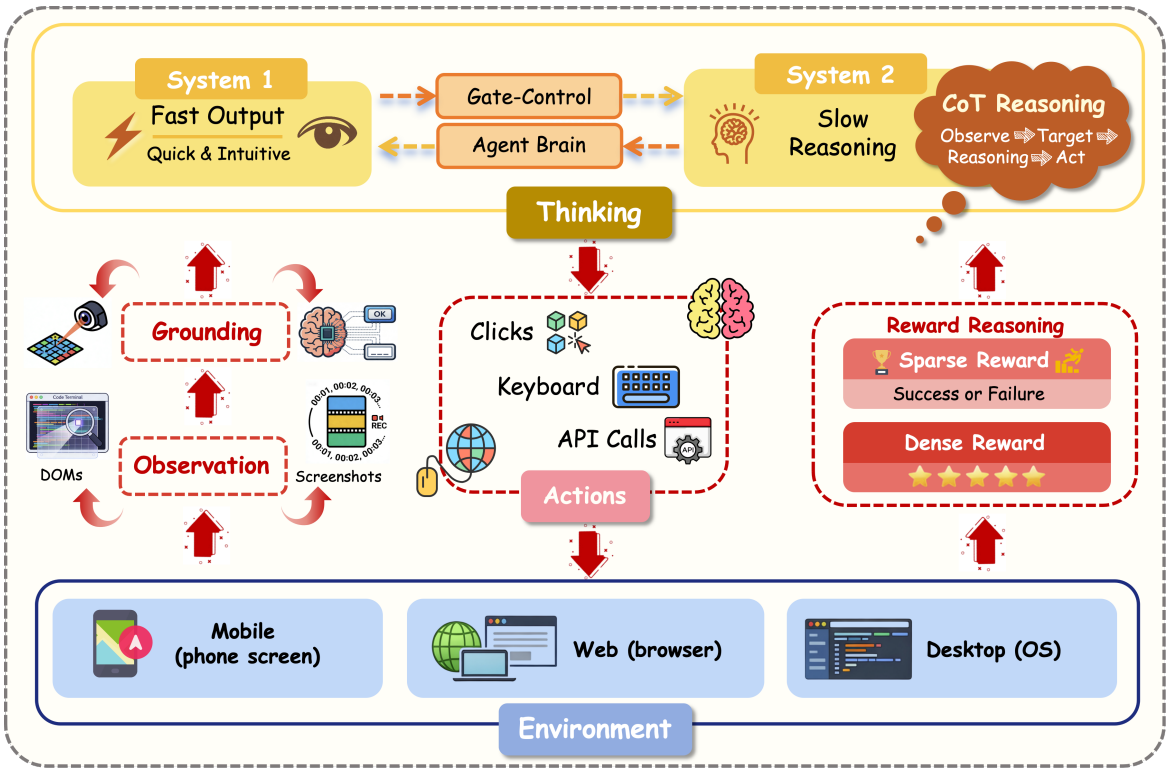

Figure 2: Overview of the RL training pipeline for GUI agents. The agent perceives the GUI environment through screenshots, reasons about the task, and executes actions. RL optimizes the policy through reward signals derived from task completion, visual grounding accuracy, and intermediate reasoning quality.

2 Related Works

Surveys on RL for LLM alignment and reasoning. A growing body of work covers Reinforcement Learning for Large Language Models in text-centric domains—alignment via RLHF and DPO, reasoning enhancements for models like OpenAI o1 and DeepSeek-R1 that use MCTS and process rewards, and multi-agent RL (Sun et al., 2024). These surveys address static, text-only generation, which differs fundamentally from the multimodal, interactive setting of GUI agents where actions require precise visual grounding and real-time interface manipulation (Cao et al., 2024; Wang et al., 2024c; Zhang et al., 2025e;g).

Surveys on GUI agents. Recent reviews of autonomous GUI agents focus predominantly on architectural components (visual encoders, grounding modules) and supervised fine-tuning strategies, with some examining the trade-offs between API-based and GUI-based agents (Zhang et al., 2025a). While these works provide broad coverage of benchmarks and model architectures, they treat RL peripherally—lacking systematic analysis of reward engineering, exploration mechanisms, and the credit-assignment challenges unique to long-horizon GUI decision-making (Nguyen et al., 2025; Hu et al., 2025; Liu et al., 2025a; Ning et al., 2025; Sager et al., 2025; Tang et al., 2025c; Wang et al., 2024b; Zhang et al., 2024b).

Gap and our contribution. To our knowledge, this is the first survey exclusively targeting the intersection of **Reinforcement Learning and GUI Agents**. We differentiate our work along three axes:

- We introduce a detailed taxonomy of RL paradigms adapted for GUI automation—offline, online, and hybrid—with cross-cutting analyses of reward engineering, data efficiency, and technical innovations.
- We conduct a systematic analysis of reward engineering techniques (rule-based, LLM-as-judge, learned) tailored to the sparse, delayed feedback characteristic of GUI tasks.
- We synthesize the rapid methodological advances of 2024–2026, offering a structured roadmap for this emerging field.

3 Preliminaries

3.1 Definition of GUI Agents

GUI agents are intelligent systems capable of autonomously completing complex, cross-application tasks by perceiving on-screen information visually and simulating mouse and keyboard operations (such as clicking, typing, and scrolling), much like a human user.

3.2 The MDP Formalism for GUI Agents

We formally model the interaction between the agent and the digital environment as a Partially Observable Markov Decision Process (POMDP), often approximated as a Markov Decision Process (MDP) augmented with interaction history. This framework is defined by the tuple:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma, \mathcal{H})$$

The state space $\mathcal{S}$ encompasses the multimodal observations of the GUI, typically represented by high-resolution screenshots (pixel space) or structured metadata such as the Document Object Model (DOM) or accessibility trees. The action space $\mathcal{A}$ consists of the set of permissible low-level interface operations—including coordinate-based clicks (x, y), keyboard typing, scrolling, and drag-and-drop gestures—that mimic human input modalities. The environment dynamics are governed by the transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}'$, which is often stochastic due to network latency, dynamic page rendering, and asynchronous UI updates.

A critical challenge in this domain is the reward signal $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$. In standard settings, rewards are sparse and binary (+1 for successful task completion, 0 otherwise), necessitating the development of auxiliary dense reward functions to facilitate efficient learning. Given the sequential and context-dependent nature of GUI tasks, the decision-making process relies heavily on the history trajectory:

$$\mathcal{H}_t = \{s_0, a_0, \ldots, s_{t-1}, a_{t-1}\}$$

which captures temporal dependencies essential for resolving non-Markovian ambiguities. The objective of the GUI agent is to learn a policy $\pi(a_t \mid s_t, \mathcal{H}_t)$ that maximizes the expected cumulative discounted reward:

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{T} \gamma^t \mathcal{R}(s_t, a_t) \right]$$

where $\gamma \in [0, 1]$ is the discount factor balancing immediate and future returns.

3.3 Reinforcement Learning

Reinforcement Learning (RL) is a computational framework for learning optimal decision-making strategies through interaction with an environment. Unlike supervised learning, which relies on static datasets of labeled examples, RL optimizes a policy π by maximizing a cumulative reward signal gathered through trial-and-error exploration.

Within the GUI agent domain, RL is essential for solving non-Markovian, long-horizon tasks where the optimal action depends on complex history dependencies and feedback is often sparse (i.e., received only

upon task completion). RL algorithms are typically categorized by their usage of data (online vs. offline) and their optimization target (value-based vs. policy-based). Most state-of-the-art GUI agents employ policy gradient methods (e.g., PPO (Schulman et al., 2017), GRPO (Shao et al., 2024)) to fine-tune Multimodal Large Language Models (MLLMs), treating the model as a policy π_θ that maps visual observations to executable actions.

3.4 Background and Historical Evolution

The trajectory of this field illustrates a shift from the broader concept of **Computer-Using Agents (CUAs)** to the specific paradigm of **GUI Agents**. While CUAs encompass any autonomous system operating a computer—including those relying on backend APIs, CLI commands, or hidden DOM states—GUI agents specifically target interaction through the visual frontend, mimicking human perception (vision) and actuator control (mouse/keyboard). This evolution reflects a transition from efficient but brittle backend automation to robust, general-purpose interaction that utilizes software exactly as humans do. We trace this development through three distinct phases.

Phase 1: Rule-based automation (1990s–2010s). Early UI automation relied on hard-coded scripts and Robotic Process Automation (RPA) tools that strictly replayed recorded sequences of coordinates or accessibility API calls. While effective for repetitive, static workflows, these systems were brittle: minor interface changes—such as a shifted button or a renamed text field—would cause catastrophic failure. They lacked perception and could not adapt to dynamic content (Pasupat et al., 2018).

Phase 2: Deep reinforcement learning in isolated environments (2015–2022). With the rise of Deep Q-Networks (DQN) (Mnih et al., 2013) and policy gradients, researchers began applying RL to GUI navigation. Early works like World of Bits (Shi et al., 2017) and MiniWoB++ (Liu et al., 2018a) demonstrated that agents could learn to interact with web elements via trial and error. However, these agents were typically trained on small, synthetic environments using shallow neural networks. They struggled with generalization, often overfitting to specific DOM structures or visual layouts, and lacked the semantic understanding to process complex natural language instructions.

Phase 3: The multimodal LLM era (2023–present). The current breakthrough is driven by the integration of Multimodal Large Language Models (MLLMs) such as GPT-4V (OpenAI, 2023), Gemini (Comanici et al., 2025), and Qwen-VL (Wang et al., 2024a). These models provide agents with comprehensive semantic understanding of both screenshots (vision) and complex instructions (text). Unlike previous generations, modern GUI agents can reason about user intent, interpret novel interfaces zero-shot, and plan long-horizon tasks. Recent milestones include high-fidelity benchmarks like OSWorld (Xie et al., 2024) and WebArena (Zhou et al., 2023), where agents now perform real-world tasks ranging from data entry to cross-application workflow management, though significant gaps in reliability and speed remain compared to human performance.

3.5 Frontier Models

In this subsection, we provide an overview of state-of-the-art GUI agent models trained with RL-based methods, organized roughly chronologically along four major directions: closed-source commercial systems, open-source general-purpose agents, grounding-specialized models, and reasoning-enhanced architectures.

Closed-source commercial systems. Over the past year, RL has progressively expanded the frontier of GUI agents. **OpenAI's Computer-Using Agent (CUA)** (OpenAI, 2025a;b), released in January 2025 alongside Operator, established the viability of RL-enhanced autonomous computer control, combining RLHF-style alignment with environment interaction RL on tasks with verifiable outcomes. **Anthropic's Claude Computer Use** (Anthropic, 2024), publicly available since late 2024 with continuous iterations, adopted a pure vision-based approach relying exclusively on screenshots, leveraging Constitutional AI (Bai et al., 2022) combined with RLHF for safe yet effective tool usage. These closed-source systems demonstrated that multimodal foundation models could be successfully adapted for GUI automation through RL training.

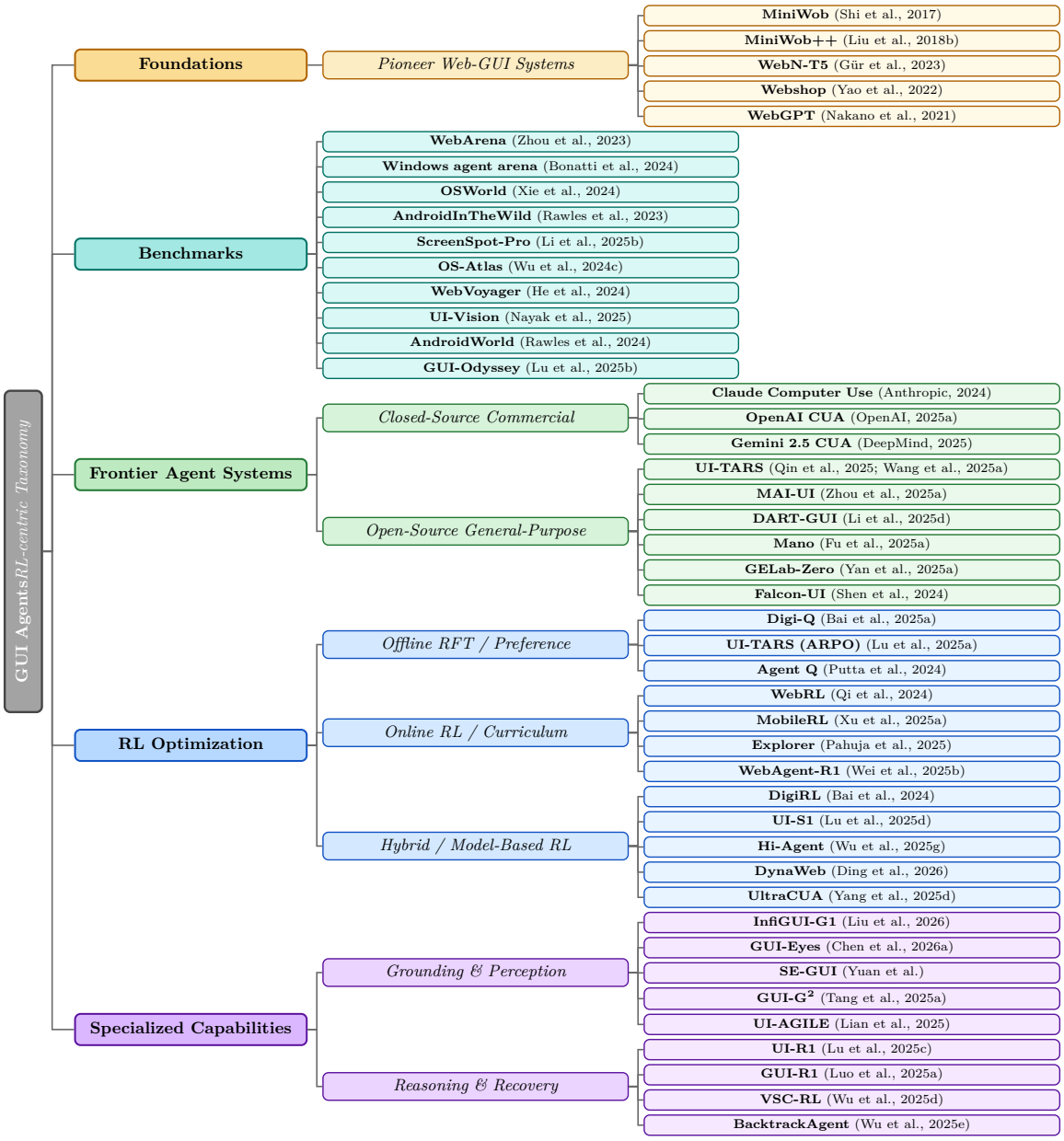

Figure 3: A taxonomy of representative GUI agent papers organised along five dimensions: **Foundations** (pioneer systems), **Environments & Benchmarks** (evaluation platforms), **Frontier Systems** (deployed agents), **RL Optimization** (training methods), and **Specialized Capabilities** (perception, reasoning, and recovery).

Open-source general-purpose agents. Open-source efforts rapidly closed the gap with proprietary systems. The **UI-TARS** series from ByteDance Seed (Qin et al., 2025; Wang et al., 2025a) represented a milestone, achieving 42.5% on OSWorld through a two-stage "SFT + RL" paradigm (detailed in Section 4.1). **MAI-UI** (Zhou et al., 2025a) from Alibaba Tongyi Lab targeted mobile-first deployment with Dynamic RL Scaling across 512 parallel environments. **DART-GUI** (Li et al., 2025d) addressed engineering challenges through decoupled multi-turn RL with adaptive data curation. **Mano** (Fu et al., 2025a) implemented a comprehensive three-stage hybrid pipeline (SFT $\rightarrow$ Offline RL $\rightarrow$ Online RL) using GRPO with composite rewards (Section 4.3). **GELab-Zero** (Yan et al., 2025a) from StepAI became the first GUI agent designed for edge deployment. Other notable contributions include **Falcon-UI** (Shen et al., 2024), which

Table 1: Comparison of representative open-source GUI agent models with RL-based training. **Modality**: T=Text, I=Image, V=Video. **Platform**: D=Desktop, M=Mobile, W=Web.

Model	Release	Params	RL Algorithm	Modality	Platform	Key Innovation
UI-TARS-1.5/2	2025-04/09	2B–72B	SFT+RL	T/I	D/M/W	Thoughts-before-actions
MAI-UI	2025-12	2B–235B	Dynamic RL Scaling	T/I/V	M	512 parallel envs, edge-cloud
DART-GUI	2025-09	7B	Decoupled RL	T/I	D/W	Adaptive data curation
Mano	2025-09	7B	GRPO (3-stage)	T/I	W	Closed-loop data cycle
GELab-Zero	2025-12	4B	Multi-turn RL	T/I	D/M	Edge deployment
InfiGUI-G1	2025-12	3B/7B	AEPO (RLVR)	T/I	D/M/W	Adaptive exploration reward
GUI-Eyes	2026-01	7B	GRPO	T/I	D/M/W	Active perception
SE-GUI	2025-05	7B	Self-evolutionary RL	T/I	D/M/W	Attention-based pseudo-labels
InfiGUI-R1	2025-04	3B	Actor2Reasoner RL	T/I	D/M	Error recovery rewards
UI-R1	2025-03	3B	Rule-based RL + DAST	T/I	D/M/W	Difficulty-adaptive thinking
GUI-R1	2025-04	7B	GRPO (RLVR)	T/I	D/M/W	Unified action space, data efficiency
UI-S1	2025-09	7B	Semi-online RL	T/I	M	Patch module, dual advantage
BacktrackAgent	2025-05	7B	Verifier-guided RL	T/I	D/M/W	Error detection & backtracking
VSC-RL	2025-02	7B	Variational subgoal RL	T/I	M/W	SGC-ELBO objective

pioneered "understanding GUI before following instructions" through large-scale unsupervised pretraining, and **DigiRL** (Bai et al., 2024), which established baseline methodologies for offline-to-online transition (Section 4.3.2). A growing ecosystem of complementary agents has further expanded the frontier: Agent S and Agent S2 (Agashe et al., 2024; 2025) introduced agentic and generalist-specialist frameworks; CogAgent (Hong et al., 2024) pioneered visual language models for GUI understanding; OS-Copilot (Wu et al., 2024b) targeted generalist computer agents; UFO2 (Zhang et al., 2025b) proposed a desktop AgentOS; and OpenCUA (Wang et al., 2025c) provided open foundations for computer-using agents. Mobile-focused efforts include SpiritSight Agent (Huang et al., 2025), Mobile-Agent-V3 (Ye et al., 2025), MagicGUI (Tang et al., 2025e), and AgentCPM-GUI (Zhang et al., 2025k), while ShowUI (Lin et al., 2025a) and InfiGUIAgent (Liu et al., 2025b) advanced vision-language-action modeling. Further contributions include UITron (Zeng et al., 2025), Ponder & Press (Wang et al., 2025e), CoAct-1 (Song et al., 2025c), OmegaUse (Zhang et al., 2026), ClickAgent (Hoscilowicz & Janicki, 2025), Step-GUI (Yan et al., 2025c;b), and GTA1 (Yang et al., 2025b) for GUI test-time scaling.

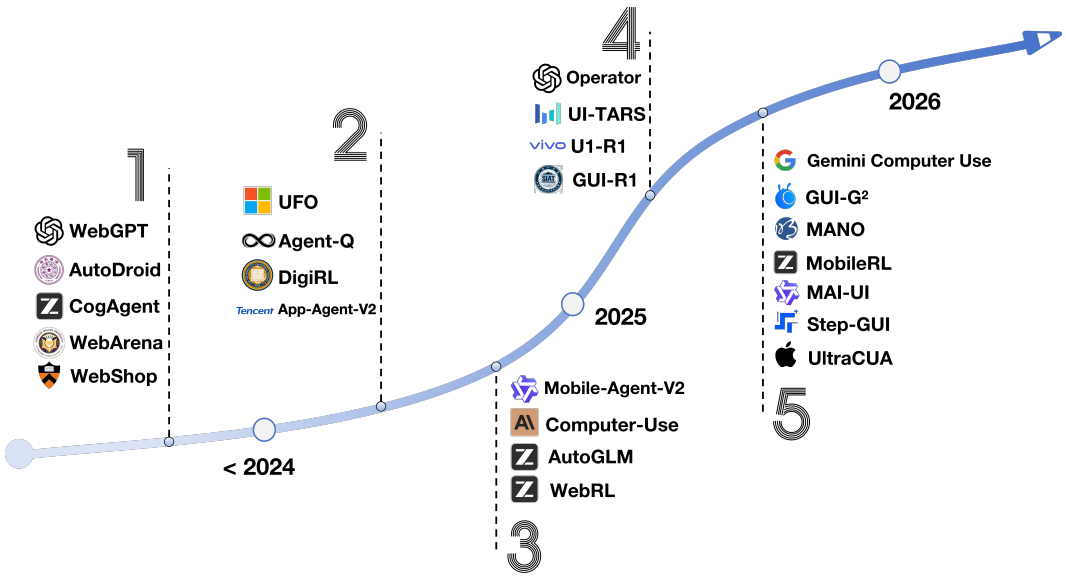

Figure 4: Timeline of GUI Agent Development.

Grounding-specialized models. Visual grounding—precise mapping from natural language to screen coordinates—has emerged as a specialized focus for RL optimization, building on foundational work in

universal visual grounding (Gou et al., 2024), unified pure vision agents (Xu et al., 2024d), Aria-UI (Yang et al., 2025c), and Phi-Ground (Zhang et al., 2025h) for perception, as well as visual test-time scaling for grounding (Luo et al., 2025c). **InfiGUI-G1** (Liu et al., 2026) introduced AEPO (Adaptive Exploration Policy Optimization) to address insufficient exploration in continuous coordinate spaces (Section 5.1.1). **GUI-Eyes** (Chen et al., 2026a) introduced active perception where models autonomously invoke visual tools before grounding (Section 5.3.2). **SE-GUI** (Yuan et al.) proposed self-evolutionary RL leveraging attention maps as intermediate supervision. **GUI-G^2** (Tang et al., 2025a) addressed the "1-pixel deviation equals failure" problem through Gaussian reward modeling (Section 5.1).

Reasoning-enhanced architectures. Reasoning capabilities have become increasingly central to GUI agent design. **InfiGUI-R1** (Liu et al., 2025c) explicitly targeted transforming agents from "Reactive Actors" to "Deliberative Reasoners" through the Actor2Reasoner framework (AAAI 2026 Oral). **UI-R1** (Lu et al., 2025c) adapted rule-based RL with DAST (Difficulty-Adaptive Slow-Thinking). **GUI-R1** (Luo et al., 2025a) demonstrated extreme data efficiency through GRPO with verifiable rewards (detailed in Section 4.1.3). Semi-online and hybrid approaches have further bridged offline learning and online interaction: **UI-S1** (Lu et al., 2025d) introduced semi-online RL with trajectory patching (Section 4.3.2); **VSC-RL** (Wu et al., 2025d) reconceptualized GUI control as subgoal-conditioned variational RL; and **BacktrackAgent** (Wu et al., 2025e) (EMNLP 2025) embraced error detection and recovery through a Generator-Verifier-Judger-Reflector architecture.

Foundation model backbones. Several multimodal foundation models serve as common backbones for GUI agent development. **Qwen3-VL** (Bai et al., 2025b) from Alibaba (2B–235B parameters) provides native Visual Agent capabilities with extended context (256K–1M tokens) and both Instruct and Thinking variants, serving as the backbone for MAI-UI, InfiGUI-G1, and GUI-Eyes. Other widely used backbones include Seed1.5-VL (Guo et al., 2025b), Kimi-VL (Team et al., 2025), and InternVL3.5 (Wang et al., 2025b). **Seed-1.8** (ByteDance, 2025) from ByteDance complements specialized GUI agents in a "general brain + specialized executor" architecture, with RL optimization on closed-loop business data. We provide a comprehensive timeline of GUI agent development in Figure 4 and detailed information on open-source models in Table 1.

4 RL Methods in GUI Agents

RL for GUI agents has branched into distinct methodological schools, each targeting a different bottleneck in the agent training lifecycle. We categorize the literature into three paradigms based on when and how the agent interacts with the environment. **Offline RL** focuses on learning from static datasets without environment interaction, enabling safe and scalable policy development. **Online RL** enables direct interaction with dynamic environments, optimizing policies through real-time trial and error. **Hybrid Strategies** bridge the gap between static pre-training and dynamic adaptation, including semi-online methods that simulate interaction dynamics on static data and staged training pipelines that combine offline initialization with online refinement. Across all paradigms, **Reinforcement Fine-Tuning (RFT)**—applying RL algorithms to fine-tune pretrained VLMs—serves as the dominant implementation approach, with offline methods typically employing DPO-based RFT and online methods employing PPO/GRPO-based RFT.

4.1 Offline Reinforcement Learning

While online interaction yields the richest learning signal, the latency, cost, and safety risks of live GUI exploration have driven adoption of **Offline Reinforcement Learning**. This paradigm distills optimal behaviors from static datasets—web interaction logs, human demonstrations, synthesized trajectories—so that agents internalize complex reasoning patterns without incurring the expense or irreversibility of real-time trial-and-error. In GUI environments, where a single environment step can take 0.5–2 s (network latency, rendering delays) and where erroneous actions may be irreversible (accidental deletions, unintended purchases), offline methods provide a critical pathway for safe, scalable agent development.

4.1.1 Theoretical Foundations

Core definition. Offline RL (also called Batch RL) learns a policy $\pi(a|s)$ entirely from a fixed dataset $\mathcal{D} = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^{|\mathcal{D}|}$ without any environment interaction during training. For GUI agents this constraint is not merely convenient but often necessary: real applications execute slowly, incur API costs, and risk irreversible side-effects.

Distribution shift and value overestimation. The fundamental challenge in offline RL is *distribution shift*. Standard Q-learning and its variants (Van Hasselt et al., 2016) update via $Q(s, a) \leftarrow r + \gamma \max_{a'} Q(s', a')$, which queries the value of actions that may never appear in the training dataset. When the learned policy selects an *out-of-distribution (OOD) action* $a \notin \text{supp}(\pi_\beta)$, neural function approximators produce erroneously optimistic Q-values; the max operator then amplifies these errors, yielding a policy that performs well on paper but poorly in deployment. If π_β denotes the behavior policy and π the learned policy, the distributional mismatch $d^\pi(s, a) \neq d^{\pi_\beta}(s, a)$ causes errors to compound as $\mathcal{O}(T \cdot \epsilon_{\text{OOD}})$ over a horizon of T steps.

Key technical approaches. To address OOD action evaluation, the GUI agent community has adopted several principled strategies. **Conservative Q-Learning (CQL)** (Kumar et al., 2020) adds a regularization term to the Q-function loss that explicitly penalizes Q-values for OOD actions while rewarding Q-values for actions observed in the dataset, learning a *lower bound* on the true Q-function that ensures policies remain conservative. **Implicit Q-Learning (IQL)** (Kostrikov et al., 2021) avoids querying OOD actions entirely by using *expectile regression* to estimate value functions within the support of the dataset, sidestepping the extrapolation problem altogether. **Decision Transformer (DT)** (Chen et al., 2021) reframes RL as a *sequence prediction problem*: given a sequence of states, actions, and a target return-to-go (RTG), the model autoregressively generates actions conditioned on achieving the specified return. By avoiding temporal-difference bootstrapping, DT circumvents value overestimation and naturally aligns with the Transformer architectures underlying modern VLMs.

4.1.2 Offline RFT Methods

Reinforcement Fine-Tuning (RFT) refers to applying RL algorithms to fine-tune a pretrained VLM that has already acquired basic instruction-following and GUI understanding capabilities through supervised fine-tuning (SFT). Unlike training RL from scratch, RFT leverages the strong prior knowledge embedded in pretrained models, dramatically accelerating convergence. The primary goals of RFT are to address hallucination problems that SFT cannot resolve and to improve credit assignment in multi-step reasoning tasks. Within the offline paradigm, two RFT approaches dominate.

Direct Preference Optimization (DPO). DPO (Rafailov et al., 2023) bypasses explicit reward modeling by exploiting the closed-form relationship between the optimal policy and the reward under a KL-constrained objective. Given preference pairs (τ^+, τ^-) where τ^+ is preferred, the loss is:

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(\tau^+,\tau^-)}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(\tau^+)}{\pi_{\text{ref}}(\tau^+)} - \beta \log \frac{\pi_\theta(\tau^-)}{\pi_{\text{ref}}(\tau^-)}\right)\right]$$

where π_{ref} is the reference policy (typically the SFT checkpoint) and β controls divergence. For GUI agents, preference pairs are constructed from successful versus failed trajectories in static datasets—sidestepping the instability of critic-network training that plagues PPO at VLM scale.

Offline GRPO with verifiable rewards. Group Relative Policy Optimization (GRPO) (Shao et al., 2024), popularized by DeepSeek-R1, replaces the learned critic with group-relative advantage estimation. For a prompt x, GRPO samples a group $\{o_1, \ldots, o_G\}$ from the current policy and computes:

$$\hat{A}_i = \frac{r(o_i) - \mu_{\mathbf{r}}}{\sigma_{\mathbf{r}}}, \quad \mu_{\mathbf{r}} = \frac{1}{G}\sum_{j=1}^{G} r(o_j), \quad \sigma_{\mathbf{r}} = \sqrt{\frac{1}{G}\sum_{j=1}^{G}(r(o_j) - \mu_{\mathbf{r}})^2}$$

The policy update maximizes $\sum_i \hat{A}_i \log \pi_\theta(o_i|x)$ subject to a KL penalty against the reference policy. For GUI agents, $r(\cdot)$ is a verifiable reward—coordinate-in-bounding-box checks, action-type matching, format compliance—computable from static trajectory data without any learned reward model. Eliminating the critic cuts GPU memory by roughly half, enabling RFT of 72B-parameter VLMs on commodity clusters.

4.1.3 Representative Methods

We categorize representative offline RL and RFT methods for GUI agents into three principal technical approaches: value-based methods that learn action-value functions from static data, preference-based optimization that leverages trajectory comparisons, and policy gradient methods with verifiable rewards.

Value-based offline RL. Value-based approaches train Q-functions to estimate long-term returns, enabling action selection through value maximization without requiring online interaction. **Digi-Q** (Bai et al., 2025a) exemplifies pure offline learning by first performing representation fine-tuning via SFT to ensure discriminative GUI state features, then training a lightweight MLP head on the *frozen* VLM backbone using IQL or CQL variants. At inference, the policy employs *Best-of-N re-ranking*: sampling N candidate actions and selecting the highest-valued one, achieving 21.2% improvement over prior offline methods on AndroidInTheWild (Rawles et al., 2023). This demonstrates that "inference-time compute" can effectively substitute for expensive online data collection. **DigiRL** (Bai et al., 2024) extends this paradigm through a two-stage offline-to-online framework; its offline stage uses the AndroidInTheWild dataset (715K trajectories) for initialization via filtered BC (Torabi et al., 2018) or AWR (Peng et al., 2019) variants, while the online stage and full hybrid pipeline are detailed in Section 4.3.2.

Preference-based optimization. DPO-based methods bypass explicit reward modeling by directly optimizing policies from trajectory preference pairs, offering training stability advantages for large VLMs. **UI-TARS** (Qin et al., 2025) targets cross-platform automation (Android, Windows, Web) through native end-to-end screenshot processing, constructing preference pairs from successful versus failed trajectories. To address the sparse reward problem where entire batches may fail, UI-TARS introduces *experience replay* (Schaul et al., 2015): maintaining a buffer of successful trajectories and sampling from it when current-batch rewards are uniformly zero, ensuring gradient validity. The ARPO (Lu et al., 2025a) variant combining DPO with GRPO reached 20.4% on OSWorld, substantially exceeding the 15.6% SFT baseline. **Agent Q** (Putta et al., 2024) advances preference optimization by integrating *Monte Carlo Tree Search (MCTS)* (Coulom, 2006) for high-quality data synthesis. Guided MCTS simulates future web states using a value model for pruning, while a *self-critique* mechanism enables AI-driven state evaluation during search. MCTS-discovered successful paths become DPO positive examples, with failed paths as negatives. This "search is data" philosophy improved Llama-3-70B's (Dubey et al., 2024) zero-shot success on real-world web booking (e.g., OpenTable) from 18.6% to 81.7%—a 340% relative gain—demonstrating that search can uncover complex trajectories inaccessible through random exploration.

Policy gradient with verifiable rewards. GRPO-based methods, inspired by DeepSeek-R1's success in mathematical reasoning, optimize policies through group-relative advantage estimation with rule-based verifiable rewards. **GUI-R1** (Luo et al., 2025a) and **UI-R1** (Lu et al., 2025c) employ a *unified action space* encoding clicks, swipes, and keyboard inputs, combined with meticulously designed binary rewards: for grounding tasks, +1 if the predicted coordinate falls within the target bounding box, 0 otherwise; for multi-step tasks, sparse terminal rewards upon goal achievement (URL change, element match). The critical insight is extreme data efficiency: GUI-R1 achieved state-of-the-art on ScreenSpot-Pro (Li et al., 2025b) and seven other benchmarks using only 3K samples (GUI-R1-3K)—merely 0.02% of OS-Atlas's (Wu et al., 2024c) 13M training examples—further analyzed from a data efficiency perspective in Section 5.2. Analysis revealed emergent reasoning patterns: models spontaneously generated internal monologues ("first observe overall layout, then locate specific elements"), suggesting that rule-guided RLVR can induce System-2-style deliberation without explicit reasoning supervision.

4.1.4 Emerging Directions

Visual-language alignment stability. A critical challenge in applying RL to VLMs is *visual forgetting*: aggressive parameter updates to optimize specific button-clicking behaviors may degrade the model's general visual recognition capabilities. Digi-Q's frozen-backbone strategy elegantly sidesteps this issue, but online RL methods require more sophisticated solutions such as KL-divergence constraints or Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) to preserve visual grounding while optimizing action policies.

Toward System-2 GUI agents. The success of reasoning-enhanced models like DeepSeek-R1 points toward GUI agents that are not merely reactive executors but deliberative reasoners. Reinforcing explicit reasoning processes through RL—where agents receive rewards for both correct actions *and* valid reasoning chains—represents a promising frontier, with process reinforcement through implicit rewards (Cui et al., 2025) and AgentPRM (Xi et al., 2025) offering principled approaches to step-level credit assignment. This direction is further discussed alongside the System 1/System 2 cognitive hybridization paradigm in Section 4.3.3 and Section 7.

Synthesis & Insight: The Safety Barrier. While Offline RL is often praised for its computational and data efficiency, its most critical role in GUI automation is acting as a **safety barrier**. Unconstrained online exploration by an untrained agent in a real operating system risks catastrophic and irreversible actions—deleting user data, sending unintended emails, or executing unauthorized financial transactions. By distilling the foundational semantics of UI interactions from static datasets, Offline RL confines the trial-and-error process to a safe proxy, ensuring "common sense" is acquired before the agent ever touches a live environment.

4.2 Online Reinforcement Learning

Online RL represents the most direct paradigm for GUI agent development, treating the GUI not as a static dataset but as a dynamic environment where agents refine policies through continuous trial. In contrast to offline RL, online RL enables agents to continuously interact with real environments, collect data in a streaming fashion, and update policies in real-time. This paradigm is particularly critical for GUI agents due to several domain-specific characteristics: *environment volatility*, where software updates and UI redesigns introduce persistent distribution shifts that render static training obsolete; *long-horizon sequential decision-making* with sparse terminal rewards, necessitating iterative trial-and-error to shape effective policies; *generalization demands* across heterogeneous applications and platforms; and *annotation bottlenecks*, where manual labeling of correct action sequences is expensive and fails to cover long-tail scenarios.

4.2.1 Theoretical Foundations

From imitation learning to online RL. Early GUI agents relied on zero-shot prompting or Behavioral Cloning (BC) (Florence et al., 2022), which treats action prediction as supervised classification. BC suffers from *covariate shift*: a single erroneous prediction at step t pushes the agent into states absent from the training distribution, and without recovery experience these errors compound quadratically—motivating the shift to online RL where the agent can learn to recover from its own mistakes.

POMDP formulation. Online RL for GUI agents is formalized as a Partially Observable Markov Decision Process (POMDP). Unlike offline methods, online agents actively interact with environments, enabling exploration and correction to learn recovery from erroneous states, as well as non-stationarity adaptation to handle dynamic GUI changes (e.g., loading speeds, popups).

State and action space heterogeneity. GUI states are highly heterogeneous, combining visual (pixels) and structural (DOM) modalities. The action space is similarly mixed, featuring both discrete types (Click, Type) and continuous parameters. Since traditional algorithms like DQN struggle with hybrid spaces, mainstream approaches favor policy gradient methods (e.g., PPO, GRPO) with specialized action decoding.

Sparse rewards and reward engineering. Acquiring reward signals is a major bottleneck. Task success (e.g., a multi-step purchase) is extremely sparse, delayed, and difficult to verify automatically. Consequently, designing dense reward functions or using Model-as-a-Judge evaluators (see Section 5.1) has become essential.

Sample efficiency and interaction cost. Online RL is bounded by the slow speed of real-world environment interactions (e.g., rendering and network latency). Training requires extensive sample collection, making techniques like curriculum learning, offline pre-training, and semi-online methods critical for improving sample efficiency.

4.2.2 Representative Methods

Curriculum-based online learning. Curriculum learning addresses the cold-start problem where random exploration yields insufficient positive rewards. WebRL (Qi et al., 2024) introduces a *Self-Evolving Online Curriculum* comprising task generation using teacher models and a *Failure Set Strategy* that collects unsuccessful tasks and generates simplified variants, ensuring training tasks remain within the agent's "Zone of Proximal Development." Related approaches include Curriculum-RLAIF (Li et al., 2025c), which combines curriculum strategies with AI feedback, and RLAIF (Lee et al., 2023), which demonstrated the viability of AI-generated feedback as an alternative to human feedback. WebRL additionally trains an *Outcome-Supervised Reward Model (ORM)* (Yu et al., 2024) that judges trajectory success from final states, providing stronger generalization than rule-based checkers. KL-divergence constraints and experience replay filtering prevent policy drift while preserving general capabilities.

Difficulty-adaptive policy optimization. Mobile GUI environments exhibit heavy-tailed task difficulty distributions where standard RL algorithms are dominated by simple task gradients. MobileRL (Xu et al., 2025a) addresses this through *AdaGRPO*, introducing difficulty weighting based on historical success rates—lower success rates yield higher gradient weights. *Shortest-Path Reward Adjustment (SPA)* suppresses reward hacking by penalizing redundant operations, while *Failure Curriculum Filtering (FCF)* temporarily removes tasks with persistent zero success rates.

Offline-to-online transition frameworks. Tabula rasa online RL is impractical due to low exploration efficiency and potentially dangerous operations. DigiRL (Bai et al., 2024) proposes a canonical two-stage paradigm combining offline initialization with online fine-tuning, pioneering VLMs as automatic evaluators for task completion judgment. We discuss DigiRL's full hybrid pipeline—including the Digi-Q algorithm and Best-of-N inference—in Section 4.3.2.

End-to-end multi-turn optimization. WebAgent-R1 (Wei et al., 2025b) proposes *Multi-Turn GRPO (M-GRPO)*, treating entire interaction trajectories as optimization samples rather than single-step actions, enabling agents to learn "delayed gratification." Recent work has extended multi-turn RL optimization further: Sweet-RL (Zhou et al., 2025c) introduced reward strategies specifically designed for multi-turn LLM agents, while RLTHF (Xu et al., 2025b) proposed targeted human feedback mechanisms for fine-grained turn-level credit assignment. **HGPO** (He et al., 2026) further sharpens this line by addressing historical-context inconsistency in stepwise GRPO/GiGPO-style updates: when prompts depend on long interaction histories, optimizing each step against a stale or partially reconstructed context can assign credit to actions under a different state than the one seen at execution time. This issue is especially acute for GUI agents whose prompts interleave screenshots, summaries, tool traces, and memory snippets. *Dynamic Context Compression* addresses context window explosion by having agents output observation summaries at each step, while *Parallel Trajectory Rollout* improves training diversity.

Grounding-specialized methods. InfiGUI-G1 (Liu et al., 2026) discovers that for concrete grounding tasks, forcing Chain-of-Thought (Wei et al., 2022) reasoning actually *decreases* precision due to hallucinations. *Fast Thinking Templates* suppress reasoning and directly regress coordinates, with "System 1" mode outperforming "System 2" for grounding. UI-AGILE (Lian et al., 2025) employs continuous distance-based rewards $R = \max(0, 1 - \text{distance}/\text{threshold})$ providing denser gradients than binary hit/miss rewards, combined with *Cropping-Based Resampling* for small element recognition.

Exploration-driven data synthesis. Explorer (Pahuja et al., 2025) addresses cold-start through a multi-agent pipeline where "explorers" randomly walk through environments discovering novel states, and "annotators" reverse-engineer natural language instructions reaching these states, synthesizing high-quality trajectories for subsequent online training.

Infrastructure-aware online RL. Beyond algorithmic exploration, recent systems emphasize that GUI RL quality depends on rollout infrastructure and signal hygiene. **AgentCPM-Explore** (Chen et al., 2026b) is representative: it studies RL under noisy real I/O, combines reward-signal denoising with context compression, and treats trajectory collection as a systems problem rather than a purely policy-optimization problem. This makes it a useful concrete anchor for the I/O-wall argument developed later in Section 5.2.

4.2.3 Emerging Directions

Curriculum learning as the dominant paradigm. From WebRL's "self-evolving failure sets" to MobileRL's "failure curriculum filtering," all high-performance frameworks abandon random sampling in favor of curriculum learning variants. This reflects the enormous heterogeneity in GUI task spaces—agents must actively select training data appropriate to current capabilities for efficient learning. Future agents will function not merely as learners but as "self-educators" capable of designing their own practice problems.

Separation of reasoning and execution. WebRL and MobileRL emphasize reasoning for planning, while InfiGUI-G1 demonstrates that reasoning interferes with precise grounding. This suggests future architectures will adopt dual-system designs—a theme we elaborate in Section 4.3.3 and Section 7.

Model-as-a-judge normalization. As environment complexity increases, writing rule-based reward functions becomes impractical. DigiRL's VLM evaluator and WebRL's ORM mark the arrival of the AI-evaluating-AI era. Reward engineering is transforming into reward modeling (Section 5.1.2). While this solves sparse reward problems, it introduces new risks—*reward hacking*—and ensuring evaluation model robustness represents the next research frontier.

Synthesis & Insight: The I/O Wall. The fundamental bottleneck in scaling Online RL for GUI agents lies not in algorithmic maturity, but in **The I/O Wall**. Unlike the microsecond transitions in simulated games like Atari or Go, real-world GUI interactions suffer from severe latency—network requests, page rendering, DOM parsing, and UI animations easily push single-step processing times to 0.5–2.0 seconds. This staggering environment feedback latency structurally caps sample collection throughput. Consequently, the core contradiction in Online RL is resolving this I/O gap, shifting the focus from purely algorithmic improvements to system-level innovations like heavily optimized simulators, parallelized cloud browser rendering, or asynchronous multi-agent rollouts.

4.3 Hybrid Strategies

Pure offline RL suffers from distribution shift—where agents fail to recover from states unseen during training—while pure online RL is sample-inefficient and carries substantial risks in real operating system environments. Hybrid strategies attempt to bridge this gap through complementary approaches that combine the safety of offline methods with the adaptability of online exploration. These approaches have emerged as the dominant paradigm for training state-of-the-art GUI agents, achieving performance levels that neither pure offline nor pure online methods can match independently.

4.3.1 Theoretical Foundations

Core motivation. Hybrid strategies resolve the tension between offline and online RL: offline methods provide safe policy initialization but suffer from distribution shift, while online methods enable adaptive exploration but are sample-inefficient and risky. Hybrid approaches leverage complementary advantages by using offline data for warm starts, online/semi-online rollouts for distribution correction, hierarchical architectures for long-term credit assignment, and world models for low-cost latent exploration.

Hybrid optimization objective. Hybrid RL extends the traditional discounted return $J(\pi)$ with auxiliary losses:

$$L_{\text{hybrid}} = \lambda_1 L_{\text{RL}} + \lambda_2 L_{\text{BC}} + \lambda_3 L_{\text{Reasoning}}$$

Here, L_{RL} optimizes long-term returns (e.g., PPO/GRPO), L_{BC} prevents catastrophic forgetting early in training via behavioral cloning, and $L_{\text{Reasoning}}$ enforces logical planning via chain-of-thought generation. Loss weights typically shift from L_{BC} to L_{RL} as training progresses.

Hybrid action space formulation. Advanced agents integrate distinct action modalities: *atomic actions* ($\mathcal{A}_{\text{low}}$), which are universal but inefficient pixel-level primitives (e.g., `click(x,y)`), and *semantic/tool actions* ($\mathcal{A}_{\text{high}}$), which are efficient but environment-dependent API operations (e.g., `checkout()`) (Song et al., 2025e). Agents dynamically route between these action spaces to balance visual robustness with execution efficiency.

Key technical approaches. The community has developed several core strategies. *Semi-online learning* simulates online dynamics (e.g., trajectory patching) on offline datasets, allowing models to learn error recovery without live interaction costs. *Staged training pipelines* sequentially transition from offline initialization to targeted online exploration, minimizing catastrophic failure risks in real environments. *Hierarchical RL (HRL)* decomposes tasks into a Planner (generating semantic subgoals) and an Executor (performing UI actions), isolating complexity and mitigating sparse rewards. *Model-Based RL (MBRL)* employs UI world models to dream realistic state transitions (Gu et al., 2024), massively accelerating exploration in latent space before grounding with limited online samples.

4.3.2 Representative Methods

We highlight six principal technical approaches within hybrid RL methods, each addressing specific bottlenecks in GUI agent training. In semi-online reinforcement learning, **UI-S1** (Lu et al., 2025d) simulates online dynamics on static data using a *Patch Module* to correct out-of-distribution actions and a dual-level advantage function, maintaining offline throughput while mimicking online error-correction. For offline-to-online transition, **DigiRL** (Bai et al., 2024) employs a two-stage pipeline—offline initialization followed by targeted online fine-tuning—using *Digi-Q* with a frozen VLM backbone and *Best-of-N* sampling to reach a 67.2% success rate on AitW. To handle long-horizon tasks, **Hi-Agent** (Wu et al., 2025g) introduces hierarchical planning and execution, jointly training a semantic *Planner* and a UI *Executor* via GRPO and a *Foresight Advantage Function*, unlocking 87.9% on AitW. Related hierarchical approaches include HiPER (Peng et al., 2026), which addresses credit assignment through hierarchical RL with structured reward decomposition, probabilistic subgoal representations for HRL (Wang et al., 2024e), and **MiRA** (Wang et al., 2026), which operationalizes milestone-based planning and potential shaping for web navigation. MiRA is particularly relevant because it turns the otherwise abstract idea of subgoal-driven GUI RL into concrete intermediate objectives, grounding multi-tier reward design in measurable navigation progress. Addressing latency and safety, **DynaWeb** (Ding et al., 2026) leverages world model-augmented learning by efficiently "dreaming" trajectory rollouts within a *Web World Model (WWM)*. In the domain of hybrid action spaces, **Ultra-CUA** (Yang et al., 2025d) unifies visual primitives and programmatic API tools, training the agent to flexibly route between universal visual fallbacks and fast deterministic executions. Finally, **UI-AGILE** (Lian et al., 2025) demonstrates training-inference dual enhancement by combining dense IoU-based grounding rewards with grid-based partitioned reasoning at inference, improving ScreenSpot accuracy by 23%.

4.3.3 Emerging Directions

Unified cross-ecosystem agents. Current agents typically specialize in single platforms. However, real user workflows are cross-ecosystem (capturing images on mobile, editing on desktop, emailing via web). Future hybrid architectures must enable automatic alignment of interaction logic across heterogeneous operating systems through RL, achieving "train once, deploy everywhere" generalization (see Section 7).

Continual learning and sim-to-real transfer. World model-based approaches like DynaWeb face simulation-reality gaps—real web environments contain CAPTCHAs, dynamic advertisements, and network

failures that are difficult for world models to perfectly predict. Future hybrid strategies must incorporate domain randomization during dreaming and online adaptation during deployment, enabling agents to leverage test-time feedback for continuous policy refinement without catastrophic forgetting.

Privacy-aware hybrid learning. As agents access sensitive data, future hybrid strategies must integrate *Privacy Critics* that impose penalties for high-risk operations, implementing Safe RL principles within the optimization framework. Constrained RL formulations (Zhang et al., 2024a) and worst-case optimization (Yang et al., 2021) provide theoretical foundations for enforcing safety constraints during policy optimization (see also Section 7).

Efficient edge deployment. Hybrid strategies involving world models and chain-of-thought reasoning substantially increase inference computation. Deploying 7B–30B parameter models on mobile devices faces energy and latency challenges. Future research directions include distillation and quantization: using powerful hybrid RL agents as teachers to guide lightweight student models (1B–3B parameters) that bypass heavy reasoning processes and directly learn optimal action mappings for efficient on-device execution.

Synthesis & Insight: Cognitive Stratification. Ultimately, Hybrid strategies embody a profound structural shift toward **Cognitive Stratification**. Rather than viewing offline and online phases as mere technical prerequisites for training efficiency, they serve distinct cognitive purposes in the agent's evolution. The offline phase initializes the agent's "System 1"—the fast, intuitive "common sense" required to robustly perceive components, interpret icons, and execute safe atomic actions. Built upon this foundation, the online phase acts as the crucible for "System 2"—honing the agent's intuition, long-horizon planning, error recovery, and complex reasoning in dynamic novel environments. This layered evolution gracefully resolves the tension between execution efficiency and goal robustness.

Cognitive hybridization: System 1 and System 2. A recurring theme across all paradigms (Sections 4.1.4 and 4.2.3): inspired by dual-process theory, future GUI agents will likely integrate fast, intuitive "System 1" modules for routine operations with slow, deliberative "System 2" modules for complex reasoning. RL can optimize the routing between these cognitive modes—learning when quick reactions suffice versus when deep thinking is required. Novel policy optimization approaches such as group-in-group optimization (Feng et al., 2025), HGPO (He et al., 2026), and multi-agent RL with state modelling (Kontogiannis et al., 2025) offer complementary perspectives on structuring agent interactions and optimization groups. **ERL** (Shi et al., 2026) adds a complementary mechanism: structured reflection can transform sparse terminal feedback into learnable intermediate signals and consolidate successful revisions across attempts. Evidence from GUI-R1's emergent reasoning (Section 4.1.3) and InfiGUI-G1's finding that explicit CoT *hurts* grounding (Section 5.3.2) suggests that cognitive hybridization represents a promising frontier.

5 Key Dimensions

Building upon the method-centric overview of RL paradigms presented in Section 4, this section adopts a *dimension-centric* perspective. Specifically, it analyzes three critical, cross-cutting dimensions that govern the design of RL-based GUI agents across all paradigms: *reward engineering*, *data efficiency*, and *technical innovations* (encompassing algorithmic, perceptual, and memory-related advances). Each dimension addresses foundational challenges in GUI automation that parallel, yet remain distinct from, the difficulties encountered in traditional RL domains. To highlight overarching design principles and facilitate cross-method comparisons, the following discussion draws upon concrete instantiations introduced in Section 4, avoiding redundant descriptions of individual systems.

5.1 Reward Engineering

Unlike classical RL environments that provide explicit rewards, GUI automation requires interpreting complex visual and semantic evidence to assess task completion (Nguyen et al., 2025). Formally, the ideal GUI

reward combines a terminal indicator and a dense shaping function $\phi(s_t, a_t, g)$:

$$\mathcal{R}^*(s_t, a_t) = \underbrace{\mathbb{1}[\text{task completed at } t]}_{\text{terminal}} + \underbrace{\lambda \cdot \phi(s_t, a_t, g)}_{\text{dense shaping}}$$

Designing ϕ is notoriously difficult: sparse signals hinder learning, while overly dense ones provoke reward hacking. To navigate this accuracy–generality trade-off, current literature converges on a three-tier taxonomy: *rule-based* rewards exploiting UI structures, *LLM-as-judge* rewards evaluating via foundation models, and *learned* rewards parameterized and optimized alongside the policy.

The shift toward verifiable environment feedback. A critical overarching insight is that reward engineering is fundamentally shifting from "manually defined formatting heuristics" to **environment-feedback verification**. Because GUI environments inherently afford objective state validations—such as exact URL transitions, deterministic DOM alterations, and observable database changes—they naturally support **Reinforcement Learning with Verifiable Rewards (RLVR)** as the definitive future trend. State-of-the-art systems increasingly anchor their optimization on these unforgeable environmental realities. **Information-aware credit assignment (ICA)** (Pang et al., 2026) strengthens this argument by explicitly tying credit to informative observations rather than treating every historical token or UI state as equally relevant. For GUI and web agents, this reinforces the case for visual-first observations: screenshots, highlighted regions, and verifiable state changes often carry denser credit information than brittle HTML parser traces alone.

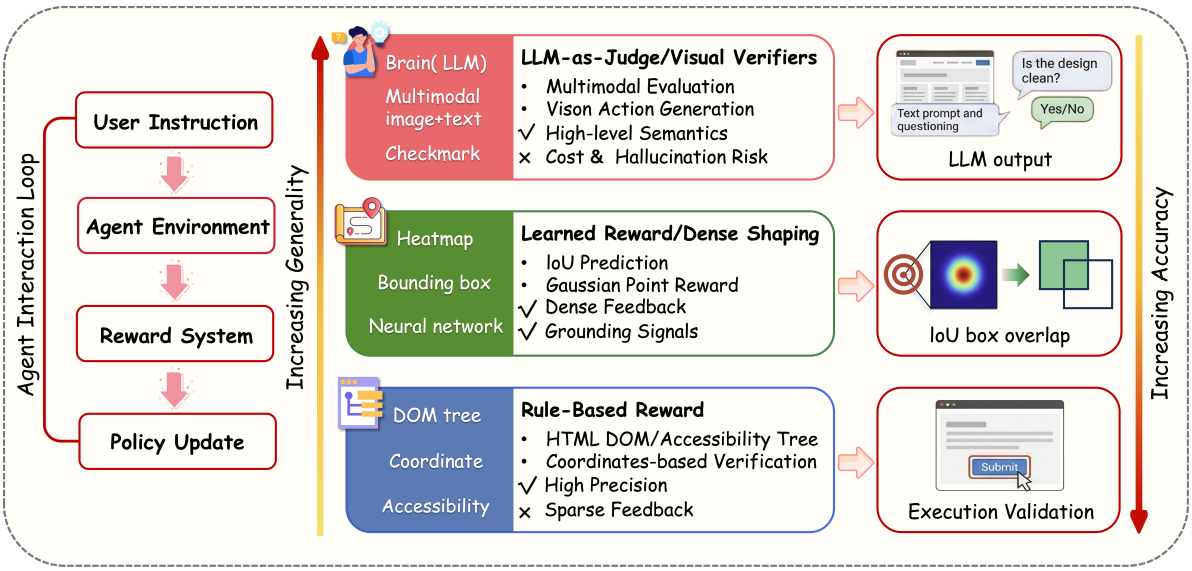

Figure 5: The Reward Engineering Pyramid balances accuracy and generality for GUI Agents: rule-based rewards (base) offer precision; learned rewards (middle) provide dense signals; LLM-as-Judge (apex) enables broad semantic task handling with hallucination risks.

5.1.1 Rule-Based Rewards

Rule-based rewards offer interpretable and computationally cheap signals by leveraging structured OS metadata (e.g., DOM trees, bounding boxes) to build explicit scoring functions without learned models. A pivotal question arises: *why do seemingly rigid, rule-based systems remain the core optimization engine for SOTA models?* The profound answer lies in their provision of **uncheat-able feedback**. In an era where LLM policies persistently exploit the semantic loopholes or subjective evaluations of AI judges (Reward Hacking), rule-based verifiable rewards provide absolute ground truths. They anchor the feedback loop in objective reality, forcing the model to achieve genuine execution correctness rather than just generating plausible-looking actions. Their design space spans from binary outcomes to dense continuous shaping.

From binary outcomes to continuous shaping. Binary rewards (success = +1, failure = 0) present clean targets but suffer from credit-assignment issues. **UI-R1** (Lu et al., 2025c) mitigates this by decomposing rewards into verifiable action and format checks. Such *verifiable* rewards—where ground truth is algorithmically determined—enable GRPO-style optimization without explicit reward models, allowing agents to consistently outperform heavily supervised baselines. Similarly, **BTL-UI** (Zhang et al., 2025i) employs a composite verifiable reward ($R = R_{\text{format}} + R_{\text{blink}} + R_{\text{link}}$) simulating a human "Blink-Think-Link" process. By integrating format compliance, region IoU, and action matching, this richer decomposition significantly boosts task success rates.

A fundamental limitation of binary rewards is that a prediction one pixel outside the bounding box is penalized identically to one that is entirely off-screen, producing vanishing gradients for all but the most accurate samples. Two approaches address this by introducing continuous spatial shaping. **GUI-G^2** (Tang et al., 2025a;b) replaces the binary indicator with a *Gaussian point reward* centered on the element centroid, with variance proportional to the bounding-box area, and adds a complementary *coverage reward* measuring distribution overlap via the Bhattacharyya coefficient; the resulting dense objective substantially improves grounding accuracy. **LPO** (Tang et al., 2025d) (Location Preference Optimization) offers an alternative continuous formulation based on window information entropy and Euclidean distance ($R = R_w \times R_d$), enhancing spatial localization and precision on benchmarks like **Multimodal Mind2Web** (Deng et al., 2023).

Combating exploration collapse. Even with dense rewards, standard single-sample RLVR can fall into a "confidence trap": when the policy is already confident in an incorrect action it never generates the correct one and therefore never receives a positive signal. **InfiGUI-G1** (Liu et al., 2026) breaks this deadlock with Adaptive Exploration Policy Optimization (AEPO), which generates multiple candidate answers per forward pass and scores them via an efficiency-derived reward $\eta = U/C$ (accuracy over candidate count). A collinearity penalty further encourages spatial diversity among candidates, creating learning signals for otherwise permanently "unlearnable" samples and improving overall semantic alignment.

5.1.2 LLM-as-Judge Rewards

For tasks too ambiguous for closed-form rules, foundation models can serve as generalized reward functions. Recent progress focuses on two intertwined threads: *improving judge accuracy* and *mitigating reward hacking*.

From passive inspection to proactive verification. Static LLM judges evaluate fixed trajectory logs passively, often struggling with borderline cases. **ProRe** (Dai et al., 2025) addresses this via a reasoner–actor architecture where models decompose evaluation into state-probing tasks executed in the live environment. By gathering active evidence, it significantly improves reward precision and downstream success rates. Similarly, **SmartSnap** (Cai et al., 2025) embeds verification directly into the agent's objective: agents are trained to both complete tasks and capture curated visual evidence, allowing lightweight judges to evaluate specific snapshots rather than full, noisy trajectories.

Reducing false positives and reward hacking. Enhancing the reliability of judge signals is critical. **ZeroGUI** (Yang et al., 2025a) employs a multi-query unanimous-agreement voting mechanism on trajectory screenshots to drastically reduce false-positive rates and self-hallucinations. To address reward hacking at its root, **WebRL** (Qi et al., 2024) trains its reward model on on-policy trajectories. This couples reward updates to the policy's evolving distribution, mitigating the mismatch exploited by adversarial actions. Nonetheless, maintaining judge robustness under sustained policy optimization remains an open challenge.

5.1.3 Learned Rewards

Learned reward functions occupy a middle ground: more flexible than hand-crafted rules, more sample-efficient than LLM judges. In the GUI domain they have been most impactful for spatial grounding, where the geometry of the interface provides a natural inductive bias. GUI-G^2's Gaussian framework (Section 5.1.1) doubles as a learned reward once its adaptive variance $\sigma \propto$ element size is treated as a geometry-conditioned function rather than a fixed hyperparameter (Tang et al., 2025a): small elements (e.g., close-button icons)

receive a tight reward landscape while large elements (e.g., banner images) receive a broad one—a distinction critical in high-resolution interfaces where targets can span fewer than 10×10 pixels. InfiGUI-G1's Adaptive Exploration Reward (Liu et al., 2026) goes further by making the reward a function of the *full candidate set* rather than a single prediction; the group-relative structure that distinguishes AEPO from naïve best-of-N reranking connects it directly to the GRPO family of algorithms.

Adjacent VLM reward-modeling work also matters even when it originates outside GUI automation. **MARVL** (Zhou et al., 2026), though robotics-focused, highlights issues that transfer directly to screen agents: learned visual rewards can mis-ground spatial relations, overfit to superficial visual cues, or be exploited by policies that optimize the evaluator rather than the task. Its remedies—stronger spatial grounding, adversarial reward validation, and tighter coupling between perception and action evidence—suggest how GUI reward models can move beyond screenshot-level plausibility toward robust process evaluation.

5.2 Data Efficiency

Online RL in live GUI environments is computationally expensive. Page rendering and network operations severely limit the throughput of environment interactions, making standard, data-hungry RL algorithms slow. Consequently, maximizing *data efficiency*—the policy improvement per environment interaction—is a central objective. Three complementary strategies have emerged to address this bottleneck: synthetic data generation via world models (increasing effective data volume at lower cost), enhancement of existing human demonstrations (improving signal quality per sample), and iterative self-improvement loops that recycle the agent's own experience. A complementary systems-level direction is automated training design: **AutoRL** (Afshar et al., 2022) suggests that hyperparameters, curricula, and even architecture choices can themselves become optimization targets, which is especially valuable when each GUI rollout is expensive.

5.2.1 Synthetic Data via World Models

The core idea is to replace or substantially supplement expensive real-environment rollouts with trajectories generated by surrogate models. Two complementary approaches have emerged. At the *reasoning* level, **DreamGym** (Chen et al., 2025a;b) distills environment dynamics into an abstract textual state space, using chain-of-thought reasoning over retrieved real trajectories to simulate experiences. Combined with an adaptive task curriculum, it enables previously infeasible online RL on complex web benchmarks. At the *action* level, **UI-Simulator** (Wang et al., 2025d) employs the LLM itself as a world simulator to predict visual or textual outcomes directly without rendering interfaces, achieving comparable performance to larger models with significantly less data. These strategies demonstrate that surrogate trajectories can efficiently match the learning value of real rollouts. Complementary methods include **SimURA** (Deng et al., 2025), **WebSynthesis** (Gao et al., 2025), **WebWorld** (Xiao et al., 2026), and **Code2World** (Zheng et al., 2026).

5.2.2 Enhancement of Human Demonstrations

Raw human demonstrations are often noisy and incomplete. Enriching them via structured post-processing or tapping alternative data sources can substantially improve the signal-to-noise ratio.

Structure-based approaches refine existing traces log. **GUI-ReWalk** (Lin et al., 2025b;c) converts undirected exploration into targeted RL training data through backward annotation, capturing complex cross-application workflows. Conversely, **Prune4Web** (Zhang et al., 2025f) addresses the DOM-tree size bottleneck by auto-generating Python scripts to prune irrelevant elements, improving grounding accuracy.

An orthogonal direction exploits *novel data sources*. **Watch-and-Learn** (Song et al., 2025b; Mischel, 2019; Song et al., 2025a) trains a dynamics model to predict actions from software usage videos on YouTube, demonstrating that instructional videos can serve as scalable alternatives to step-by-step demonstrations. These results confirm that the *structure* and *diversity* of demonstrations are as critical as their volume. Additional synthesis pipelines include **AgentTrek** (Xu et al., 2024c) and **OS-Genesis** (Sun et al., 2025).

5.2.3 Iterative Self-Improvement

Rather than relying on fixed datasets, iterative self-improvement allows agents to interact with the environment, collect fresh experience, and update their own training data. Methods vary in their feedback structures. **Co-EPG** (Zhao et al., 2025a;b) features a dual-model (Planner–Grounder) architecture refined via dynamic rewards: the planner learns executable strategies, while the grounder masters low-level intent fulfillment, enabling rapid success on Mind2Web with minimal annotations. Alternatively, Zhang et al. (2025k) iteratively use Implicit World Modeling and Self-Reflection as supervision signals to substantially boost performance across diverse tasks.

A profound by-product of self-improvement is *emergent reasoning*. As discussed in Section 4.1.3, **GUI-R1** (Luo et al., 2025a;b) spontaneously develops internal, System-2-styled monologues when trained via GRPO on limited samples without any explicit reasoning supervision. The implication is significant: when action spaces and rewards are sufficiently structured, complex deliberation can emerge natively, reducing the need for intensive reasoning annotations.

5.3 Technical Innovations

Beyond reward design and data strategies, a cluster of recent papers introduces algorithmic, perceptual, and memory innovations that address GUI-specific bottlenecks. We organize the discussion by the sub-problem each innovation targets.

5.3.1 Algorithmic Advances: Exploration and Multi-Turn Optimization

Efficient exploration in GUI agents is complicated by the hybrid action space and a sparse reward landscape. Two strategies have emerged: *curriculum-based credit assignment* and *structured exploration architectures*.

On the credit-assignment side, **WebRL** (Qi et al., 2024)'s self-evolving curriculum generates tasks from the agent's own failure set and simplifies them until they fall within its "Zone of Proximal Development." At a finer granularity, the reward-shaping innovations of **GUI-G^2** (Tang et al., 2025a) and **InfiGUI-G1** (Liu et al., 2026) demonstrate that the *shape* of the reward landscape accelerates convergence by providing non-zero gradients. **Agentic Entropy-Balanced Policy Optimization (AEBPO)** (Dong et al., 2025) targets rollout-entropy collapse through dynamic entropy-balanced rollout and optimization, significantly improving data efficiency during training.

On the structural side, **Nested Browser-Use Learning** (Li et al., 2025a) separates web agent reasoning into an outer loop for tool-integration and an inner loop for in-page goal-driven exploration. This hierarchical decomposition proves remarkably data-efficient without requiring massive sets of synthetic trajectories, highlighting that structuring the exploration process is as important as scaling data.

5.3.2 Multimodal Perception: Active and Adaptive Visual Grounding

A GUI agent that understands *what* to do may still fail if it cannot *see* the correct pixel. Recent innovations attack this perceptual bottleneck through *active perception* and *attention alignment*.

Decoupling System 1 execution and System 2 planning. Active perception converts grounding into an iterative process. **GUI-Eyes** (Chen et al., 2026a) lets the agent autonomously invoke visual tools (crop, zoom) before making coordinate predictions, with tool-use decisions learned efficiently through GRPO. Crucially, findings from **GUI-G^2** (Tang et al., 2025a) and **InfiGUI-G1** (Liu et al., 2026) reveal a counter-intuitive phenomenon: forcing explicit Chain-of-Thought (CoT) reasoning for coordinate prediction actually *hurts* grounding accuracy. This serves as a powerful pushback against the prevailing LLM expectation that "CoT improves everything." It uncovers a fundamental principle for GUI agents: **decisions demand thought, but execution demands reflex**. Consequently, state-of-the-art architectures are decoupling multimodal formulation into a *System 1* (fast, intuitive direct coordinate regression for UI localization) and a *System 2* (slow, deliberative logical planning for task strategy). Forcing a model to articulate text descriptions before localizing a pixel disrupts its spatial representations, explaining why direct action heads now robustly outperform text-mediated spatial grounding.

Attention-alignment methods attack the same problem from the model-internals side. **GUI-AIMA** (Zhou et al., 2025b) designs patch-level labels and an `<ANCHOR>` token for intrinsic multimodal attention alignment. Alternatively, **GUI-Actor** (Wu et al., 2025c) bypasses explicit coordinate regression entirely by predicting a heatmap directly on the feature map through an attention-driven action head. Together, these approaches demonstrate that visual grounding is substantially improved by reshaping *how* the model attends.

5.3.3 Memory and Planning: Sustaining Context over Long Horizons

GUI tasks are inherently non-Markovian. Effective solutions selectively compress the screenshot history, treating memory management as a *learned behavior* jointly optimized with the policy. Emerging approaches include **MemR** (Du et al., 2025) for memory modeling, **MemSearcher** (Yuan et al., 2025) for memory retrieval, **auto-scaling continuous memory** (Wu et al., 2025f), **ELMUR** (Cherepanov et al., 2025) for extending effective horizons in partially observable settings, and **AgentProg** (Tian et al., 2025) for program-guided context management.

Dynamic textual compression is the most direct strategy. **WebAgent-R1** (Wei et al., 2025b) has the agent output a textual summary alongside each action to drastically reduce token consumption. **MGA** (Cheng et al., 2025) further structures this idea through independent context state triplets managed by an Abstract Memory Agent. Similarly, **MAGNET** (Sun et al., 2026) constructs a memory-driven knowledge evolution framework that dynamically updates a skill library from environmental feedback. **HAR** (Wang et al., 2025f) complements these strategies with a reflective learning process and a Think-More-Than-Step policy that explicitly re-examines past decisions before acting.

The overarching principle is that memory compression is not a pre-processing step but a learned capability. For long-horizon tasks, **Plan-and-Act** (Erdogan et al., 2025) explicitly separates planning from execution to sustain coherent behavior. In GUI automation, reward design, data collection, perception, and memory management form a coupled system that determines whether the agent can sustain coherent behavior in complex real-world tasks.

6 Training Resources

Robust training of RL-based GUI agents requires a comprehensive ecosystem spanning interactive environments for policy learning, large-scale datasets for pre-training, and specialized frameworks for implementing RL algorithms on multimodal models. This section provides a systematic overview of the training resource landscape that underpins the RL paradigms (Section 4) and cross-cutting dimensions (Section 5) discussed above. Rather than re-describing algorithmic innovations, we focus on the characteristics of each resource and its role in the RL training pipeline.

6.1 Datasets

The efficacy of RL-based GUI agents fundamentally depends on the quality and diversity of training data. Unlike traditional supervised learning paradigms that prioritize scale and human similarity, RL-centric datasets must satisfy distinct requirements: completeness of state representations (for Critic networks), density of reward signals (for sparse reward mitigation), and environmental interactivity (for large-scale online exploration). This section systematically categorizes the data landscape into three strategic dimensions that collectively enable the RL training pipeline—from policy cold-start to self-evolution.

6.1.1 Demonstration and Trajectory Datasets

Demonstration datasets serve as the cornerstone for policy initialization and offline RL, addressing the fundamental challenge of exploration in high-dimensional action spaces where pixel-level click actions can reach millions of possibilities. These datasets vary significantly in their structural characteristics, each offering distinct advantages for different RL training requirements. Table 2 summarizes the key demonstration and trajectory datasets.

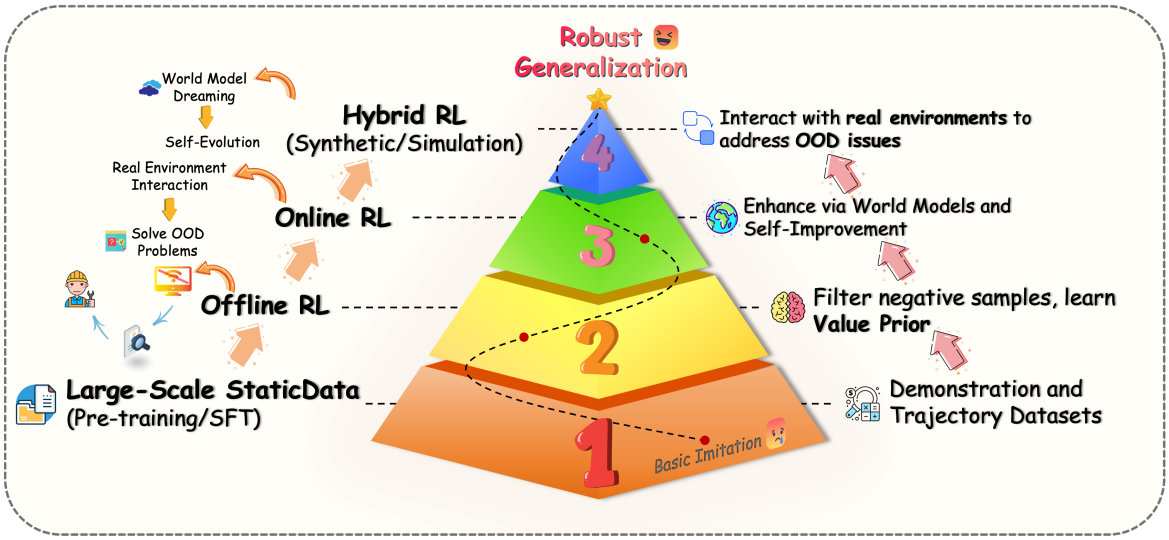

Figure 6: This pyramid depicts a four-stage data-training pipeline for agent capability, progressing from static data imitation to offline RL, synthetic simulation, and online RL, to achieve robust generalization.

Table 2: Demonstration and trajectory datasets for policy initialization and offline RL.

Dataset	Platform	Scale	RL Role
AitW	Mobile	715K traj.	Offline pre-training
AndroidControl	Mobile	—	Hierarchical credit assignment
Mind2Web	Web	2K+ tasks	Action space compression
OmniACT	Desktop/Web	9.8K tasks	Macro-action generation

For mobile platforms, **Android-in-the-Wild (AitW)** (Rawles et al., 2023) provides the largest publicly available corpus with 715K trajectories spanning the Google application ecosystem. Its visual-only dependency—lacking DOM or accessibility trees—forces agents to develop robust pure-vision policies, proving advantageous for generalization to real-world applications (games, Flutter/Unity apps) that deny structured UI access. Notably, AitW's inclusion of human operation noise (mis-taps, hesitations, failed swipes) transforms traditional liabilities into assets for offline RL: through *Advantage Weighting* techniques, models learn to distinguish high-value from low-value actions by mining sub-optimal trajectories, as demonstrated by DigiRL's successful offline pre-training (Section 4.3.2). In contrast, **AndroidControl** (Li et al., 2024a) prioritizes structural richness over scale, introducing hierarchical instruction annotations that directly address the credit assignment problem. By providing both high-level intents and intermediate low-level instructions (e.g., "open clock app," "tap add button"), it enables *Hierarchical Reinforcement Learning (HRL)* where sparse terminal rewards decompose into dense step-wise signals. Its complete XML View Hierarchy metadata further supports structured state representations via Graph Neural Networks—critical for stable value function estimation in offline algorithms like IQL and CQL.

Web environments present fundamentally different challenges, as action spaces are inherently discrete (selecting DOM elements) rather than continuous. **Mind2Web** (Deng et al., 2023) addresses this by spanning 2000+ tasks across 137 websites with comprehensive DOM tree annotations, enabling training of efficient *Grounding Models* that compress action spaces from $O(10^4)$ to $O(10^1)$ candidates via semantic filtering—making RL optimization computationally feasible. Its cross-site diversity enforces learning of HTML tag semantics (e.g., `<input type="search">` universally indicates search functionality) rather than brittle coordinate memorization, yielding policies that generalize beyond training domains. For desktop environments, **OmniACT** (Kapoor et al., 2024) pioneered the *Action-as-Code* paradigm where agents generate executable Python scripts (PyAutoGUI) rather than atomic actions. This structured action space fundamentally alters the RL time horizon: instead of executing dozens of fragile atomic clicks, agents output coherent macro-

Table 3: Perception and grounding datasets for reward shaping and state evaluation.

Dataset	Platform	Scale	RL Role
ScreenSpot Series	Cross-platform	—	IoU-based reward shaping
Ferret-UI	Mobile	—	Visual CoT; anti-hallucination
Rico	Mobile	66K screens	Grounding pre-training
Screen2Words	Mobile	112K pairs	State compression
Widget Captioning	Mobile	163K caps.	Semantic state abstraction
UIGuard	Cross-platform	—	Safety reward signals

action scripts, shortening episode lengths and enabling more effective sparse reward propagation across its 9802 desktop/web tasks.

6.1.2 Perception and Grounding Datasets

Accurate perception forms the sensory foundation for RL value networks and reward models—in open GUI environments lacking API-level feedback, visual understanding is the sole mechanism for state evaluation and task completion verification. These grounding datasets train the "judges" that enable RL agents to assess their own performance, with applications spanning reward shaping, hallucination reduction, and state compression. Table 3 provides an overview of the principal perception and grounding datasets.

The **ScreenSpot Series (V1/V2/Pro)** (Li et al., 2025b) has emerged as the gold standard for visual grounding precision, directly enabling *reward shaping* in RL pipelines. In GRPO training loops (e.g., UI-R1), when an agent outputs a click action (x, y) without immediate environmental feedback, a reward model fine-tuned on ScreenSpot computes the Intersection-over-Union (IoU) between predicted coordinates and ground-truth UI elements, yielding dense signals that guide policy optimization without requiring environmental interaction during early training phases. ScreenSpot-Pro's high-resolution challenges specifically target modern MLLM hallucination issues at production scales. Complementing coordinate-level precision, **Ferret-UI** (You et al., 2024; Li et al., 2024b) tackles mobile-specific visual reasoning through *any-resolution* adaptability—standard vision encoders (CLIP's 224×224 squares) severely distort mobile screenshots' elongated aspect ratios. Ferret-UI's fine-grained regional annotations enable *Visual Chain-of-Thought (CoT)* capabilities where agents verbalize spatial reasoning before acting, measurably reducing hallucination behaviors during RL exploration. Similarly, **Rico** (Deka et al., 2017) provides rich UI element annotations for Android interfaces that support both grounding model pre-training and UI understanding tasks.

Beyond localization, semantic understanding datasets address the critical challenge of *state compression* for long-horizon tasks. **Screen2Words** (Wang et al., 2021) and **Widget Captioning** (Li et al., 2020) convert pixel states into textual descriptions—encoders trained on these datasets compress high-dimensional visual states into concise semantic summaries (e.g., "login page with username/password fields"), enabling RL agents to maintain textual memory of multi-page workflows while reserving pixel-level processing for the current frame only. This hybrid multimodal state representation is essential for scaling RL to tasks spanning dozens of interface transitions. For safety-critical applications, **UIGuard** (Chen et al., 2023) provides dark pattern detection annotations—UI designs that mislead users toward unintended actions. These negative samples enable construction of *safety reward functions* that impose penalties when agents attempt interaction with deceptive elements, implementing Safe RL principles for production deployment.

6.1.3 Synthetic and RL-Generated Corpora

The frontier of RL-based GUI agents has shifted toward *"environment-as-data"* paradigms where agents generate unbounded training curricula through autonomous interaction—static datasets, regardless of scale, suffer from distribution mismatch that self-generated on-policy data eliminates. This category encompasses both interactive environments that enable online reinforcement learning and frameworks that synthesize reasoning-augmented trajectories.

Table 4: Synthetic and RL-generated corpora for on-policy data generation and self-improvement.

Dataset	Platform	Scale	RL Role
AndroidWorld	Mobile	116 tasks	Ground-truth reward signals
WebArena	Web	812 tasks	Executable state verification
VisualWebArena	Web	910 tasks	Visual reward verification
OSWorld	Desktop	369 tasks	Cross-app state tracking
GUI-Bee	Cross-platform	—	Entropy-driven exploration
Explorer	Web	94K+ traj.	Instruction reverse-engineering
UI-TARS	Cross-platform	—	Reasoning-augmented triplets
GUI-R1	Cross-platform	3K samples	Emergent reasoning patterns

Dynamic interactive environments form the foundation of this paradigm. **AndroidWorld** (Rawles et al., 2024) functions as the definitive "Gymnasium" for mobile RL, generating millions of task variants through parameterized templates (e.g., "add contact" with randomized names/numbers) that prevent rote memorization. Its *non-invasive state inspection* via ADB interfaces delivers 100% accurate ground-truth reward signals by querying Android's underlying SQLite databases—enabling agents like **AppAgent** (Zhang et al., 2025c) and **UI-TARS** (Qin et al., 2025) to perform millions of trial-and-error interactions that form self-improving data flywheels. For web environments, **WebArena** (Zhou et al., 2023) and **VisualWebArena** (Koh et al., 2024) provide self-hostable simulated internet platforms with executable verification—running backend scripts to validate database state changes rather than shallow HTML comparison. Additional web benchmarks include **WebCanvas** (Pan et al., 2024) for online evaluation, **BearCubs** (Song et al., 2025d) for web agent benchmarking, **WebWalker** (Wu et al., 2025a) for LLM-based web traversal, **Agent-X** (Ashraf et al., 2025) for evaluating deep reasoning, and **TheAgentCompany** (Xu et al., 2024a) for real-world enterprise tasks. Beyond browsing-based approaches, **Song et al. (2025e)** explored API-based web agents, while end-to-end navigation with **VLMs** (Goetting et al., 2024) demonstrated direct visual navigation capabilities. **WebAgent-R1** demonstrated that synthetic success trajectories generated through parallel exploration in WebArena outperform human demonstrations, as self-generated data reflects agents' actual capability boundaries. **OSWorld** (Xie et al., 2024) extends this paradigm to desktop environments, providing file system state tracking and cross-application workflow support essential for complex multi-app tasks.

Complementing task-directed environments, exploration-focused approaches address cold-start and generalization challenges. **GUI-Bee** (Fan et al., 2025) introduces *exploration data*—trajectories generated via entropy-maximizing autonomous exploration rather than goal completion. Through Q-ICRL mechanisms, agents construct exploration graphs mapping state transition structures and navigation dead-ends, enabling zero-shot adaptation to unseen applications (analogous to humans casually familiarizing themselves with new software). The **Explorer** (Pahuja et al., 2025) framework employs multi-agent pipelines where "explorers" randomly walk through web environments discovering novel states while "annotators" reverse-engineer natural language instructions, synthesizing 94K+ high-quality trajectories covering long-tail scenarios.

Most recently, reasoning-augmented data generation has emerged as a critical frontier inspired by DeepSeek-R1's success. **UI-TARS** (Qin et al., 2025) and **AutoPlay** (Ramrakhya et al., 2025) generate *(State, Thought, Action)* triplets where teacher models (e.g., GPT-4o) produce detailed reasoning chains ("I need to click the search bar because the desired product isn't visible on the homepage..."), with lightweight verifiers filtering logically inconsistent samples. These datasets enable training of *Process Reward Models (PRMs)* that reward intermediate reasoning validity—not just final outcomes—guiding agents from blind trial-and-error toward logical problem decomposition. The resulting data flywheel—iteratively generating, filtering, and fine-tuning on reasoning traces—has propelled continuous SOTA improvements, as exemplified by **GUI-R1**'s extreme data efficiency with merely 3K curated samples (Section 4.1.3).

6.2 Interactive Environments

Unlike static benchmarks used primarily for evaluation, training environments for online RL must support the standard Markov Decision Process (MDP) interface and efficient resetting mechanisms. The evolution of

GUI RL environments has progressed from synthetic sandboxes toward high-fidelity digital twins of real-world interfaces, addressing specific challenges in state representation, action space design, and reward engineering.

6.2.1 Web and Browser Environments

Web environments benefit from standardized rendering protocols (HTML/CSS/JS), evolving from synthetic micro-tasks to full internet simulations. Early foundational work like **MiniWoB++** (Liu et al., 2018a) isolated interaction primitives in HTML5 sandboxes, exposing dual modalities but imposing extremely sparse rewards that demanded distributed training solutions like **CC-Net** (Humphreys et al., 2022).

The field subsequently shifted toward complex semantic understanding. **WebShop** (Yao et al., 2022) simulated a vast e-commerce platform with dense attribute-overlap rewards, successfully demonstrating sim-to-real transfer. Modern approaches operate on real-world snapshots: **Mind2Web** (Deng et al., 2023) preserves webpage states for deterministic replay and compresses DOM action spaces via semantic filtering. **WebArena** (Zhou et al., 2023) advances this with self-hostable platforms supporting executable verification, powering frameworks like **WebRL** (Section 4.2.2) to utilize self-evolving curricula and Outcome-supervised Reward Models. To unify this fragmented landscape, **BrowserGym** (Chezelles et al., 2024) aggregates major benchmarks (**WebArena**, **MiniWoB++**, **VisualWebArena** (Koh et al., 2024), **WebChoreArena** (Miyai et al., 2025)) under a standardized API that encapsulates critical infrastructure.

6.2.2 Desktop and OS Environments

Desktop environments introduce higher-dimensional challenges like file management and multi-app switching. Their engineering foundation relies on headless virtualization: **OSWorld** (Xie et al., 2024) utilizes Docker and QEMU with virtual framebuffers to parallelize environments for massive sample collection. Crucially, it employs execution-based evaluation to inspect side-effects directly (e.g., file state changes), neutralizing hallucination issues common in text-matching and revealing a substantial human-machine gap.

Optimizing these environments focuses on action space design and training stability. **ComputerRL** (Lai et al., 2025) introduces an *API-GUI hybrid action paradigm* to leverage both API determinism and GUI universality, combating entropy collapse during long-horizon tasks via an interleaved RL and SFT *Entropulse* strategy. For perception robustness, **ScreenAgent** (Niu et al., 2024) uses VNC protocols for pure pixel-stream control independent of Accessibility APIs, adopting a self-correcting *Plan-Act-Reflect* loop. Ultimately, scaling desktop environments depends on infrastructure maturity alongside algorithmic innovation.

6.2.3 Mobile Environments

Mobile platforms require specialized environments due to dense, gesture-based controls and isolated app ecosystems. **AndroidEnv** (Toyama et al., 2021) provides a standard MDP, modeling continuous virtual finger movements that extend time horizons and complicate credit assignment. Due to standard emulators' high resource costs, recent tools like **UISim** (Xiang et al., 2025) offer streamlined image-based UI simulators, typically deployed with massive parallelization.

Task and reward designs have also advanced. **AndroidWorld** (Rawles et al., 2024) prevents overfitting through dynamic task parameterization and extracts zero-noise rewards by directly querying system internals. To combat reward sparsity, **Mobile-Env** (Zhang et al., 2023) employs background evaluators to monitor system states and provide dense intermediate feedback.

To overcome covariate drift inherent in static behavioral cloning, the field is shifting toward dynamic interaction paradigms. **DigiRL** (Section 4.3.2) combines offline and online RL. Modern infrastructures easily scale this paradigm by decoupling CPU simulation from GPU inference (e.g., **MAI-UI** (Zhou et al., 2025a), **MobileGUI-RL** (Shi et al., 2025)), often leveraging GRPO with efficiency rewards to train agents that discover concise operational paths.

6.2.4 Cross-Platform Trends and Synthesis

Several cross-cutting trends emerge from this environmental evolution. First, *visual-first state representation* is becoming dominant: as structured metadata access is increasingly precluded, convergence is driven toward pixel-first approaches (e.g., **ScreenAgent**) that prioritize cross-platform generality. Second, *action spaces are evolving* from raw coordinates toward API-GUI hybrids (**ComputerRL**) or coordinate-free semantic grounding (**GUI-Actor**). Recent benchmarks like **OSWorld-MCP** (Jia et al., 2025) and **MCPWorld** (Yan et al., 2025e) formalize tool invocation and hybrid API/GUI evaluation. Third, *reward engineering* has progressed from binary system state verification (**AndroidWorld**, **OSWorld**) to learned Outcome-supervised Reward Models (**WebRL**); as task complexity increases, deterministic verification becomes infeasible, driving the adoption of RLAIF paradigms. Finally, *Sim-to-Real gaps* persist; successful transfer pipelines combine offline pretraining, simulated fine-tuning, and real-world deployment (e.g., **DigiRL**), with domain randomization emerging as a critical technique.

6.3 RL Infrastructure and Tools

Constructing scalable training loops for Multimodal LLM-based GUI agents requires specialized infrastructure addressing critical computational bottlenecks unique to this domain. Unlike text-only RL systems where training is typically compute-bound, GUI agent training transitions to *I/O-bound* workloads: environment rendering, screenshot transmission, and perception processing dominate execution time, making traditional synchronous RL frameworks inefficient. The emerging infrastructure ecosystem comprises four interdependent dimensions: VLM-RL algorithm libraries optimizing for multimodal inputs, distributed architectures decoupling rollout and training, reward engineering tools solving the sparse/deceptive feedback problem, and memory management systems enabling long-horizon coherent reasoning.

6.3.1 VLM-RL Algorithm Libraries and Framework Evolution

The foundational algorithmic layer has evolved from generic RLHF libraries (Hu et al., 2024) toward GUI-specialized implementations balancing computational efficiency with learning signal quality.

GRPO and Memory-Efficient Policy Optimization: Group Relative Policy Optimization (GRPO) has emerged as the dominant algorithm for GUI RL, employed in systems like **Mano**, **GUI-R1**, **InfiGUI-G1**, and **GUI-Eyes**. GRPO eliminates the memory-intensive Critic network, a critical optimization for long visual sequences; it achieves advantage estimation across trajectory groups instead of learned value functions:

$$\hat{A}_i = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}, \qquad \mathcal{L}_{\text{GRPO}}(\theta) = -\frac{1}{G}\sum_{i=1}^{G} \min\Big(\rho_i \hat{A}_i,\ \text{clip}(\rho_i, 1-\epsilon, 1+\epsilon)\hat{A}_i\Big) + \beta\, D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}})$$

where $\rho_i = \pi_\theta(o_i|x)/\pi_{\text{old}}(o_i|x)$. Libraries like **veRL** (Sheng et al., 2025) and **ReaL** (Mei et al., 2025) provide production-grade GRPO implementations. They achieve dramatic throughput improvements through *HybridFlow* (Sheng et al., 2025), a decoupled paradigm enabling Actor models to transition seamlessly between inference backends (**vLLM**, built on PagedAttention (Kwon et al., 2023)) and training frameworks (**Megatron-LM** (Shoeybi et al., 2019), **FSDP** (Zhao et al., 2023)) without redundant weight replication.

Hybrid Offline-to-Online Frameworks: To address sample inefficiency during cold-start phases, frameworks increasingly support offline-to-online transitions. **OpenRLHF** (Hu et al., 2024) implements Advantage-Weighted Regression (AWR) to extract behavioral priors from static demonstrations before online refinement. **DigiRL** (Section 4.3.2) exemplifies this by combining offline initialization with online fine-tuning via instruction-level value functions.

Generative and Reasoning-Augmented Reward Models: Emerging frameworks integrate generative reward modeling, where VLMs directly compare states. **RewardDance** (Wu et al., 2025b) reformulates this as a binary classification task ("Is state s_A better than state s_B?"), using the positive token's log-probability as the reward:

$$r = \log P(\text{“Yes”} \mid s_A, s_B, \mathcal{O})$$

This approach utilizes foundation models' full representational capacity and exhibits robust resistance to reward hacking.

6.3.2 Distributed Rollout and Training Architectures

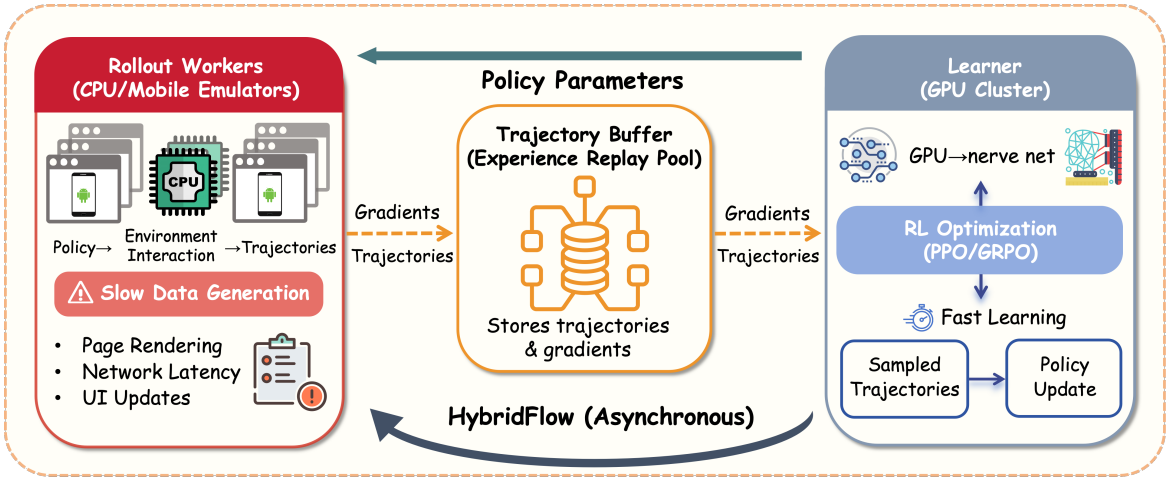

Figure 7: An asynchronous distributed architecture for GUI RL agent training, addressing slow environment interaction latency. It decouples slow data generation (CPU/mobile emulators as Rollout Workers) from fast GPU-cluster learning. Rollout Workers feed trajectories/gradients to a buffer, while the Learner sends updated policy parameters back asynchronously via HybridFlow, enabling massive parallelism and efficient GPU utilization.

The infrastructure transition from synchronized training to fully asynchronous architectures represents a fundamental paradigm shift driven by I/O latency. A single GUI environment step (screenshot capture, OCR, action execution, rendering) commonly takes 0.5–2 seconds, rendering synchronous PPO inefficient: GPUs remain idle during rollout phases while rollout workers stall during training phases. Modern systems decouple these workloads entirely.

Fully Asynchronous Actor-Trainer Separation: AReaL (Fu et al., 2025b) pioneered production-scale asynchronous RL for long-horizon agent tasks. Its architecture partitions workflows into decoupled *Rollout Workers* (environment-facing processes continuously sampling trajectories) and *Trainer Workers* (GPU-centric processes consuming data from replay buffers). This decoupling introduces *data staleness*—when trainers update policy $\pi_{\theta_{t+1}}$, rollout workers may still use π_{θ_t}—yet AReaL's algorithm-system co-design mitigates this through PPO variants tolerant of stale data. Benchmark results demonstrate 3× acceleration over synchronous systems, with linear scaling to 1000+ GPUs. The architectural pattern is now standard across research and production systems.

Heterogeneous Hardware Scheduling: HETHUB (Xu et al., 2024b) addresses the practical reality that GPU clusters are rarely homogeneous. Modern datacenters mix NVIDIA A100 (high-bandwidth, suited for vision encoding), H100 (compute-intensive decoders), and consumer GPUs. HETHUB's automatic parallel planner analyzes hardware specifications and dynamically assigns model components: vision encoders to bandwidth-optimized A100s, transformer layers to compute-dense H100s, etc. This fine-grained scheduling reduces cluster completion time by 30–50% compared to naive partitioning. DistRL (Wang et al., 2024d) and Agent.xpu (Wei et al., 2025a) further extend this paradigm to edge and mobile scenarios: DistRL implements centralized training with decentralized rollout across mobile devices, while Agent.xpu schedules antagonistic tasks (low-latency user interaction vs. high-throughput RL training) on System-on-Chip devices through kernel-level preemption.

API-GUI Hybrid Action Paradigms: ComputerRL introduces a critical architectural innovation: permitting agents to choose between low-level GUI actions (pixel coordinates) and high-level system APIs. This hybrid paradigm allows agents to call `get_file_content(path)` instead of laboriously opening a file

browser, reducing trajectory length and enabling learning in extremely constrained sample budgets. The unified action interface abstracts platform-specific implementation (Win32 APIs, X11 calls, macOS Cocoa) behind a standard interface, simplifying distributed training across heterogeneous operating systems.

6.3.3 Reward Engineering and Verification Systems

Reward design is a critical bottleneck, balancing between uninformative sparse terminal rewards and exploitable hand-crafted dense signals. Modern systems address this tension through multi-layered architectures. To establish reliable feedback, VAGEN (Cui et al., 2026) introduces **Agentic Verification**, replacing passive LLM observers with active environmental probing—such as executing system commands to verify side effects—to enable noise-free Reinforcement Learning from Verifiable Rewards (RLVR). To overcome long-horizon sparsity, **Process Reward Models** have emerged; ProgRM (Zhang et al., 2025d), for instance, automatically extracts intermediate milestones from demonstrations to provide dense progress estimations. Concurrently, methods including InfiGUI-G1 (Liu et al., 2026) and Mano (Fu et al., 2025a) utilize **Composite Rewards** that integrate multiple constraints—such as IoU, realistic bounding box sizes, and format validity—to prevent reward hacking and mode collapse. Finally, **Autonomous Evaluation** pipelines, as demonstrated by systems like ZeroGUI, leverage aggregated VLM-as-judge scoring to construct self-evolving, zero-human-cost training curricula.

6.3.4 Memory Management and Long-Horizon Reasoning

Extended interaction horizons pose acute challenges to context management: full trajectory concatenation becomes prohibitively expensive, yet lossy compression risks critical information loss. Contemporary systems implement learnable and adaptive memory mechanisms.

Reinforcement Learning-Driven Memory Operations: Memory-R1 (Yan et al., 2025d) treats memory management as a learnable decision process. A Memory Manager Agent operates an action space $\{\texttt{ADD}, \texttt{UPDATE}, \texttt{DELETE}, \texttt{NOOP}\}$ on an external memory store. Critically, this Agent only receives reward when updated memory helps solve the downstream task. This outcome-driven training forces agents to actively suppress stale information and consolidate persistent facts, substantially outperforming fixed-length context windows.

Hierarchical Working Memory and Chunking: Hi-Agent (Wu et al., 2025g) (Section 4.3.2) enforces hierarchical decomposition. Agents first propose subgoals, then execute low-level primitive actions. Once achieved, the system automatically collapses the action sequence into a high-level summary, ensuring context windows retain fine-grained details for current subtasks alongside coarse history summaries. This chunking mechanism extends effective planning horizons while maintaining manageable token counts.

Structured State Representation and Memory Triads: Memory-Driven GUI Agent (MGA) (Cheng et al., 2025) decouples reasoning from historical artifacts through a three-component state representation, where Spatial Cues represent extracted UI layout information and Structured Memory contains dynamic summaries. This explicitly separates current decisions from historical influences, reducing trajectory collapse caused by historical context pollution.

6.3.5 Integration and Ecosystem Standardization

Modern infrastructure increasingly emphasizes interoperability. **BrowserGym** (Chezelles et al., 2024) provides a unified Gymnasium API aggregating diverse benchmarks (**WebArena**, **MiniWoB++**, **VisualWebArena**) while handling infrastructure concerns like Docker sandboxing and DOM parsing. Similarly, **GUI-MCP** standardizes tool-calling interfaces, allowing reward functions to universally verify task completion across heterogeneous applications. Open-source scaffolding platforms like **OpenHands** (Wang et al., 2024f) and **AutoGen** (Wu et al., 2024a) wrap raw LLMs with memory management, tool interfaces, and trajectory logging, reducing engineering friction for practitioners building RL systems.

7 Challenges and Future Directions

The preceding sections show that RL already improves GUI agents along three tightly coupled axes: reward design, data efficiency, and long-horizon decision making. Looking forward, the central question is no longer whether RL is useful for GUI automation, but what kind of agents these training paradigms are ultimately producing. In our view, the next stage of the field is not simply "better tool use," but the emergence of agents that can persist, adapt, and act reliably within evolving software ecosystems.

7.1 Digital Worlds

We first discuss the conceptual transition from task-specific GUI agents to persistent computer-use agents embedded in broader digital environments.

7.1.1 Digital Inhabitants

We use the term **digital inhabitants** to describe a stronger class of computer-use agents: systems that do not merely execute isolated instructions on a screen, but maintain persistent competence within digital environments. A digital inhabitant should be able to internalize interface regularities, adapt to software updates, accumulate reusable experience across tasks, and operate under stable behavioral constraints over long horizons. This perspective extends GUI agents from one-shot task solvers to continual actors embedded in a broader digital world.

For RL-based GUI agents, this shift is especially natural. As discussed in Sections 4 and 5, GUI environments expose sequential structure, delayed rewards, and partial observability in a form that is difficult to handle with static imitation alone. RL provides the mechanism for converting interaction into competence. More importantly, it offers a path toward agents that learn not only *which* action sequence succeeds once, but *why* certain interface patterns, failure modes, and recovery strategies recur across applications and platforms. In this sense, GUI agents may become the most practical substrate for studying general computer-use agents: they sit at the boundary between narrow application automation and open-ended digital interaction.

7.1.2 Agent-Native Environments

At the same time, a fully developed digital inhabitant may not ultimately operate within computers designed primarily for humans. Current GUI agents are trained to act through human-oriented abstractions such as windows, icons, forms, and cursor-level manipulation. This setting is important because it covers the existing software world, but it may also be transitional. In the longer run, the natural endpoint is likely to be *agent-native operating environments*: operating systems, execution substrates, and even hardware interfaces designed explicitly for machine actors rather than retrofitted from human-computer interaction.

From this perspective, today's GUI agents play a dual role. In the short term, they are practical automation systems for the legacy digital ecosystem. In the long term, they are a bridge technology that reveals which components of computer use should remain embodied and interactive, and which should be re-designed into machine-readable primitives. The future infrastructure of agent society may therefore include persistent agent identities, explicit permission and accountability layers, auditable action logs, machine-native task protocols, and regulatory rules that define what an autonomous agent is allowed to perceive, remember, exchange, and execute. Only within such infrastructure can computer-use agents become true digital inhabitants rather than highly capable users of human software.

To make this bridge actionable for practitioners and standards bodies, we see three near-term standardization targets. First, interfaces should expose *machine-readable UI schemas* that provide stable semantics for roles, affordances, constraints, and state transitions beyond raw pixels. Second, platforms should define *verifiable outcome APIs* that expose task completion evidence, side-effect traces, and policy-compliance checks in a form that can be used directly by reward and evaluation pipelines. Third, the community needs *reference sandbox specifications* that formalize permission scopes, rollback behavior, logging requirements, and human override mechanisms, so that training and deployment can be compared under shared safety assumptions.

7.2 Technical Roadmap

The next set of challenges concerns the technical conditions under which RL can support robust, scalable, and generalizable computer-use behavior.

7.2.1 Reward Interfaces

One of the clearest lessons of this survey is that verifiability is both the main opportunity and the main bottleneck for RL in GUI agents. Rule-based rewards, LLM-as-judge signals, and learned reward models (Section 5) have made it possible to train agents on increasingly realistic tasks, yet each remains incomplete. The difficulty is that real computer-use goals are rarely exhausted by terminal success predicates. Tasks such as purchasing, scheduling, document editing, or enterprise workflow execution require semantic correctness, procedural compliance, and often user-specific preferences, none of which are fully captured by URL changes or form submission events.

This suggests an important future direction: reward design must move from *task completion* toward *intent satisfaction under constraints*. For GUI agents, that means richer evaluation pipelines that jointly assess outcome quality, process correctness, and recoverability after mistakes. For computer-use agents more broadly, it implies that RLVR will increasingly depend on layered evaluators, combining executable checks, environment feedback, and model-based judgment. The open problem is not simply building stronger reward models, but constructing reward interfaces that remain reliable as the task space expands from benchmark-style episodes to open-world digital work.

7.2.2 I/O-Constrained Learning

As argued in Section 5.2, environment interaction in GUI settings is fundamentally slow. Rendering, network delay, and screenshot transmission make online RL expensive in a way that is qualitatively different from classic simulators. This I/O wall is not just an engineering nuisance; it is a structural constraint that shapes which learning strategies are viable. It explains why progress has increasingly relied on hybrid pipelines that combine demonstrations, offline optimization, selective online exploration, and synthetic data generation.

Table 5 makes this constraint more operational. The numbers are best read as order-of-magnitude planning ranges than fixed benchmark results, since latency depends on browser engine, emulator density, network locality, screenshot resolution, and reward instrumentation. Even under optimistic assumptions, a single live GUI environment usually produces only sub-Hz to low-Hz interaction streams, while parallel rollout primarily hides I/O stalls rather than eliminating them.

The practical implication is that scaling online GUI RL is rarely a matter of adding GPUs alone. Training systems must either increase environment multiplicity through asynchronous rollout workers, reduce per-step observability costs through state compression and pruning, or shift more exploration into surrogate dynamics before periodically re-grounding on live interfaces.

This also makes automation of the training stack increasingly important. AutoRL-style methods (Afshar et al., 2022) can reduce manual search over curricula, reward weights, rollout schedules, and model sizes, while infrastructure-centered systems such as AgentCPM-Explore (Chen et al., 2026b) show that reward denoising and context compression are first-order design choices under real browser and app I/O noise.

For this reason, world models and latent-space training are likely to become more central rather than less. A promising long-term direction is to train agents that can alternate between two regimes: grounded interaction with the real interface, and accelerated imagination over learned interface dynamics. Such models would not replace real environments, because final success still depends on precise grounding in pixels, latency, and platform-specific behaviors. However, they could substantially reduce the cost of exploration, improve long-horizon credit assignment, and enable counterfactual reasoning about alternative action sequences before expensive execution. For GUI agents, this is a path to practical RL at scale; for computer-use agents, it is a step toward building internal models of how software ecosystems behave.

Table 5: Representative GUI step-time bottlenecks and rollout throughput across common training settings (environment-side latency only; model inference excluded).

Setup	Rendering /capture	Network /backend	DOM /state parsing	Single-env throughput (steps/s)	Parallel-env throughput (steps/s)
Syn-Web (MiniWoB++, BrowserGym)	0.5-2.0s	negligible–50 ms	10–80 ms	5–20	100–800 (64 envs)
Self-Web (WebArena, VWA)	0.3–1.0 s	0.1–0.8 s	0.1–0.5 s	0.5–2	15–150 (32–128 envs)
Desktop VM (OSWorld, ScreenAgent)	0.5–2.0 s	var. file/app I/O	0.1–1.0 s	0.2–1.5	5–60 (16–64 VMs)
Mobile Emu (AndroidEnv, AndroidWorld)	0.5–2.0 s	0–1.0+ s	0.1–0.5 s	0.3–2	20–300 (32–256 emus)
WM/Latent rollout	bypassed	bypassed	light symbolic state	10–100+ (sim)	10^3–10^4 (sim)

Abbreviations: Syn-Web = synthetic browser tasks; Self-Web = self-hosted web benchmarks; VWA = VisualWebArena; Emu = emulator; WM = world model. Ranges denote order-of-magnitude planning values and exclude policy/model inference time. Columns are not strictly additive: components (rendering, network, parsing) run partially in pipeline, and throughput reflects end-to-end interaction rates under practical parallelism and caching. Representative sources: MiniWoB++ and CC-Net (Liu et al., 2018a; Humphreys et al., 2022); WebArena, VisualWebArena, and BrowserGym (Zhou et al., 2023; Koh et al., 2024; Chezelles et al., 2024); OSWorld and ScreenAgent (Xie et al., 2024; Niu et al., 2024); AndroidEnv and AndroidWorld (Toyama et al., 2021; Rawles et al., 2024); DreamGym and UI-Simulator (Chen et al., 2025a; Wang et al., 2025d).

7.2.3 Hierarchical Control

Another recurring insight from this survey is that computer use is not a monolithic reasoning problem. The evidence reviewed in Section 5.3.2 shows that explicit deliberation can improve planning but degrade visual grounding, while Section 5.3.3 shows that long-horizon success depends on learned memory compression and retrieval rather than ever-longer context windows. Taken together, these findings point toward a hierarchical view of future agents: fast perceptual-action loops for local execution, slower reasoning modules for strategy shifts, and memory systems that preserve task-relevant state across long trajectories.

Here RL can play a broader role than optimizing action tokens alone. It can be used to learn *when to think*, *when to look closer*, *when to retrieve memory*, and *when to ask for help*. This kind of adaptive control is likely to matter even more for general computer-use agents than for current GUI benchmarks, because open environments contain a wider mixture of routine operations and rare high-stakes decisions. A mature computer-use agent should therefore optimize not only task reward, but also its allocation of attention, latency, and reasoning budget.

7.3 Deployment and Governance

Beyond capability, the long-term trajectory of computer-use agents depends on whether they can be deployed safely, evaluated realistically, and integrated into a governed digital ecosystem.

7.3.1 Safety, Adaptation, and Evaluation

The path toward digital inhabitants also raises a harder safety question. Current concerns already include phishing prompts, deceptive layouts, unsafe clicks, and irreversible operations (Zhang et al., 2025j; Kuntz et al., 2025). But once agents become persistent and self-improving, the relevant risk is no longer a single bad action; it is the accumulation of miscalibrated behavior over time. Continual RL, replay-based adaptation, and cross-platform transfer are therefore double-edged: they are necessary for maintaining stable performance and environmental robustness under gradual interface drift, distribution shifts, and dynamic task changes, but they can also propagate unsafe behavioral shortcuts, amplify hidden vulnerability risks, or even gradually erase previously aligned safety guardrails and ethical constraints if not equipped with explicit regularization and strict boundary restrictions. Without well-designed constraint mechanisms and continual safety supervision, these adaptive learning paradigms will easily compromise model alignment and lead to unpredictable risky behaviors in open-ended real-world scenarios.

This is why future evaluation must move beyond static benchmark success rates. For GUI agents, we need protocols that test reliability under interface updates, adversarial perturbations, partial failures, and recovery scenarios. For computer-use agents more broadly, the field will need benchmarks closer to digital operations than to isolated tasks: persistent identities, multi-application workflows, interrupt handling, permission boundaries, and human oversight at varying levels of granularity. In parallel, constrained RL and human-in-the-loop mechanisms should be treated as first-class components of the training objective rather

than deployment-time patches. More broadly, safe deployment will require infrastructure-level guarantees in addition to policy-level alignment: identity systems that make agents legible, rule systems that specify their authority boundaries, and execution environments that support auditing, rollback, and accountability by default.

7.3.2 Future Computer Use

Taken together, these trends suggest that GUI agents are not merely one application area of RL, but a concrete route toward more general computer-use intelligence. They expose the full stack of difficulties that such systems must eventually solve: grounding in messy interfaces, acting under delayed and imperfect feedback, remembering long interaction histories, adapting to non-stationary software, and operating safely under real-world constraints. RL is unlikely to solve all of these challenges alone, but it is the framework that most naturally connects them through sequential optimization.

From the perspective of this survey, the distinctive opportunity is therefore not simply to make agents click more accurately or finish benchmarks more efficiently. It is to develop agents that can *live in* digital environments in the same sense that modern language models can already *speak in* natural language: persistently, adaptively, and under meaningful feedback. Yet the full realization of that vision may require a deeper transition, from agents operating human-oriented computers to agents inhabiting machine-oriented digital worlds. If that transition occurs, the study of reinforcement learning for GUI agents may ultimately be remembered not as a niche subfield, but as an early foundation for the broader science of digital inhabitants and agent-native infrastructure.

8 Conclusion

This survey provides a comprehensive analysis of Reinforcement Learning for GUI agents, covering offline, online, and hybrid paradigms alongside key dimensions like reward engineering, data efficiency, and technical innovations. Three core findings emerge from this landscape. First, the lack of easily readable rewards in GUI environments forces agents to interpret multimodal evidence, driving the adoption of hybrid reward schemes. Second, severe I/O latency makes data efficiency a binding constraint, motivating the use of latent-space world models over on-policy algorithms. Third, while reasoning can emerge from structured action spaces without explicit supervision, balancing fast intuitive grounding with slow deliberative planning remains a fundamental challenge.

Looking ahead, future research will likely focus on process reward models for reasoning trajectories, continual learning for interface updates, cross-platform agents, and dynamic cognitive architectures. As agents inevitably transition to production, establishing formal safety guarantees and real-world deployment benchmarks becomes indispensable. More importantly, the long-term trajectory of the field may extend beyond training agents to operate human-oriented interfaces more effectively. If computer-use agents are to become genuine digital inhabitants, they will likely require not only stronger policies, but also agent-native infrastructure: persistent identities, explicit authority boundaries, auditable execution environments, and operating substrates designed for machine actors. Ultimately, the convergence of reasoning-enhanced VLMs with hybrid RL training signals a fundamental shift in intelligent digital interaction, where RL will remain central to addressing sequential decision-making under uncertainty. We hope this survey inspires further advances not only toward robust, generalizable, and safe GUI agents, but also toward consolidating and expanding the broader theoretical and technical foundations of digital inhabitants.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Reza Refaei Afshar, Yingqian Zhang, Joaquin Vanschoren, and Uzay Kaymak. Automated reinforcement learning: An overview. *arXiv preprint arXiv:2201.05000*, 2022.

Saaket Agashe, Jiuzhou Han, Shuyu Gan, Jiachen Yang, Ang Li, and Xin Eric Wang. Agent s: An open agentic framework that uses computers like a human. *arXiv preprint arXiv:2410.08164*, 2024.

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

Anthropic. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku — anthropic.com. `https://www.anthropic.com/news/3-5-models-and-computer-use`, 2024. [Accessed 09-02-2026].

Tajamul Ashraf, Amal Saqib, Hanan Ghani, Muhra AlMahri, Yuhao Li, Noor Ahsan, Umair Nawaz, Jean Lahoud, Hisham Cholakkal, Mubarak Shah, et al. Agent-x: Evaluating deep multimodal reasoning in vision-centric agentic tasks. *arXiv preprint arXiv:2505.24876*, 2025.

Hao Bai, Yifei Zhou, Jiayi Pan, Mert Cemri, Alane Suhr, Sergey Levine, and Aviral Kumar. Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 12461–12495, 2024.

Hao Bai, Yifei Zhou, Li Erran Li, Sergey Levine, and Aviral Kumar. Digi-q: Learning q-value functions for training device-control agents. *arXiv preprint arXiv:2502.15760*, 2025a.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025b. URL `https://arxiv.org/abs/2511.21631`.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Patrice Bechard, Orlando Marquez Ayala, Emily Chen, Jordan Skelton, Sagar Davasam, Srinivas Sunkara, Vikas Yadav, and Sai Rajeswar. Terminal agents suffice for enterprise automation. *arXiv preprint arXiv:2604.00073*, 2026.

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

ByteDance. Seed News - ByteDance Seed Team — seed.bytedance.com. `https://seed.bytedance.com/en/blog/official-release-of-seed1-8-a-generalized-agentic-model`, 2025. [Accessed 09-02-2026].

S. Cai, Y. Qin, H. Lin, Z. Xu, G. Li, Y. Shi, Z. Li, Y. Mao, S. Cai, X. Tan, Y. Liang, K. Li, and X. Sun. Smartsnap: Proactive evidence seeking for self-verifying agents. *arXiv preprint arXiv:2512.22322*, 2025.

Yuji Cao, Huan Zhao, Yuheng Cheng, Ting Shu, Yue Chen, Guolong Liu, Gaoqi Liang, Junhua Zhao, Jinyue Yan, and Yun Li. Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

C. Chen, J. Shao, D. Lu, H. Hu, X. Liu, H. Yao, and W. Liu. Gui-eyes: Tool-augmented perception for visual grounding in gui agents. *arXiv preprint arXiv:2601.09770*, 2026a.

Haotian Chen, Xin Cong, Shengda Fan, Yuyang Fu, Ziqin Gong, Yaxi Lu, Yishan Li, Boye Niu, Chengjun Pan, Zijun Song, et al. Agentcpm-explore: Realizing long-horizon deep exploration for edge-scale agents. *arXiv preprint arXiv:2602.06485*, 2026b.

Jieshan Chen, Jiamou Sun, Sidong Feng, Zhenchang Xing, Qinghua Lu, Xiwei Xu, and Chunyang Chen. Unveiling the tricks: Automated detection of dark patterns in mobile applications. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–20, 2023.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pp. 15084–15097, 2021.

Z. Chen, Z. Zhao, K. Zhang, B. Liu, Q. Qi, Y. Wu, T. Kalluri, S. Cao, Y. Xiong, H. Tong, H. Yao, H. Li, J. Zhu, X. Li, D. Song, B. Li, J. Weston, and D. Huynh. Scaling agent learning via experience synthesis. *arXiv preprint arXiv:2511.03773*, 2025a.

Zhaorun Chen, Zhuokai Zhao, Kai Zhang, Bo Liu, Qi Qi, Yifan Wu, Tarun Kalluri, Sara Cao, Yuanhao Xiong, Haibo Tong, et al. Scaling agent learning via experience synthesis. *arXiv preprint arXiv:2511.03773*, 2025b.

Weihua Cheng, Ersheng Ni, Wenlong Wang, Yifei Sun, Junming Liu, Wangyu Shen, Yirong Chen, Botian Shi, and Ding Wang. Mga: Memory-driven gui agent for observation-centric interaction. *arXiv preprint arXiv:2510.24168*, 2025.

Egor Cherepanov, Alexey K Kovalev, and Aleksandr I Panov. Elmur: External layer memory with update/rewrite for long-horizon rl. *arXiv preprint arXiv:2510.07151*, 2025.

De Chezelles, Thibault Le Sellier, Sahar Omidi Shayegan, Lawrence Keunho Jang, Xing Han Lù, Ori Yoran, Dehan Kong, Frank F Xu, Siva Reddy, Quentin Cappart, et al. The browsergym ecosystem for web agent research. *arXiv preprint arXiv:2412.05467*, 2024.

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

Rémi Coulom. Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*, pp. 72–83. Springer, 2006.

Chaoqun Cui, Jing Huang, Shijing Wang, Liming Zheng, Qingchao Kong, and Zhixiong Zeng. Agentic reward modeling: Verifying gui agent via online proactive interaction. *arXiv preprint arXiv:2602.00575*, 2026.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.

G. Dai, S. Jiang, T. Cao, Y. Yang, Y. Li, R. Tan, M. Li, and L. Qiu. Prore: A proactive reward system for gui agents via reasoner–actor collaboration. *arXiv preprint arXiv:2509.21823*, 2025.

Google DeepMind. Introducing the gemini 2.5 computer use model. `https://blog.google/innovation-and-ai/models-and-research/google-deepmind/gemini-computer-use-model/`, 2025. [Accessed 09-02-2026].

Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th annual ACM symposium on user interface software and technology*, pp. 845–854, 2017.

Mingkai Deng, Jinyu Hou, Zhiting Hu, and Eric Xing. Simura: A world-model-driven simulative reasoning architecture for general goal-oriented agents. *arXiv preprint arXiv:2507.23773*, 2025.

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

Hang Ding, Peidong Liu, Junqiao Wang, Ziwei Ji, Meng Cao, Rongzhao Zhang, Lynn Ai, Eric Yang, Tianyu Shi, and Lei Yu. Dynaweb: Model-based reinforcement learning of web agents. *arXiv preprint arXiv:2601.22149*, 2026.

Guanting Dong, Licheng Bao, Zhongyuan Wang, Kangzhi Zhao, Xiaoxi Li, Jiajie Jin, Jinghan Yang, Hangyu Mao, Fuzheng Zhang, Kun Gai, et al. Agentic entropy-balanced policy optimization. *arXiv preprint arXiv:2510.14545*, 2025.

Xingbo Du, Loka Li, Duzhen Zhang, and Le Song. Memr^3: Memory retrieval via reflective reasoning for llm agents. *arXiv preprint arXiv:2512.20237*, 2025.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv e-prints*, pp. arXiv–2407, 2024.

Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. Plan-and-act: Improving planning of agents for long-horizon tasks. *arXiv preprint arXiv:2503.09572*, 2025.

Yue Fan, Handong Zhao, Ruiyi Zhang, Yu Shen, Xin Eric Wang, and Gang Wu. Gui-bee: Align gui action grounding to novel environments via autonomous exploration. *arXiv preprint arXiv:2501.13896*, 2025.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *arXiv preprint arXiv:2505.10978*, 2025.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on robot learning*, pp. 158–168. PMLR, 2022.

Tianyu Fu, Anyang Su, Chenxu Zhao, Hanning Wang, Minghui Wu, Zhe Yu, Fei Hu, Mingjia Shi, Wei Dong, Jiayao Wang, et al. Mano technical report. *arXiv preprint arXiv:2509.17336*, 2025a.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, et al. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *arXiv preprint arXiv:2505.24298*, 2025b.

Kanishk Gandhi, Shivam Garg, Noah D Goodman, and Dimitris Papailiopoulos. Endless terminals: Scaling rl environments for terminal agents. *arXiv preprint arXiv:2601.16443*, 2026.

Yifei Gao, Junhong Ye, Jiaqi Wang, and Jitao Sang. Websynthesis: World-model-guided mcts for efficient webui-trajectory synthesis. *arXiv preprint arXiv:2507.04370*, 2025.

Dylan Goetting, Himanshu Gaurav Singh, and Antonio Loquercio. End-to-end navigation with vision language models: Transforming spatial reasoning into question-answering. *arXiv preprint arXiv:2411.05755*, 2024.

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

Yu Gu, Kai Zhang, Yuting Ning, Boyuan Zheng, Boyu Gou, Tianci Xue, Cheng Chang, Sanjari Srivastava, Yanan Xie, Peng Qi, et al. Is your llm secretly a world model of the internet? model-based planning for web agents. *arXiv preprint arXiv:2411.06559*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Nature*, 2025a.

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

Izzeddin Gür, Ofir Nachum, Yingjie Miao, Mustafa Safdari, Austin Huang, Aakanksha Chowdhery, Sharan Narang, Noah Fiedel, and Aleksandra Faust. Understanding html with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2803–2821, 2023.

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

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. *arXiv preprint arXiv:2602.22817*, 2026.

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

Jakub Hoscilowicz and Artur Janicki. Clickagent: Enhancing ui location capabilities of autonomous agents. In *Proceedings of the 26th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 471–476, 2025.

Jian Hu, Xibin Wu, Zilin Zhu, Weixun Wang, Dehao Zhang, Yu Cao, et al. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP)*, 2024.

Xueyu Hu, Tao Xiong, Biao Yi, Zishu Wei, Ruixuan Xiao, Yurun Chen, Jiasheng Ye, Meiling Tao, Xiangxin Zhou, Ziyu Zhao, et al. Os agents: A survey on mllm-based agents for computer, phone and browser use. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7436–7465, 2025.

Zhiyuan Huang, Ziming Cheng, Junting Pan, Zhaohui Hou, and Mingjie Zhan. Spiritsight agent: Advanced gui agent with one look. In *Proceedings of the computer vision and pattern recognition conference*, pp. 29490–29500, 2025.

Peter C Humphreys, David Raposo, Toby Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Alex Goldin, Adam Santoro, and Timothy Lillicrap. A data-driven approach for learning to control computers. *arXiv preprint arXiv:2202.08137*, 2022.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Hongrui Jia, Jitong Liao, Xi Zhang, Haiyang Xu, Tianbao Xie, Chaoya Jiang, Ming Yan, Si Liu, Wei Ye, and Fei Huang. Osworld-mcp: Benchmarking mcp tool invocation in computer-use agents. *arXiv preprint arXiv:2510.24563*, 2025.

Raghav Kapoor, Yash Parag Butala, Melisa Russak, Jing Yu Koh, Kiran Kamble, Waseem AlShikh, and Ruslan Salakhutdinov. Omniact: A dataset and benchmark for enabling multimodal generalist autonomous agents for desktop and web. In *European Conference on Computer Vision*, pp. 161–178. Springer, 2024.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 881–905, 2024.

Andreas Kontogiannis, Konstantinos Papathanasiou, Yi Shen, Giorgos Stamou, Michael M Zavlanos, and George Vouros. Enhancing cooperative multi-agent reinforcement learning with state modelling and adversarial exploration. *arXiv preprint arXiv:2505.05262*, 2025.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 1179–1191, 2020.

Thomas Kuntz, Agatha Duzan, Hao Zhao, Francesco Croce, Zico Kolter, Nicolas Flammarion, and Maksym Andriushchenko. Os-harm: A benchmark for measuring safety of computer use agents. *arXiv preprint arXiv:2506.14866*, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Hanyu Lai, Xiao Liu, Yanxiao Zhao, Han Xu, Hanchen Zhang, Bohao Jing, Yanyu Ren, Shuntian Yao, Yuxiao Dong, and Jie Tang. Computerrl: Scaling end-to-end online reinforcement learning for computer use agents. *arXiv preprint arXiv:2508.14040*, 2025.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, et al. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Baixuan Li, Jialong Wu, Wenbiao Yin, Kuan Li, Zhongwang Zhang, Huifeng Yin, Zhengwei Tao, Liwen Zhang, Pengjun Xie, Jingren Zhou, et al. Nested browser-use learning for agentic information seeking. *arXiv preprint arXiv:2512.23647*, 2025a.

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

Mengdi Li, Jiaye Lin, Xufeng Zhao, Wenhao Lu, Peilin Zhao, Stefan Wermter, and Di Wang. Curriculum-rlaif: Curriculum alignment with reinforcement learning from ai feedback. *arXiv preprint arXiv:2505.20075*, 2025c.

Pengxiang Li, Zechen Hu, Zirui Shang, Jingrong Wu, Yang Liu, Hui Liu, Zhi Gao, Chenrui Shi, Bofei Zhang, Zihao Zhang, et al. Efficient multi-turn rl for gui agents via decoupled training and adaptive data curation. *arXiv preprint arXiv:2509.23866*, 2025d.

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

Xiangyi Li, Wenbo Chen, Yimin Liu, Shenghan Zheng, Xiaokun Chen, Yifeng He, Yubo Li, Bingran You, Haotian Shen, Jiankai Sun, et al. Skillsbench: Benchmarking how well agent skills work across diverse tasks. *arXiv preprint arXiv:2602.12670*, 2026.

Yang Li, Gang Li, Luheng He, Jingjie Zheng, Hong Li, and Zhiwei Guan. Widget captioning: Generating natural language description for mobile user interface elements. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Zhangheng Li, Keen You, Haotian Zhang, Di Feng, Harsh Agrawal, Xiujun Li, Mohana Prasad Sathya Moorthy, Jeff Nichols, Yinfei Yang, and Zhe Gan. Ferret-ui 2: Mastering universal user interface understanding across platforms. In *International Conference on Learning Representations (ICLR)*, 2024b.

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

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

M. Lin, M. Liu, T. Lu, L. Yuan, Y. Liu, H. Xu, Y. Miao, Y. Chao, and Z. Li. Gui-rewalk: Massive data generation for gui agent via stochastic exploration and intent-aware reasoning. *arXiv preprint arXiv:2509.15738*, 2025b.

Musen Lin, Minghao Liu, Taoran Lu, Lichen Yuan, Yiwei Liu, Haonan Xu, Yu Miao, Yuhao Chao, and Zhaojian Li. Gui-rewalk: Massive data generation for gui agent via stochastic exploration and intent-aware reasoning. *arXiv preprint arXiv:2509.15738*, 2025c.

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

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. In *International Conference on Learning Representations (ICLR)*, 2018b. URL `https://arxiv.org/abs/1802.08802`.

Guangyi Liu, Pengxiang Zhao, Yaozhen Liang, Liang Liu, Yaxuan Guo, Han Xiao, Weifeng Lin, Yuxiang Chai, Yue Han, Shuai Ren, et al. Llm-powered gui agents in phone automation: Surveying progress and prospects. *arXiv preprint arXiv:2504.19838*, 2025a.

Y. Liu, Z. Liu, S. Zhu, P. Li, C. Xie, J. Wang, X. Hu, X. Han, J. Yuan, X. Wang, S. Zhang, H. Yang, and F. Wu. Infigui-g1: Advancing gui grounding with adaptive exploration policy optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026. arXiv preprint arXiv:2508.05731.

Yuhang Liu, Pengxiang Li, Zishu Wei, Congkai Xie, Xueyu Hu, Xinchen Xu, Shengyu Zhang, Xiaotian Han, Hongxia Yang, and Fei Wu. Infiguiagent: A multimodal generalist gui agent with native reasoning and reflection. *arXiv preprint arXiv:2501.04575*, 2025b.

Yuhang Liu, Zeyu Liu, Shuanghe Zhu, Pengxiang Li, Congkai Xie, Jiasheng Wang, Xavier Hu, Xiaotian Han, Jianbo Yuan, Xinyao Wang, et al. Infigui-g1: Advancing gui grounding with adaptive exploration policy optimization. *arXiv preprint arXiv:2508.05731*, 2025c.

Fanbin Lu, Zhisheng Zhong, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Arpo: End-to-end policy optimization for gui agents with experience replay. *arXiv preprint arXiv:2505.16282*, 2025a.

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

Z. Lu, Y. Chai, Y. Guo, X. Yin, L. Liu, H. Wang, H. Xiao, S. Ren, G. Xiong, and H. Li. Ui-r1: Enhancing efficient action prediction of gui agents by reinforcement learning. *arXiv preprint arXiv:2503.21620*, 2025c.

Zhengxi Lu, Jiabo Ye, Fei Tang, Yongliang Shen, Haiyang Xu, Ziwei Zheng, Weiming Lu, Ming Yan, Fei Huang, Jun Xiao, et al. Ui-s1: Advancing gui automation via semi-online reinforcement learning. *arXiv preprint arXiv:2509.11543*, 2025d.

R. Luo, L. Wang, W. He, and X. Xia. Gui-r1: A generalist r1-style vision-language action model for gui agents. *arXiv preprint arXiv:2504.10458*, 2025a.

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

Tiange Luo, Lajanugen Logeswaran, Justin Johnson, and Honglak Lee. Visual test-time scaling for gui agent grounding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19989–19998, 2025c.

Zhiyu Mei, Wei Fu, Kaiwei Li, Guangju Wang, Huanchen Zhang, and Yi Wu. Real: Efficient rlhf training of large language models with parameter reallocation. *Proceedings of Machine Learning and Systems*, 7, 2025.

Leann J Mischel. Watch and learn? using edpuzzle to enhance the use of online videos. *Management Teaching Review*, 4(3):283–289, 2019.

Atsuyuki Miyai, Zaiying Zhao, Kazuki Egashira, Atsuki Sato, Tatsumi Sunada, Shota Onohara, Hiromasa Yamanishi, Mashiro Toyooka, Kunato Nishina, Ryoma Maeda, et al. Webchorearena: Evaluating web browsing agents on realistic tedious web tasks. *arXiv preprint arXiv:2506.01952*, 2025.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

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

Dang Nguyen, Jian Chen, Yu Wang, Gang Wu, Namyong Park, Zhengmian Hu, Hanjia Lyu, Junda Wu, Ryan Aponte, Yu Xia, et al. Gui agents: A survey. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 22522–22538, 2025.

Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, et al. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 6140–6150, 2025.

Runliang Niu, Jindong Li, Shiqi Wang, Yali Fu, Xiyu Hu, Xueyuan Leng, He Kong, Yi Chang, and Qi Wang. Screenagent: A vision language model-driven computer control agent. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2024.

OpenAI. GPT-4V(ision) technical work and authors — openai.com. `https://openai.com/contributions/gpt-4v/`, 2023. [Accessed 09-02-2026].

OpenAI. Computer-using agent — openai.com. `https://openai.com/index/computer-using-agent/`, 2025a. [Accessed 09-02-2026].

OpenAI. Computer-using agent, 2025b. URL `https://openai.com/index/computer-using-agent/`.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Vardaan Pahuja, Yadong Lu, Corby Rosset, Boyu Gou, Arindam Mitra, Spencer Whitehead, Yu Su, and Ahmed Hassan. Explorer: Scaling exploration-driven web trajectory synthesis for multimodal web agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 6300–6323, 2025.

Yichen Pan, Dehan Kong, Sida Zhou, Cheng Cui, Yifei Leng, Bing Jiang, Hangyu Liu, Yanyi Shang, Shuyan Zhou, Tongshuang Wu, et al. Webcanvas: Benchmarking web agents in online environments. *arXiv preprint arXiv:2406.12373*, 2024.

Cong Pang, Xuyu Feng, Yujie Yi, Zixuan Chen, Jiawei Hong, Tiankuo Yao, Nang Yuan, Jiapeng Luo, Lewei Lu, and Xin Lou. Ica: Information-aware credit assignment for visually grounded long-horizon information-seeking agents. *arXiv preprint arXiv:2602.10863*, 2026.

Panupong Pasupat, Tian-Shun Jiang, Evan Liu, Kelvin Guu, and Percy Liang. Mapping natural language commands to web elements. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 4970–4976, 2018.

Jiangweizhi Peng, Yuanxin Liu, Ruida Zhou, Charles Fleming, Zhaoran Wang, Alfredo Garcia, and Mingyi Hong. Hiper: Hierarchical reinforcement learning with explicit credit assignment for large language model agents. *arXiv preprint arXiv:2602.16165*, 2026.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent q: Advanced reasoning and learning for autonomous ai agents. *arXiv preprint arXiv:2408.07199*, 2024.

Z. Qi, X. Liu, I. L. Iong, H. Lai, X. Sun, X. Yang, J. Sun, Y. Yang, S. Yao, T. Zhang, W. Xu, J. Tang, and Y. Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2024.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Ram Ramrakhya, Andrew Szot, Omar Attia, Yuhao Yang, Anh Nguyen, Bogdan Mazoure, Zhe Gan, Harsh Agrawal, and Alexander Toshev. Scaling synthetic task generation for agents via exploration. *arXiv preprint arXiv:2509.25047*, 2025.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy Lillicrap. Androidinthewild: A large-scale dataset for android device control. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 59708–59728, 2023.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, et al. Androidworld: A dynamic benchmarking environment for autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Pascal J Sager, Benjamin Meyer, Peng Yan, Rebekka von Wartburg-Kottler, Layan Etaiwi, Aref Enayati, Gabriel Nobel, Ahmed Abdulkadir, Benjamin F Grewe, and Thilo Stadelmann. A comprehensive survey of agents for computer use: Foundations, challenges, and future directions. *arXiv preprint arXiv:2501.16150*, 2025.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *International Conference on Learning Representations (ICLR)*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Selenium Contributors. Selenium: Browser automation framework. `https://www.selenium.dev/`, 2023. Accessed: 2023.

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

Huawen Shen, Chang Liu, Gengluo Li, Xinlong Wang, Yu Zhou, Can Ma, and Xiangyang Ji. Falcon-ui: Understanding gui before following user instructions. *arXiv preprint arXiv:2412.09362*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Taiwei Shi, Sihao Chen, Bowen Jiang, Linxin Song, Longqi Yang, and Jieyu Zhao. Experiential reinforcement learning. *arXiv preprint arXiv:2602.13949*, 2026.

Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and Percy Liang. World of bits: An open-domain platform for web-based agents. In *International Conference on Machine Learning*, pp. 3135–3144. PMLR, 2017.

Yucheng Shi, Wenhao Yu, Zaitang Li, Yonglin Wang, Hongming Zhang, Ninghao Liu, Haitao Mi, and Dong Yu. Mobilegui-rl: Advancing mobile gui agent through reinforcement learning in online environment. *arXiv preprint arXiv:2507.05720*, 2025.

Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

C. H. Song, Y. Song, P. Goyal, Y. Su, O. Riva, H. Palangi, and T. Pfister. Watch and learn: Learning to use computers from online videos. *arXiv preprint arXiv:2510.04673*, 2025a.

Chan Hee Song, Yiwen Song, Palash Goyal, Yu Su, Oriana Riva, Hamid Palangi, and Tomas Pfister. Watch and learn: Learning to use computers from online videos. *arXiv preprint arXiv:2510.04673*, 2025b.

Linxin Song, Yutong Dai, Viraj Prabhu, Jieyu Zhang, Taiwei Shi, Li Li, Junnan Li, Silvio Savarese, Zeyuan Chen, Jieyu Zhao, et al. Coact-1: Computer-using agents with coding as actions. *arXiv preprint arXiv:2508.03923*, 2025c.

Yixiao Song, Katherine Thai, Chau Minh Pham, Yapei Chang, Mazin Nadaf, and Mohit Iyyer. Bearcubs: A benchmark for computer-using web agents. *arXiv preprint arXiv:2503.07919*, 2025d.

Yueqi Song, Frank F Xu, Shuyan Zhou, and Graham Neubig. Beyond browsing: Api-based web agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 11066–11085, 2025e.

Chuanneng Sun, Songjun Huang, and Dario Pompili. Llm-based multi-agent reinforcement learning: Current and future directions. *arXiv preprint arXiv:2405.11106*, 2024.

Libo Sun, Jiwen Zhang, Siyuan Wang, and Zhongyu Wei. Magnet: Towards adaptive gui agents with memory-driven knowledge evolution. *arXiv preprint arXiv:2601.19199*, 2026.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, et al. Os-genesis: Automating gui agent trajectory construction via reverse task synthesis. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5555–5579, 2025.

F. Tang, Z. Gu, Z. Lu, X. Liu, S. Shen, C. Meng, W. Wang, W. Zhang, Y. Shen, W. Lu, J. Xiao, and Y. Zhuang. Gui-g^2: Gaussian reward modeling for gui grounding. *arXiv preprint arXiv:2507.15846*, 2025a.

Fei Tang, Zhangxuan Gu, Zhengxi Lu, Xuyang Liu, Shuheng Shen, Changhua Meng, Wen Wang, Wenqi Zhang, Yongliang Shen, Weiming Lu, et al. Gui-g^2: Gaussian reward modeling for gui grounding. *arXiv preprint arXiv:2507.15846*, 2025b.

Fei Tang, Haolei Xu, Hang Zhang, Siqi Chen, Xingyu Wu, Yongliang Shen, Wenqi Zhang, Guiyang Hou, Zeqi Tan, Yuchen Yan, et al. A survey on (m) llm-based gui agents. *arXiv preprint arXiv:2504.13865*, 2025c.

Fei Tang, Zhiqiong Lu, Boxuan Zhang, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Clawgui: A unified framework for training, evaluating, and deploying gui agents. *arXiv preprint arXiv:2604.11784*, 2026.

Jiaqi Tang, Yu Xia, Yi-Feng Wu, Yuwei Hu, Yuhui Chen, Qing-Guo Chen, Xiaogang Xu, Xiangyu Wu, Hao Lu, Yanqing Ma, et al. Lpo: Towards accurate gui agent interaction via location preference optimization. *arXiv preprint arXiv:2506.09373*, 2025d.

Liujian Tang, Shaokang Dong, Yijia Huang, Minqi Xiang, Hongtao Ruan, Bin Wang, Shuo Li, Zhiheng Xi, Zhihui Cao, Hailiang Pang, et al. Magicgui: A foundational mobile gui agent with scalable data pipeline and reinforcement fine-tuning. *arXiv preprint arXiv:2508.03700*, 2025e.

Kimi Team, Angang Du, Bohong Yin, Bowei Xing, Bowen Qu, Bowen Wang, Cheng Chen, Chenlin Zhang, Chenzhuang Du, Chu Wei, et al. Kimi-vl technical report. *arXiv preprint arXiv:2504.07491*, 2025.

Shizuo Tian, Hao Wen, Yuxuan Chen, Jiacheng Liu, Shanhui Zhao, Guohong Liu, Ju Ren, Yunxin Liu, and Yuanchun Li. Agentprog: Empowering long-horizon gui agents with program-guided context management. *arXiv preprint arXiv:2512.10371*, 2025.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

Daniel Toyama, Philippe Hamel, Anita Gergely, Gheorghe Comanici, Amelia Glaese, Zafarali Ahmed, Tyler Jackson, Shibl Mourad, and Doina Precup. Androidenv: A reinforcement learning platform for android. *arXiv preprint arXiv:2105.13231*, 2021.

Wil MP Van der Aalst, Martin Bichler, and Armin Heinzl. Robotic process automation. *Business & information systems engineering*, 60(4):269–272, 2018.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Bryan Wang, Gang Li, Xin Zhou, Zhourong Chen, Tovi Grossman, and Yang Li. Screen2words: Automatic mobile ui summarization with multimodal learning. In *The 34th Annual ACM Symposium on User Interface Software and Technology*, pp. 498–510, 2021.

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

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

Shuai Wang, Weiwen Liu, Jingxuan Chen, Yuqi Zhou, Weinan Gan, Xingshan Zeng, Yuhan Che, Shuai Yu, Xinlong Hao, Kun Shao, et al. Gui agents with foundation models: A comprehensive survey. *arXiv preprint arXiv:2411.04890*, 2024b.

Shuhe Wang, Shengyu Zhang, Jie Zhang, Runyi Hu, Xiaoya Li, Tianwei Zhang, Jiwei Li, Fei Wu, Guoyin Wang, and Eduard Hovy. Reinforcement learning enhanced llms: A survey. *arXiv preprint arXiv:2412.10400*, 2024c.

Taiyi Wang, Zhihao Wu, Jianheng Liu, Jianye Hao, Jun Wang, and Kun Shao. Distrl: An asynchronous distributed reinforcement learning framework for on-device control agents. *arXiv preprint arXiv:2410.14803*, 2024d.

Taiyi Wang, Sian Gooding, Florian Hartmann, Oriana Riva, and Edward Grefenstette. A subgoal-driven framework for improving long-horizon llm agents. *arXiv preprint arXiv:2603.19685*, 2026.

Vivienne Huiling Wang, Tinghuai Wang, Wenyan Yang, Joni-Kristian Kämäräinen, and Joni Pajarinen. Probabilistic subgoal representations for hierarchical reinforcement learning. *arXiv preprint arXiv:2406.16707*, 2024e.

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

Xingyao Wang, Boxuan Li, Yufan Song, Frank F Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, et al. Openhands: An open platform for ai software developers as generalist agents. In *International Conference on Learning Representations (ICLR)*, 2024f.

Xinyuan Wang, Bowen Wang, Dunjie Lu, Junlin Yang, Tianbao Xie, Junli Wang, Jiaqi Deng, Xiaole Guo, Yiheng Xu, Chen Henry Wu, et al. Opencua: Open foundations for computer-use agents. *arXiv preprint arXiv:2508.09123*, 2025c.

Y. Wang, D. Yin, Y. Cui, R. Zheng, Z. Li, Z. Lin, D. Wu, X. Wu, C. Ye, Y. Zhou, and K.-W. Chang. Llms as scalable, general-purpose simulators for evolving digital agent training. *arXiv preprint arXiv:2510.14969*, 2025d.

Yiqin Wang, Haoji Zhang, Jingqi Tian, and Yansong Tang. Ponder & press: Advancing visual gui agent towards general computer control. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 1461–1473, 2025e.

Ziwei Wang, Leyang Yang, Xiaoxuan Tang, Sheng Zhou, Dajun Chen, Wei Jiang, and Yong Li. History-aware reasoning for gui agents. *arXiv preprint arXiv:2511.09127*, 2025f.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Xinming Wei, Jiahao Zhang, Haoran Li, Jiayu Chen, Rui Qu, Maoliang Li, Xiang Chen, and Guojie Luo. Agent. xpu: Efficient scheduling of agentic llm workloads on heterogeneous soc. *arXiv preprint arXiv:2506.24045*, 2025a.

Z. Wei, W. Yao, Y. Liu, W. Zhang, Q. Lu, L. Qiu, C. Yu, P. Xu, C. Zhang, B. Yin, H. Yun, and L. Li. Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025b.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, et al. Webwalker: Benchmarking llms in web traversal. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10290–10305, 2025a.

Jie Wu, Yu Gao, Zilyu Ye, Ming Li, Liang Li, Hanzhong Guo, Jie Liu, Zeyue Xue, Xiaoxia Hou, Wei Liu, et al. Rewarddance: Reward scaling in visual generation. *arXiv preprint arXiv:2509.08826*, 2025b.

Qianhui Wu, Kanzhi Cheng, Rui Yang, Chaoyun Zhang, Jianwei Yang, Huiqiang Jiang, Jian Mu, Baolin Peng, Bo Qiao, Reuben Tan, et al. Gui-actor: Coordinate-free visual grounding for gui agents. *arXiv preprint arXiv:2506.03143*, 2025c.

Qingyuan Wu, Jianheng Liu, Jianye Hao, Jun Wang, and Kun Shao. Vsc-rl: Advancing autonomous vision-language agents with variational subgoal-conditioned reinforcement learning. *arXiv e-prints*, pp. arXiv–2502, 2025d.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversations. In *First Conference on Language Modeling*, 2024a.

Qinzhuo Wu, Pengzhi Gao, Wei Liu, and Jian Luan. Backtrackagent: Enhancing gui agent with error detection and backtracking mechanism. *arXiv preprint arXiv:2505.20660*, 2025e.

Wenyi Wu, Kun Zhou, Ruoxin Yuan, Vivian Yu, Stephen Wang, Zhiting Hu, and Biwei Huang. Auto-scaling continuous memory for gui agent. *arXiv preprint arXiv:2510.09038*, 2025f.

Zhe Wu, Hongjin Lu, Junliang Xing, Changhao Zhang, Yin Zhu, Yuhao Yang, Yuheng Jing, Kai Li, Kun Shao, Jianye Hao, et al. Hi-agent: Hierarchical vision-language agents for mobile device control. *arXiv preprint arXiv:2510.14388*, 2025g.

Zhiyong Wu, Chengcheng Han, Zichen Ding, Zhenmin Weng, Zhoumianze Liu, Shunyu Yao, Tao Yu, and Lingpeng Kong. Os-copilot: Towards generalist computer agents with self-improvement. *arXiv preprint arXiv:2402.07456*, 2024b.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, et al. Os-atlas: A foundation action model for generalist gui agents. In *International Conference on Learning Representations (ICLR)*, 2024c.

Zhiheng Xi, Chenyang Liao, Guanyu Li, Yajie Yang, Wenxiang Chen, Zhihao Zhang, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, et al. Agentprm: Process reward models for llm agents via step-wise promise and progress. *arXiv preprint arXiv:2511.08325*, 2025.

Jiannan Xiang, Yun Zhu, Lei Shu, Maria Wang, Lijun Yu, Gabriel Barcik, James Lyon, Srinivas Sunkara, and Jindong Chen. Uisim: An interactive image-based ui simulator for dynamic mobile environments. *arXiv preprint arXiv:2509.21733*, 2025.

Zikai Xiao, Jianhong Tu, Chuhang Zou, Yuxin Zuo, Zhi Li, Peng Wang, Bowen Yu, Fei Huang, Junyang Lin, and Zuozhu Liu. Webworld: A large-scale world model for web agent training. *arXiv preprint arXiv:2602.14721*, 2026.

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

Frank F Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Z Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, et al. Theagentcompany: benchmarking llm agents on consequential real world tasks. *arXiv preprint arXiv:2412.14161*, 2024a.

Si Xu, Zixiao Huang, Yan Zeng, Shengen Yan, Xuefei Ning, Quanlu Zhang, Haolin Ye, Sipei Gu, Chunsheng Shui, Zhezheng Lin, Hao Zhang, Sheng Wang, Guohao Dai, and Yu Wang. Hethub: A distributed training system with heterogeneous cluster for large-scale models. *arXiv preprint arXiv:2405.16256*, 2024b.

Yifan Xu, Xiao Liu, Xinghan Liu, Jiaqi Fu, Hanchen Zhang, Bohao Jing, Shudan Zhang, Yuting Wang, Wenyi Zhao, and Yuxiao Dong. Mobilerl: Online agentic reinforcement learning for mobile gui agents. *arXiv preprint arXiv:2509.18119*, 2025a.

Yifei Xu, Tusher Chakraborty, Emre Kıcıman, Bibek Aryal, Eduardo Rodrigues, Srinagesh Sharma, Roberto Estevao, Maria Angels de Luis Balaguer, Jessica Wolk, Rafael Padilha, et al. Rlthf: Targeted human feedback for llm alignment. *arXiv preprint arXiv:2502.13417*, 2025b.

Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials. *arXiv preprint arXiv:2412.09605*, 2024c.

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

Haolong Yan, Yeqing Shen, Xin Huang, Jia Wang, Kaijun Tan, Zhixuan Liang, Hongxin Li, Zheng Ge, Osamu Yoshie, Si Li, Xiangyu Zhang, and Daxin Jiang. Gui exploration lab: Enhancing screen navigation in agents via multi-turn reinforcement learning, 2025a. URL `https://arxiv.org/abs/2512.02423`.

Haolong Yan, Jia Wang, Xin Huang, Yeqing Shen, Ziyang Meng, Zhimin Fan, Kaijun Tan, Jin Gao, Lieyu Shi, Mi Yang, Shiliang Yang, Zhirui Wang, Brian Li, Kang An, Chenyang Li, Lei Lei, Mengmeng Duan, Danxun Liang, Guodong Liu, Hang Cheng, Hao Wu, Jie Dong, Junhao Huang, Mei Chen, Renjie Yu, Shunshan Li, Xu Zhou, Yiting Dai, Yineng Deng, Yingdan Liang, Zelin Chen, Wen Sun, Chengxu Yan, Chunqin Xu, Dong Li, Fengqiong Xiao, Guanghao Fan, Guopeng Li, Guozhen Peng, Hongbing Li, Hang Li, Hongming Chen, Jingjing Xie, Jianyong Li, Jingyang Zhang, Jiaju Ren, Jiayu Yuan, Jianpeng Yin, Kai Cao, Liang Zhao, Liguo Tan, Liying Shi, Mengqiang Ren, Min Xu, Manjiao Liu, Mao Luo, Mingxin Wan, Na Wang, Nan Wu, Ning Wang, Peiyao Ma, Qingzhou Zhang, Qiao Wang, Qinlin Zeng, Qiong Gao, Qiongyao Li, Shangwu Zhong, Shuli Gao, Shaofan Liu, Shisi Gao, Shuang Luo, Xingbin Liu, Xiaojia Liu, Xiaojie Hou, Xin Liu, Xuanti Feng, Xuedan Cai, Xuan Wen, Xianwei Zhu, Xin Liang, Xin Liu, Xin Zhou, Yingxiu Zhao, Yukang Shi, Yunfang Xu, Yuqing Zeng, Yixun Zhang, Zejia Weng, Zhonghao Yan, Zhiguo Huang, Zhuoyu Wang, Zheng Ge, Jing Li, Yibo Zhu, Binxing Jiao, Xiangyu Zhang, and Daxin Jiang. Step-gui technical report, 2025b. URL `https://arxiv.org/abs/2512.15431`.

Haolong Yan, Jia Wang, Xin Huang, Yeqing Shen, Ziyang Meng, Zhimin Fan, Kaijun Tan, Jin Gao, Lieyu Shi, Mi Yang, et al. Step-gui technical report. *arXiv preprint arXiv:2512.15431*, 2025c.

Sikuan Yan, Xiufeng Yang, Zuchao Huang, Ercong Nie, Zifeng Ding, Zonggen Li, Xiaowen Ma, Kristian Kersting, Jeff Z Pan, Hinrich Schütze, et al. Memory-r1: Enhancing large language model agents to manage and utilize memories via reinforcement learning. *arXiv preprint arXiv:2508.19828*, 2025d.

Yunhe Yan, Shihe Wang, Jiajun Du, Yexuan Yang, Yuxuan Shan, Qichen Qiu, Xianqing Jia, Xinge Wang, Xin Yuan, Xu Han, et al. Mcpworld: A unified benchmarking testbed for api, gui, and hybrid computer use agents. *arXiv preprint arXiv:2506.07672*, 2025e.

Chenyu Yang, Shiqian Su, Shi Liu, Xuan Dong, Yue Yu, Weijie Su, Xuehui Wang, Zhaoyang Liu, Jinguo Zhu, Hao Li, et al. Zerogui: Automating online gui learning at zero human cost. *arXiv preprint arXiv:2505.23762*, 2025a.

John Yang, Akshara Prabhakar, Karthik Narasimhan, and Shunyu Yao. Intercode: Standardizing and benchmarking interactive coding with execution feedback. *Advances in Neural Information Processing Systems*, 36:23826–23854, 2023.

Qisong Yang, Thiago D Simão, Simon H Tindemans, and Matthijs TJ Spaan. Wcsac: Worst-case soft actor critic for safety-constrained reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 10639–10646, 2021.

Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe Huang, Amrita Saha, Zeyuan Chen, et al. Gta1: Gui test-time scaling agent. *arXiv preprint arXiv:2507.05791*, 2025b.

Yuhao Yang, Yue Wang, Dongxu Li, Ziyang Luo, Bei Chen, Chao Huang, and Junnan Li. Aria-ui: Visual grounding for gui instructions. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 22418–22433, 2025c.

Yuhao Yang, Zhen Yang, Zi-Yi Dou, Anh Nguyen, Keen You, Omar Attia, Andrew Szot, Michael Feng, Ram Ramrakhya, Alexander Toshev, et al. Ultracua: A foundation model for computer use agents with hybrid action. *arXiv preprint arXiv:2510.17790*, 2025d.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 20744–20757, 2022.

Jiabo Ye, Xi Zhang, Haiyang Xu, Haowei Liu, Junyang Wang, Zhaoqing Zhu, Ziwei Zheng, Feiyu Gao, Junjie Cao, Zhengxi Lu, et al. Mobile-agent-v3: Fundamental agents for gui automation. *arXiv preprint arXiv:2508.15144*, 2025.

Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, and Zhe Gan. Ferret-ui: Grounded mobile ui understanding with multimodal llms. In *European Conference on Computer Vision*, pp. 240–255. Springer, 2024.

Fei Yu, Anningzhe Gao, and Benyou Wang. Ovm, outcome-supervised value models for planning in mathematical reasoning. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pp. 858–875, 2024.

Qianhao Yuan, Jie Lou, Zichao Li, Jiawei Chen, Yaojie Lu, Hongyu Lin, Le Sun, Debing Zhang, and Xianpei Han. Memsearcher: Training llms to reason, search and manage memory via end-to-end reinforcement learning. *arXiv preprint arXiv:2511.02805*, 2025.

Xinbin Yuan, Jian Zhang, Kaixin Li, Zhuoxuan Cai, Lujian Yao, Jie Chen, Enguang Wang, Qibin Hou, Jinwei Chen, Peng-Tao Jiang, et al. Se-gui: Enhancing visual grounding for gui agents via self-evolutionary reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Zhixiong Zeng, Jing Huang, Liming Zheng, Wenkang Han, Yufeng Zhong, Lei Chen, Longrong Yang, Yingjie Chu, Yuzhi He, and Lin Ma. Uitron: Foundational gui agent with advanced perception and planning. *arXiv preprint arXiv:2508.21767*, 2025.

Baohe Zhang, Yuan Zhang, Lilli Frison, Thomas Brox, and Joschka Bödecker. Constrained reinforcement learning with smoothed log barrier function. *arXiv preprint arXiv:2403.14508*, 2024a.

Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue Liu, Qingwei Lin, et al. Large language model-brained gui agents: A survey. *arXiv preprint arXiv:2411.18279*, 2024b.

Chaoyun Zhang, Shilin He, Liqun Li, Si Qin, Yu Kang, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Api agents vs. gui agents: Divergence and convergence. *arXiv preprint arXiv:2503.11069*, 2025a.

Chaoyun Zhang, He Huang, Chiming Ni, Jian Mu, Si Qin, Shilin He, Lu Wang, Fangkai Yang, Pu Zhao, Chao Du, et al. Ufo2: The desktop agentos. *arXiv preprint arXiv:2504.14603*, 2025b.

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

Danyang Zhang, Lu Chen, and Kai Yu. Mobile-env: A universal platform for training and evaluation of mobile interaction. *CoRR*, 2023.

Danyang Zhang, Situo Zhang, Ziyue Yang, Zichen Zhu, Zihan Zhao, Ruisheng Cao, Lu Chen, and Kai Yu. Progrm: Build better gui agents with progress rewards. *arXiv preprint arXiv:2505.18121*, 2025d.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, et al. The landscape of agentic reinforcement learning for llms: A survey. *arXiv preprint arXiv:2509.02547*, 2025e.

Jiayuan Zhang, Kaiquan Chen, Zhihao Lu, Enshen Zhou, Qian Yu, and Jing Zhang. Prune4web: Dom tree pruning programming for web agent. *arXiv preprint arXiv:2511.21398*, 2025f.

Kaiyan Zhang, Yuxin Zuo, Bingxiang He, Youbang Sun, Runze Liu, Che Jiang, Yuchen Fan, Kai Tian, Guoli Jia, Pengfei Li, et al. A survey of reinforcement learning for large reasoning models. *arXiv preprint arXiv:2509.08827*, 2025g.

Le Zhang, Yixiong Xiao, Xinjiang Lu, Jingjia Cao, Yusai Zhao, Jingbo Zhou, Lang An, Zikan Feng, Wanxiang Sha, Yu Shi, et al. Omegause: Building a general-purpose gui agent for autonomous task execution. *arXiv preprint arXiv:2601.20380*, 2026.

Miaosen Zhang, Ziqiang Xu, Jialiang Zhu, Qi Dai, Kai Qiu, Yifan Yang, Chong Luo, Tianyi Chen, Justin Wagle, Tim Franklin, et al. Phi-ground tech report: Advancing perception in gui grounding. *arXiv preprint arXiv:2507.23779*, 2025h.

Shaojie Zhang, Ruoceng Zhang, Pei Fu, Shaokang Wang, Jiahui Yang, Xin Du, Shiqi Cui, Bin Qin, Ying Huang, Zhenbo Luo, et al. Btl-ui: Blink-think-link reasoning model for gui agent. *arXiv preprint arXiv:2509.15566*, 2025i.

Yanzhe Zhang, Tao Yu, and Diyi Yang. Attacking vision-language computer agents via pop-ups. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8387–8401, 2025j.

Zhong Zhang, Yaxi Lu, Yikun Fu, Yupeng Huo, Shenzhi Yang, Yesai Wu, Han Si, Xin Cong, Haotian Chen, Yankai Lin, et al. Agentcpm-gui: Building mobile-use agents with reinforcement fine-tuning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 155–180, 2025k.

Y. Zhao, H. Zhu, T. Jiang, S. Li, X. Xu, and H. H. Wang. Co-epg: A framework for co-evolution of planning and grounding in autonomous gui agents. *arXiv preprint arXiv:2511.10705*, 2025a.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. In *Proceedings of the VLDB Endowment*, 2023.

Yuan Zhao, Hualei Zhu, Tingyu Jiang, Shen Li, Xiaohang Xu, and Hao Henry Wang. Co-epg: A framework for co-evolution of planning and grounding in autonomous gui agents. *arXiv preprint arXiv:2511.10705*, 2025b.

Yuhao Zheng, Li'an Zhong, Yi Wang, Rui Dai, Kaikui Liu, Xiangxiang Chu, Linyuan Lv, Philip Torr, and Kevin Qinghong Lin. Code2world: A gui world model via renderable code generation. *arXiv preprint arXiv:2602.09856*, 2026.

Hanzhang Zhou, Xu Zhang, Panrong Tong, Jianan Zhang, Liangyu Chen, Quyu Kong, Chenglin Cai, Chen Liu, Yue Wang, Jingren Zhou, et al. Mai-ui technical report: Real-world centric foundation gui agents. *arXiv preprint arXiv:2512.22047*, 2025a.

Shijie Zhou, Viet Dac Lai, Hao Tan, Jihyung Kil, Wanrong Zhu, Changyou Chen, and Ruiyi Zhang. Gui-aima: Aligning intrinsic multimodal attention with a context anchor for gui grounding. *arXiv preprint arXiv:2511.00810*, 2025b.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations (ICLR)*, 2023.

Xunlan Zhou, Xuanlin Chen, Shaowei Zhang, Xiangkun Li, ShengHua Wan, Xiaohai Hu, Lei Yuan, Le Gan, and De-chuan Zhan. Marvl: Multi-stage guidance for robotic manipulation via vision-language models. *arXiv preprint arXiv:2602.15872*, 2026.

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. *arXiv preprint arXiv:2503.15478*, 2025c.