# NATURALGAIA: PUSHING THE FRONTIERS OF GUI AGENTS WITH A CHALLENGING BENCHMARK AND HIGH-QUALITY TRAJECTORY DATASET

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

The rapid advancement of Large Language Model (LLM)-driven Graphical User Interface (GUI) agents is significantly hampered by the profound limitations of existing evaluation benchmarks in terms of accuracy, reproducibility, and scalability. To address this critical gap, we introduce NaturalGAIA, a novel benchmark engineered on the principle of Causal Pathways. This design paradigm structures complex tasks into a series of programmatically verifiable atomic steps, ensuring a rigorous, fully automated, and reproducible standard for assessment. Concurrently, to mitigate the inherent capability deficits of agents, we developed LightManus, a hierarchical agent architecture specifically optimized for long-horizon tasks. We leveraged this agent to generate a high-quality, human-verified trajectory dataset that uniquely captures diverse and even self-correcting interaction patterns of LLMs. We then utilized this dataset to perform Reinforcement Fine-Tuning (RFT) on the Qwen2.5-VL-7B model. Our experiments reveal that NaturalGAIA presents a formidable challenge to current state-of-the-art LLMs; even the top-performing Claude-sonnet-4 achieved a Weighted Pathway Success Rate (WPSR) of only 34.6%. Moreover, while RFT substantially improved the smaller model's GUI execution capabilities (WPSR increased from 3.3% to 10.8%), its performance degraded sharply when handling complex scenarios. This outcome highlights the inherent capability ceiling of smaller models when faced with comprehensive tasks that integrate perception, decision-making, and execution. This research contributes a rigorous evaluation standard and a high-quality dataset to the community, aiming to guide the future development of GUI agents.

## 1 INTRODUCTION

The emergence of LLM-driven GUI agents (DeepSeek-AI et al., 2025; OpenAI et al., 2024b; Qwen et al., 2025) represents a significant advance over traditional automation (Yang et al., 2023; Zhang et al., 2023; Tang et al., 2025; Liu et al., 2025a; Durante et al., 2024), yet their progress is difficult to assess due to severe limitations in existing benchmarks. Current evaluation methods often rely on ambiguous final-state verification, which depends on fallible MLLMs (Bai et al., 2023; OpenAI et al., 2024a) and costly manual checks, thus compromising the scalability, reproducibility, and automation of evaluations (Mialon et al., 2023; Rawles et al., 2024). Moreover, these benchmarks inadequately measure performance on long-horizon tasks and fail to consider diverse valid solution paths, resulting in insufficient evaluation comprehensiveness (Liu et al., 2023; Li et al., 2025a; Chezelles et al., 2025).

Concurrently, these agents exhibit significant deficits in handling complex, real-world workflows, a problem exacerbated by prevailing learning paradigms (Koh et al., 2024b; Xu et al., 2024). Agents trained on successful execution traces Wang et al. (2025); Jiang et al. (2025); Qiu et al. (2025) tend to form rigid "shortcut" strategies. While effective for repeated tasks in static environments, this approach leads to poor generalization and robustness in novel scenarios, indicating a reliance on memory over transferable knowledge (Cheng et al., 2024; He et al., 2024; Gou et al., 2025). These generalization failures fundamentally stem from the LLM's deficiencies in GUI operational knowledge and long-range planning (Ye et al., 2025b; Muennighoff et al., 2025; Huang et al., 2024), a constraint rooted in the scarcity of high-quality, structured trajectory data He et al. (2025).

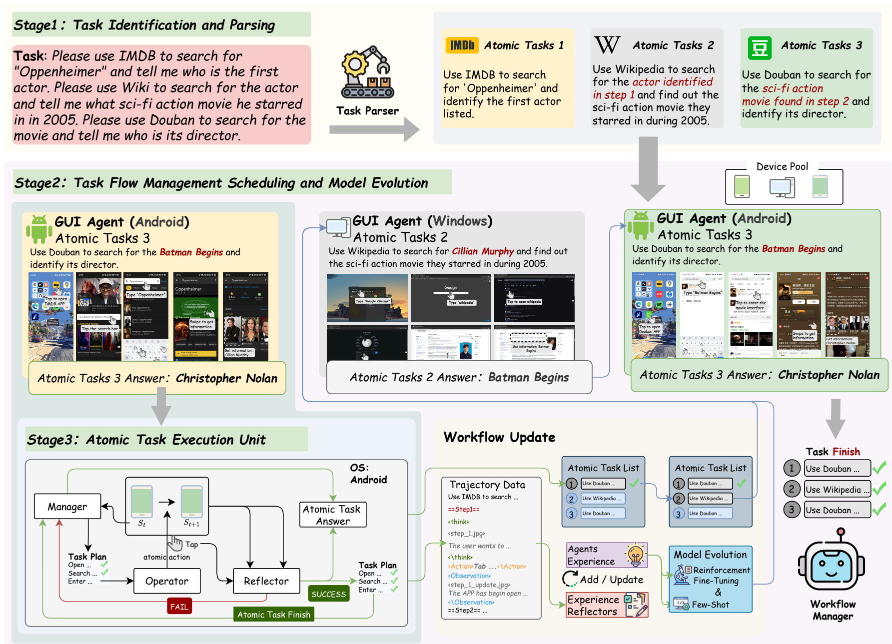



Figure 1: An overview of our three-stage framework **LightManus**, executing a complex, cross-platform task from NaturalGAIA. **Task Parsing** (Stage 1): A complex instruction is decomposed into a sequence of atomic tasks that form a Causal Pathway. **Workflow Management & Evolution** (Stage 2): The system schedules these tasks across different devices and uses collected Trajectory Data for model evolution via RFT. **Atomic Task Execution Unit** (Stage 3): This core unit, comprising a Manager, Operator, and Reflector, handles the execution of each atomic task.

To address these issues, we introduce NaturalGAIA, a novel benchmark with 276 tasks designed to address critical evaluation challenges in GUI agents. Unlike benchmarks such as GAIAMialon et al. (2023) that focus on flexible, goal-oriented reasoning, NaturalGAIA is engineered to rigorously test an agent's ability to execute complex, structured procedures reliably. This is achieved through Causal Pathways (CPs)—task structures that model realistic, high-frequency user workflows across multiple platforms, thereby simulating how humans sequentially use tools to achieve goals. Each pathway is composed of interdependent, programmatically verifiable atomic steps. An error at any step triggers a deterministic "Path Collapse", which not only provides an unambiguous failure signal but also pinpoints the source of the failure. Consequently, NaturalGAIA ensures task complexity and generalizability while significantly enhancing evaluation timeliness, accuracy, and automation, thereby establishing a rigorous and reproducible standard for assessing GUI agents.

Furthermore, advanced learning paradigms were explored to advance the capabilities of GUI agents. We developed a hierarchical agent architecture featuring a Task Parser and a Workflow Manager to improve long-horizon task handling through effective decomposition and dynamic scheduling. Additionally, we constructed a high-quality GUI operation trajectory dataset. Generated by our agent and iteratively refined through expert validation, this dataset ultimately contains 523 trajectories with detailed planning rationale. It was subsequently used to investigate the efficacy of Reinforcement Fine-Tuning (RFT) and serves as a foundational resource for developing next-generation agents. A representative example is presented in Figure 1.

Experimental results demonstrate that our hierarchical agent, LightManus (driven by Claude-sonnet-4), significantly outperforms mainstream baselines with a 42.9% Pass@1 success rate, yet achieves only a 34.6% Weighted Pathway Success Rate (WPSR) on our NaturalGAIA. We also validated the efficacy of our collected data through RFT, which substantially enhanced the Qwen2.5-VL-7B model's capability by increasing its WPSR from 3.3% to 10.8%. However, this enhancement does not generalize to complex, long-horizon tasks where performance degrades sharply. These findings reveal a critical gap between specialized, data-driven skill acquisition and the generalizable, long-range planning required for true GUI autonomy.

The main contributions of this paper can be summarized as follows:

- We introduce NaturalGAIA, a novel benchmark meticulously designed around Causal Pathways principles and equipped with a multi-level evaluation framework to precisely evaluate agent performance on complex, long-horizon tasks. Experimental results reveal that even state-of-the-art models encounter a formidable challenge on this suite.
- We construct and publicly release a high-quality dataset of GUI operation trajectories, comprising 523 trajectories curated through an iterative process of LLM generation and rigorous human verification, to provide a critical resource for future agent training.
- We conduct an in-depth investigation into trajectory-based reinforcement fine-tuning. This study offers initial evidence and key insights into leveraging this method efficiently to enhance agent capabilities.

## 2 Related Work

The advancement of GUI agents faces three principal challenges. First, an evaluation-realism dilemma persists in benchmarks. While environments have evolved towards higher fidelity, from web-based Zhou et al. (2023); Koh et al. (2024a) to mobile-centric Chai et al. (2025) and long-horizon tasks Ye et al. (2025a), this pursuit of realism compromises scientific rigor. The dynamic nature of modern applications impedes reproducibility Garg et al. (2025), and the existence of multiple valid solution paths renders fixed trajectory matching unreliable Zhou et al. (2023); Liu et al. (2023), often necessitating costly or fallible evaluations Chai et al. (2025); Mialon et al. (2023). Second, a data bottleneck constrains the development of increasingly sophisticated agents. Advances in high-resolution visual perception Hong et al. (2024); Li et al. (2025b); Cheng et al. (2024) and modular architectures Zhang et al. (2025; 2024); Wang et al. (2024) demand structured, fine-grained data that existing datasets lack. Third, a deficit of experiential data hinders the shift from simple imitation learning Rasheed et al. (2024); Deng et al. (2023) to more advanced paradigms like learning from self-exploration Wang et al. (2023); Jiang et al. (2025) or trial-and-error Shinn et al. (2023); Qiu et al. (2025); Wang et al. (2024), which require diverse trajectory data. This work addresses these gaps directly.

NaturalGAIA resolves the evaluation dilemma by providing a stable, reproducible, and fully automated benchmark with programmatically verifiable answers. Concurrently, our trajectory dataset addresses both the data bottleneck and the experiential data deficit by supplying structured and diverse interaction data to foster the development of more capable and robust agents.

## 3 NaturalGAIA

While prominent benchmarks such as GAIA excel at assessing flexible, goal-oriented reasoning, they lack rigorous testing of an agent's ability to reliably execute complex, structured procedures grounded in natural user tasks. To fill this gap, NaturalGAIA introduces Causal Pathways (CPs)—a task architecture of strict, causally-dependent sequences explicitly designed to model these authentic, cross-platform procedures. In this framework, any procedural error triggers a deterministic Path Collapse, yielding an unambiguous, process-level failure signal that transcends the limitations of final-state evaluations. NaturalGAIA thus functions as a precise diagnostic tool, designed to probe an agent's core capabilities in long-range planning, memory, and operational fidelity.

### 3.1 Atomic Tasks: The Fundamental Nodes of Causal Pathways

At the core of NaturalGAIA are *atomic tasks*—the smallest indivisible units of interaction. Each atomic task is designed to retrieve a single, specific piece of information (i.e., the "answer") through a sequence of primitive GUI operations. Importantly, every atomic task is associated with a unique and deterministic correct answer, allowing for objective, automated evaluation. This granularity enables the benchmark to assess an agent's performance in perception, interface manipulation, and reasoning independently. By isolating each task's input, action sequence, and output, NaturalGAIA facilitates precise error analysis and supports targeted improvements in sub-capabilities. These atomic tasks serve as the fundamental nodes that constitute each Causal Pathway.

### 3.2 Causal Pathways: Structuring Tasks with Logical Dependencies

The core construct of NaturalGAIA is the Causal Pathway (CP), an executable representation of procedural knowledge for structured tasks. A key design principle is the decoupling of a CP's abstract logic from its physical execution across diverse platforms, such as different applications or websites. This allows for the flexible generation of varied task instances to mitigate data contamination and model overfitting, while simultaneously upholding the rigorous evaluation of underlying causal dependencies.

Nodes within a CP are linked by strict causal dependencies, where an error at any step triggers a "Path Collapse"—a deterministic failure of the entire sequence. This mechanism provides an unambiguous signal, pinpointing the exact moment an agent's execution deviates from the required causal logic. Consequently, NaturalGAIAevaluates not just task completion but, more critically, an agent's capacity for long-range planning, abstract reasoning, and robust execution.

The specific construction principles and processes of CPs and tasks in NaturalGAIA are shown in Appendix D.1 and D.2 respectively.

### 3.3 Multi-level Evaluation Framework

To precisely evaluate agent performance within NaturalGAIA and meticulously identify the root causes of errors, this study constructs a hierarchical evaluation framework. This framework enables comprehensive scrutiny from macroscopic, difficulty-weighted task completion down to microscopic error attribution.

**Level 1: Weighted Pathway Success Rate (WPSR)** To account for task complexity, we introduce a difficulty score $D_{j,i}$ for each task instance $i$ of a Causal Pathway $\text{CP}_j$. This score is proportional to the number of nodes in the pathway and the number of distinct applications involved in its corresponding task. WPSR is defined by weighting each successful completion ($\mathbb{S}_{\textbf{task}}(j,i) = 1$) by its normalized difficulty $w_{j,i} = D_{j,i} / \sum_{k,l} D_{k,l}$, where the sum in the denominator is over all task instances across all pathways.

$$\text{WPSR} = \sum_{j,i} w_{j,i} \cdot \mathbb{S}_{\textbf{task}}(j,i)$$

WPSR serves as a holistic metric for an agent's final task completion capability, weighted by difficulty.

**Level 2: Fine-grained Traversal Metrics** To quantify partial progress and resilience, we introduce two complementary metrics that assess traversal quality.

**Mean Atomic Tasks Completion Ratio (MATCR):** This metric quantifies an agent's ability to successfully complete sequences of atomic tasks. For each task sequence $j$ within the benchmark, a completion ratio $R_j = k_j/n_j$ is calculated, where $k_j$ is the number of consecutive atomic tasks successfully executed from the start, and $n_j$ is the total number of atomic tasks in sequence $j$.

$$\text{MATCR} = \frac{1}{N} \sum_{j=1}^{N} \frac{k_j}{n_j}$$

**Positional-Weighted Atomic Tasks Success Rate (p-ATSR):** This metric evaluates an agent's ability to maintain long-term coherence by assigning greater weight to successes in the later stages of a Causal Pathway. Let $n_j$ be the total number of atomic tasks in pathway $\text{CP}_j$. We introduce a positional weight, denoted as $p(i)$, which is a monotonically increasing function of the step index $i$. The p-ATSR is then defined as:

$$\text{p-ATSR} = \frac{\sum_{j=1}^{N} \sum_{i=1}^{n_j} p(i) \cdot \mathbb{S}_{\text{atomic}}(j,i)}{\sum_{j=1}^{N} \sum_{i=1}^{n_j} p(i)}$$

where $\mathbb{S}_{\text{atomic}}(j, i) = 1$ for a success at step $i$ of pathway $j$, and 0 otherwise.

In summary, MATCR assesses an agent's foundational reliability by quantifying its average execution length, whereas p-ATSR places greater emphasis on its long-term coherence by rewarding success in the later stages of a task.

**Level 3: Error Attribution Analysis** For atomic tasks that fail, as identified by the Level 2 metrics, this stage involves analyzing the atomic operation sequence to attribute Execution Errors (EE) to one of three primary types:

- *Knowledge Deficit (KD)*: The agent lacks the necessary domain-specific or procedural knowledge regarding application operations or function invocation required to achieve the objective.
- *Perceptual Error (PE)*: Although the agent possesses the correct intent and planning, it fails during the information extraction phase due to misreading or misinterpretation of visual or textual information from the interface.
- *Operational Error (OE)*: The agent understands the task and correctly perceives the interface, but fails to take appropriate or precise actions during specific operations.

### 3.4 Task Difficulty Stratification

To enable a systematic evaluation of agent capabilities, tasks in NaturalGAIA are stratified into three difficulty levels. This stratification is defined by the structural properties of their underlying Causal Pathways, including their length (number of nodes), the complexity of transitions between nodes, and the diversity of applications involved.

**Level 1:** Consisting of short Causal Pathways involving 1-2 atomic tasks (nodes). These tasks typically operate within 1-2 applications and assess an agent's basic sequential execution and information-passing capabilities.

**Level 2:** Featuring Causal Pathways of intermediate length, containing 3-4 nodes. These pathways often require transitions across 3-5 different applications or involve complex interactions within a smaller set of applications, testing an agent's ability to maintain context over a longer duration.

**Level 3:** Involving complex and lengthy Causal Pathways spanning 5-7 nodes and 3-7 applications. Successfully traversing these pathways demands advanced planning, long-term memory, and robust handling of intricate inter-node dependencies, posing a significant challenge to the agent's cognitive and operational architecture.

Finally, we constructed 276 tasks. Data statistics and examples of tasks at different levels are shown in the Appendix D.3 and D.4.

## 4 Method

### 4.1 Cross-Platform Agent Architecture

To facilitate the generation of high-quality, long-horizon task trajectories, this study introduces LightManus, a specialized framework engineered for complex, cross-platform operations. Its hierarchical architecture is systematically organized into three distinct stages. The first stage, Task Parsing, analyzes the high-level user instruction and decomposes it into a logical sequence of executable sub-tasks. The second stage, the Workflow Manager, orchestrates this sequence by dynamically scheduling and delegating each sub-task to the appropriate device. The final stage is the Atomic Task Execution Unit, which executes the ground-level operations. This unit consists of specialized low-level agents—Mobile-Agent-e Wang et al. (2025) for Android and PC-Agent Liu et al. (2025b) for PC—that handle direct interaction with the graphical user interface. This structured, three-stage design provides a robust mechanism for generating high-fidelity execution trajectories and optimizes long-horizon task performance by ensuring seamless coordination across disparate operating systems.

### 4.2 HUMAN-VERIFIED TRAJECTORY COLLECTION

To construct a high-quality dataset aimed at enhancing the capabilities of GUI agents, we implemented a **rigorous** two-phase data collection pipeline. This methodology is designed to synergize the scalability and diverse exploratory power of a Large Language Model (LLM)-driven agent with the precision and reliability of human experts. The core motivation is to create a dataset that not only ensures procedural correctness but also captures a rich diversity of interaction behaviors, thereby transcending the traditional paradigm of singular, "optimal" paths.

**Phase 1: LLM-Driven Trajectory Generation** The data generation process was initiated using 265 unique atomic tasks as prompts. Our agent, LightManus, powered by Gemini-2.5-Pro, was tasked with autonomously completing these tasks. For each task, LightManus engaged in a **cyclical action process** at every step, which can be decomposed as follows:

- **Perception and State Understanding**: The agent first captures and analyzes the current user interface (UI) screenshot.
- **Deliberation and Planning**: The agent integrates the task objective with visual information to perform multi-level reasoning. This includes a high-level `Thought` process to maintain the overall plan and a micro-level `Action Thought` to determine the specific UI operation to be executed.
- **Action Execution**: The agent formulates and dispatches the determined action command.
- **Observation and Reflection**: Following the action, the agent captures and observes the new interface state, then provides feedback on the outcome to reflect on its progress toward the goal.

This automated procedure yielded an initial corpus of over 600 successfully executed trajectories. However, this raw corpus suffered from a critical flaw that necessitated curation: in certain instances, the model would bypass genuine UI interaction by directly generating answers from its internal, parametric knowledge. This highlighted the indispensable value of the human verification phase.

**Phase 2: Expert Verification and Curation** Every successfully generated trajectory was subjected to a meticulous review by a team of two graduate researchers. This review involved a detailed source-trace analysis of all execution information, including step-by-step screenshots and corresponding log files. Through an iterative refinement of this analysis, 523 trajectories were ultimately curated for the final dataset, adhering to the following principles:

- **Enforcing Grounded Execution**: A trajectory was deemed valid only if all of the agent's actions were strictly grounded in the visual information presented on-screen. This principle explicitly filters out any solutions derived from the model's internal knowledge, ensuring the data reflects genuine agent-environment interaction.
- **Preserving Behavioral Diversity and Corrective Processes**: Our curation criteria deliberately deviate from selecting only the "most efficient" paths, aiming instead to retain a variety of valid interaction strategies. Furthermore, we intentionally preserved trajectories containing recoverable *error steps*—cases where the agent successfully self-corrects through reflection after executing an erroneous action or exploring a wrong path. We posit that these trajectories, containing corrective processes, provide an invaluable learning signal for training more robust agents capable of reasoning about and recovering from their own errors.

### 4.3 FINAL DATA STRUCTURE

The curated dataset is composed of structured data points, each containing the initial instruction, the final solution, and the interaction trajectory. A complete trajectory $T$ is represented as a sequence of time steps, $T = (t_1, t_2, \ldots, t_n)$. Each time step $t_i$ is defined as a tuple $(s_i, R_i, a_i, s_{i+1}, o_i)$ that encapsulates a full state transition, where: $s_i$ is the initial state before the action, represented by a UI screenshot. $R_i$ is the agent's cognitive reasoning trace, encapsulating its high-level strategic `Thought` and micro-level `Action Thought`. $a_i$ is the specific `Action` executed by the agent in

| Method | Level-1 | | | | | Level-2 | | | | | Level-3 | | | | | Overall | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | SR | | WPSR | MATCR | pATSR | SR | | WPSR | MATCR | pATSR | SR | | WPSR | MATCR | pATSR | SR | | WPSR | MATCR | pATSR |
| | P@1 | P@4 | | | | P@1 | P@4 | | | | P@1 | P@4 | | | | P@1 | P@4 | | | |
| **PC-Agent** | | | | | | | | | | | | | | | | | | | | |
| • Gemini-2.5-Pro | 40.0 | 66.7 | 37.5 | 60.7 | 54.1 | 10.0 | 40.0 | 12.5 | 43.1 | 31.8 | 0.0 | 20.0 | 5.1 | 25.0 | 16.0 | 20.0 | 45.7 | 14.6 | 38.6 | 25.8 |
| **Mobile-Agent-e** | | | | | | | | | | | | | | | | | | | | |
| • Gemini-2.5-Pro | 46.7 | **100.0** | 57.1 | **75.0** | 70.0 | 10.0 | 60.0 | 20.0 | 52.5 | 41.9 | 0.0 | 30.0 | 9.3 | 24.6 | 17.2 | 22.9 | 68.6 | 23.2 | 44.5 | 31.6 |
| **LightManus** | | | | | | | | | | | | | | | | | | | | |
| • Gemini-2.5-Pro | **73.3** | **100.0** | 60.7 | 71.4 | **72.7** | 20.0 | **70.0** | 27.5 | 53.1 | 46.4 | 10.0 | **40.0** | 16.9 | 29.7 | 23.9 | 40.0 | **74.3** | 29.9 | 46.3 | 37.1 |
| • Gemini-2.5-Flash | 53.3 | 93.3 | **62.5** | 74.1 | 70.7 | 0.0 | 60.0 | 22.5 | 50.0 | 42.7 | 0.0 | 30.0 | 11.4 | 20.3 | 15.7 | 22.9 | 65.7 | 26.2 | 41.5 | 31.1 |
| • GPT-o3 | 60.0 | - | 57.1 | 67.9 | 64.5 | 30.0 | - | 30.0 | **57.5** | 49.6 | 0.0 | - | 0.0 | 32.2 | 20.8 | 34.3 | - | 22.0 | 48.0 | 35.1 |
| • Claude-sonnet-4 | 60.0 | - | 57.1 | 60.7 | 59.1 | **40.0** | - | **40.0** | **57.5** | **52.1** | **20.0** | - | **20.3** | **37.3** | **30.0** | **42.9** | - | **34.6** | **48.8** | **40.4** |
| • Qwen3 | 40.0 | 60.0 | 28.6 | 50.0 | 44.5 | 0.0 | 30.0 | 10.0 | 32.5 | 25.2 | 10.0 | 10.0 | 5.1 | 16.1 | 10.2 | 20.0 | 37.1 | 11.8 | 28.7 | 19.2 |
| • Qwen2.5-Max | 60.0 | 86.7 | 50.0 | 58.9 | 55.9 | 10.0 | 20.0 | 10.0 | 30.6 | 23.2 | 10.0 | 20.0 | 4.7 | 11.0 | 7.5 | 31.4 | 48.6 | 16.3 | 27.8 | 18.6 |
| • Qwen2.5-7B | 6.7 | 26.7 | 7.1 | 18.8 | 15.7 | 10.0 | 10.0 | 2.5 | 21.9 | 15.1 | 0.0 | 10.0 | 2.1 | 14.8 | 7.8 | 5.7 | 17.1 | 3.3 | 17.9 | 11.0 |
| • Qwen2.5-7B-RFT | 33.3 | 73.3 | 28.6 | 40.2 | 37.0 | 10.0 | 20.0 | 7.5 | 23.8 | 17.3 | 0.0 | 20.0 | 4.7 | 16.5 | 10.0 | 17.1 | 42.8 | 10.8 | 24.0 | 15.8 |

Table 1: Main results on the NaturalGAIA, comparing the performance of different agent frameworks and foundation models across varying task difficulty levels. We evaluate using Success Rate (SR%) at Pass@1 and Pass@4, Weighted Pathway Success Rate (WPSR%), Mean Atomic Tasks Completion Ratio (MATCR%), and Positional-Weighted Atomic Tasks Success Rate (p-ATSR%). Our proposed LightManus framework consistently outperforms baseline agents. While Reinforcement Fine-Tuning (RFT) remarkably improves the smaller model's performance, its effectiveness diminishes as task complexity increases.

that step. $s_{i+1}$ is the new state reached after executing action $a_i$, also represented by a UI screenshot. $o_i$ is the agent's `Observation` and reflection on the outcome of the state transition from $s_i$ to $s_{i+1}$.

This granular, structured format transparently captures the agent's complete perception-reasoning-action loop, making it an ideal resource for process-supervision and model fine-tuning. We present detailed trajectory data samples and corresponding visualization execution trajectories in the Appendix E.

### 4.4 Trajectory-based LLM Enhancement

The enhancement of the Qwen-2.5-VL-7B model was conducted via Trajectory-Reinforced Fine-Tuning (T-RFT), a methodological choice deliberately positioned between two common paradigms: Supervised Fine-Tuning (SFT) and end-to-end Reinforcement Learning (RL). While SFT offers a straightforward approach by mimicking expert demonstrations, it typically yields brittle policies that overfit to the specific, static trajectories seen during training. This reliance on behavioral cloning limits the agent's ability to generalize to novel situations or recover from errors. Conversely, conventional end-to-end RL, though theoretically more robust, is often practically infeasible for complex GUI tasks. It necessitates extensive online exploration, entailing prohibitive computational overhead, significant communication latency, and long training durations.

T-RFT presents a pragmatic and resource-efficient alternative. By leveraging a high-quality, static dataset of human-verified trajectories, it circumvents the need for costly online interaction. Yet, unlike SFT, it moves beyond simple imitation by applying policy optimization principles to learn a more generalizable decision-making model. This allows the agent to distill robust navigational strategies from the expert data, rather than merely memorizing action sequences. Ultimately, this approach was chosen to specifically augment the model's capacity to interpret and strategically act upon the sequential visual and textual inputs inherent to GUI environments.

## 5 Experiment

### 5.1 Setup

**Baselines.** The performance of our proposed LightManus framework was benchmarked against two baseline agents: `PC-Agent` (PCA) for desktop environments and `Mobile-Agent-e` (MAe) for Android environments.

**Foundation Models.** Our evaluation includes a diverse range of state-of-the-art proprietary and open-source MLLMs. Proprietary models include Google's Gemini series (Gemini-2.5-Pro, Gemini-2.5-Flash), OpenAI's GPT-o3, and Anthropic's Claude-sonnet-4. Open-source models assessed are from the Qwen series, including Qwen3-235B-A22B, Qwen2.5-Max, and the base Qwen2.5-VL-7B model which was targeted for fine-tuning.

**Training.** The enhancement of the smaller model was achieved through T-RFT. Specifically, the Qwen2.5-VL-7B model was fine-tuned using our curated dataset comprising 278 human-verified trajectories. The training was executed on a high-performance computing cluster equipped with four NVIDIA A100 GPUs. Further training details can be found in Appendix F.

### 5.2 MAIN RESULTS

The experimental results, detailed in Table 1, reveal several key findings. Our proposed LightManus framework consistently demonstrates superior performance over baseline agents across all evaluated metrics. Among the foundation models, large-scale proprietary models, particularly Claude-sonnet-4 and Gemini-2.5-Pro, exhibit the most robust capabilities, achieving the highest success rates on complex, long-horizon tasks. While open-source models show promise, a discernible performance gap remains. Furthermore, T-RFT substantially enhances the capabilities of smaller models on basic tasks, yet this advantage diminishes markedly with increasing task complexity, highlighting the intrinsic limitations of model scale. The subsequent analysis dissects these findings by evaluating the distinct contributions of the agent framework, the foundation models, and the fine-tuning process.

**Framework Efficacy: The Architectural Advantage of LightManus** The LightManus framework demonstrates a significant advantage in GUI execution tasks. Driven by the same Gemini-2.5-Pro model, LightManus's overall performance surpasses that of PCA and MAe. Specifically, its success rate (Pass@1) reaches 40.0%, which is considerably higher than PCA's 20.0% and MAe's 22.9%. Its WPSR is 29.9%, also outperforming PCA's 14.6% and MAe's 23.2%.

This superior performance stems from its higher execution reliability and long-horizon coherence. LightManus, powered by Gemini-2.5-Pro, shows substantial leads in both MATCR (46.3% vs. 39.2%/43.1%) and p-ATSR (37.1% vs. 25.8%/31.6%) metrics. A particularly compelling piece of evidence is that LightManus equipped with the smaller Gemini-2.5-Flash achieves a p-ATSR (31.1%) that is comparable to that of MAe equipped with the more powerful Gemini-2.5-Pro (31.6%). This result strongly indicates that the framework design of LightManus provides more robust execution support for complex, long-horizon tasks.

**The Impact of LLM Scale on Performance** The experimental results clearly reveal the critical impact of LLM scale on performance. Larger models, with their extensive knowledge bases, yield higher operational reliability. This is reflected in the MATCR metric: LightManus driven by Gemini-2.5-Pro (46.3%) outperforms its Gemini-2.5-Flash version (41.5%), indicating that larger models can execute atomic operations more stably.

However, a counter-intuitive phenomenon was observed in simple tasks: larger models occasionally exhibit a tendency to "over-plan", breaking down simple instructions into excessively verbose steps. For instance, in Level-1 tasks, the agent equipped with Gemini-2.5-Flash slightly outperformed in WPSR (62.5% vs. 60.7%) and MATCR (74.1% vs. 71.4%). This suggests that for low-complexity tasks, the more concise planning strategy of smaller models can sometimes be more efficient.

**Efficacy and Limitations of Small Model Specialization** Fine-tuning smaller models highlights both their potential and inherent limitations. By fine-tuning Qwen2.5-VL-7B with our trajectory data, the model's performance on basic tasks was significantly enhanced. In Level-1 tasks, its Pass@1 success rate increased nearly fivefold (6.7% $\rightarrow$ 33.3%), WPSR nearly quadrupled (7.1% $\rightarrow$ 28.6%), and MATCR also doubled (18.8% $\rightarrow$ 40.2%).

However, this specialized capability gained through fine-tuning does not generalize well to complex scenarios. As task complexity increases, the model's intrinsic capability ceiling becomes apparent. Its WPSR plummets from 28.6% at Level-1 to 7.5% at Level-2 and 4.7% at Level-3. This precipitous decline in performance indicates that core capabilities required for executing complex GUI tasks, such as long-range planning and memory, cannot be compensated for solely by fine-tuning on domain-specific data.

### 5.3 Generalization Across Application Scenarios

| Model | SR(%) |
|---|---|
| Qwen2.5-VL-7B | 26.3 |
| Qwen2.5-VL-7B-RFT | 36.8 |

Table 2: Generalization performance on the A3 benchmark Chai et al. (2025). The evaluation is conducted on applications unseen during training to assess the model's transferability.

As shown in table 2, to rigorously evaluate the model's ability to generalize to new environments, we conducted a supplementary experiment on the A3 benchmark Chai et al. (2025). For this test, we selected six task subsets involving applications that are entirely disjoint from those present in our training trajectory data. This setup ensures a stringent assessment of transferable skills rather than memorized procedures. The results show that the base Qwen2.5-VL-7B model achieved a success rate of 26.3%. After T-RFT, the model's performance on these novel applications rose to 36.8%, representing a substantial relative improvement of approximately 40%. This marked enhancement strongly suggests that the fine-tuning process did not lead to overfitting. Instead, it indicates the model has internalized more generalizable operational principles for GUI interaction, underscoring the value of our high-quality trajectory data in fostering robust and transferable agent capabilities.

### 5.4 Error analysis

As shown in Figure 2, we classify agent failures into three primary types: Planning & Reasoning Errors (PRE), which refer to when the agent formulates a logically flawed sequence of steps; Structural Compliance Errors (SCE), where the model fails to generate output in the required action format; and Execution Errors (EE), a category that encompasses the finer-grained Operational Errors (OE), Knowledge Deficits (KD), and Perceptual Errors (PE).

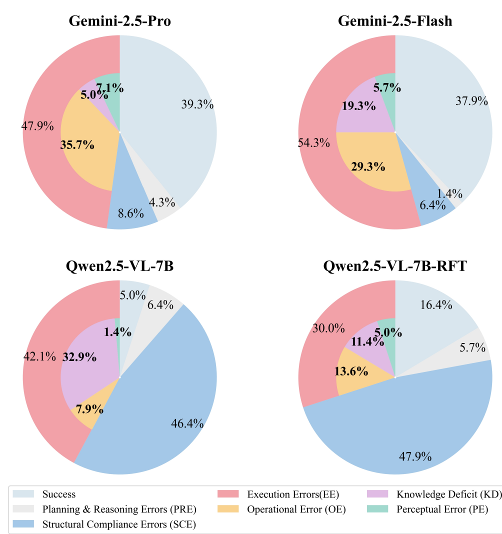



Figure 2: A comparative breakdown of failure modes for four vision-language models. The chart categorizes errors into three primary types: Planning & Reasoning Errors (PRE), Structural Compliance Errors (SCE), and Execution Errors (EE). The Execution Errors category is further subdivided into Operational Errors (OE), Knowledge Deficits (KD), and Perceptual Errors (PE).

The error analysis reveals significant disparities in failure modes among the models. High-performance models, such as the Gemini series (-2.5-Pro and -2.5-Flash), exhibits low PRE and SCE rates, demonstrating robust capabilities in planning and structural compliance. Their failures are predominantly concentrated in EE, accounting for 47.8% and 54.3% of their respective outcomes. A deeper analysis of the EE category shows that the primary bottleneck for Gemini-2.5-Pro is OE (35.7%), highlighting a challenge in execution accuracy. For the smaller Gemini-2.5-Flash, its EE composition is more heavily skewed towards KD (19.3% vs. 5.0%), reflecting a trade-off between model scale and embedded knowledge.

In contrast, the Qwen2.5-VL-7B series of models exhibits a more fundamental bottleneck. The root cause of failure for the base model is a critical rate of SCE (46.4%), followed by EE (42.2%). While T-RFT substantially reduces the total EE rate from 42.2% to 30.0%, this improvement is almost entirely attributable to the successful mitigation of KD (32.9% down to 11.4%). It is also noted that the SCE rate remains at a relatively high level of 47.9%. This indicates that while our fine-tuning method effectively imparts domain knowledge to improve execution, it has not yet fully resolved the inherent challenges in structural compliance found in certain smaller models.

## 6 Conclusion and Discussion

To address the critical challenges in evaluating GUI agents, we introduce NaturalGAIA, a novel benchmark based on "Causal Pathways", a hierarchical LightManus architecture, and a high-quality trajectory dataset featuring diverse and self-correcting behaviors. The experiments demonstrate that even state-of-the-art models like Claude-sonnet-4 face formidable challenges on NaturalGAIA, revealing deficiencies in long-horizon planning. Meanwhile, although Trajectory-Reinforced Fine-Tuning (T-RFT) enhances the foundational performance of smaller models (e.g., Qwen2.5-VL-7B), their capability ceiling in complex tasks remains constrained: while fine-tuning supplements some world knowledge, it struggles to fundamentally improve their operational precision and perceptual capabilities, and has not yet fully addressed the underlying challenge in structural compliance. This set of limitations highlights the critical gap between domain-specific skills and generalizable reasoning abilities.

While NaturalGAIA provides a robust standard, it does not fully capture the dynamic nature of real-world interfaces. Future research can proceed in three directions: (1) developing agents with enhanced structural and logical reasoning abilities to overcome their innate bottlenecks; (2) creating dynamic benchmarks capable of auto-generating novel "Causal Pathways" to mitigate the risk of data contamination; (3) exploring advanced learning algorithms that can efficiently learn from self-correcting behaviors to build more resilient agents.

## References

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond, 2023. URL https://arxiv.org/abs/2308.12966.

Yuxiang Chai, Hanhao Li, Jiayu Zhang, Liang Liu, Guangyi Liu, Guozhi Wang, Shuai Ren, Siyuan Huang, and Hongsheng Li. A3: Android agent arena for mobile gui agents, 2025. URL https://arxiv.org/abs/2501.01149.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents, 2024. URL https://arxiv.org/abs/2401.10935.

Thibault Le Sellier De Chezelles, Maxime Gasse, Alexandre Drouin, Massimo Caccia, Léo Boisvert, Megh Thakkar, Tom Marty, Rim Assouel, Sahar Omidi Shayegan, Lawrence Keunho Jang, Xing Han Lù, Ori Yoran, Dehan Kong, Frank F. Xu, Siva Reddy, Quentin Cappart, Graham Neubig, Ruslan Salakhutdinov, Nicolas Chapados, and Alexandre Lacoste. The browsergym ecosystem for web agent research, 2025. URL https://arxiv.org/abs/2412.05467.

DeepSeek-AI et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web, 2023. URL https://arxiv.org/abs/2306.06070.

Zane Durante, Qiuyuan Huang, Naoki Wake, Ran Gong, Jae Sung Park, Bidipta Sarkar, Rohan Taori, Yusuke Noda, Demetri Terzopoulos, Yejin Choi, Katsushi Ikeuchi, Hoi Vo, Li Fei-Fei, and Jianfeng Gao. Agent ai: Surveying the horizons of multimodal interaction, 2024. URL https://arxiv.org/abs/2401.03568.

Divyansh Garg, Shaun VanWeelden, Diego Caples, Andis Draguns, Nikil Ravi, Pranav Putta, Naman Garg, Tomas Abraham, Michael Lara, Federico Lopez, James Liu, Atharva Gundawar, Prannay Hebbar, Youngchul Joo, Jindong Gu, Charles London, Christian Schroeder de Witt, and Sumeet Motwani. Real: Benchmarking autonomous agents on deterministic simulations of real websites, 2025. URL https://arxiv.org/abs/2504.11543.

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, 2025. URL https://arxiv.org/abs/2410.05243.

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, 2024. URL https://arxiv.org/abs/2401.13919.

Yanheng He, Jiahe Jin, and Pengfei Liu. Efficient agent training for computer use, 2025. URL https://arxiv.org/abs/2505.13909.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents, 2024. URL https://arxiv.org/abs/2312.08914.

Zhen Huang, Haoyang Zou, Xuefeng Li, Yixiu Liu, Yuxiang Zheng, Ethan Chern, Shijie Xia, Yiwei Qin, Weizhe Yuan, and Pengfei Liu. O1 replication journey – part 2: Surpassing o1-preview through simple distillation, big progress or bitter lesson?, 2024. URL https://arxiv.org/abs/2411.16489.

Wenjia Jiang, Yangyang Zhuang, Chenxi Song, Xu Yang, and Chi Zhang. Appagentx: Evolving gui agents as proficient smartphone users, 2025. URL https://arxiv.org/abs/2503.02268.

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 Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 881–905, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.50. URL https://aclanthology.org/2024.acl-long.50/.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks, 2024b. URL https://arxiv.org/abs/2401.13649.

Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. Screenspot-pro: Gui grounding for professional high-resolution computer use, 2025a. URL https://arxiv.org/abs/2504.07981.

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, 2025b. URL https://arxiv.org/abs/2410.18967.

Guangyi Liu, Pengxiang Zhao, Liang Liu, Yaxuan Guo, Han Xiao, Weifeng Lin, Yuxiang Chai, Yue Han, Shuai Ren, Hao Wang, Xiaoyu Liang, Wenhao Wang, Tianze Wu, Linghao Li, Hao Wang, Guanjing Xiong, Yong Liu, and Hongsheng Li. Llm-powered gui agents in phone automation: Surveying progress and prospects, 2025a. URL https://arxiv.org/abs/2504.19838.

Haowei Liu, Xi Zhang, Haiyang Xu, Yuyang Wanyan, Junyang Wang, Ming Yan, Ji Zhang, Chunfeng Yuan, Changsheng Xu, Weiming Hu, and Fei Huang. Pc-agent: A hierarchical multi-agent collaboration framework for complex task automation on pc, 2025b. URL https://arxiv.org/abs/2502.14282.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents, 2023. URL https://arxiv.org/abs/2308.03688.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants, 2023. URL https://arxiv.org/abs/2311.12983.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling, 2025. URL https://arxiv.org/abs/2501.19393.

OpenAI et al. Gpt-4 technical report, 2024a. URL https://arxiv.org/abs/2303.08774.

OpenAI et al. Openai o1 system card, 2024b. URL https://arxiv.org/abs/2412.16720.

Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, Xing Zhou, Dongrui Liu, Ling Yang, Yue Wu, Kaixuan Huang, Shilong Liu, Hongru Wang, and Mengdi Wang. Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution, 2025. URL https://arxiv.org/abs/2505.20286.

Qwen et al. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Hanoona Rasheed, Muhammad Maaz, Sahal Shaji Mullappilly, Abdelrahman Shaker, Salman Khan, Hisham Cholakkal, Rao M. Anwer, Erix Xing, Ming-Hsuan Yang, and Fahad S. Khan. Glamm: Pixel grounding large multimodal model, 2024. URL https://arxiv.org/abs/2311.03356.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. Androidworld: A dynamic benchmarking environment for autonomous agents, 2024.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. URL https://arxiv.org/abs/2303.11366.

Fei Tang, Haolei Xu, Hang Zhang, Siqi Chen, Xingyu Wu, Yongliang Shen, Wenqi Zhang, Guiyang Hou, Zeqi Tan, Yuchen Yan, Kaitao Song, Jian Shao, Weiming Lu, Jun Xiao, and Yueting Zhuang. A survey on (m)llm-based gui agents, 2025. URL https://arxiv.org/abs/2504.13865.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration, 2024. URL https://arxiv.org/abs/2406.01014.

Zhenhailong Wang, Haiyang Xu, Junyang Wang, Xi Zhang, Ming Yan, Ji Zhang, Fei Huang, and Heng Ji. Mobile-agent-e: Self-evolving mobile assistant for complex tasks. *arXiv preprint arXiv:2501.11733*, 2025.

Yifan Xu, Xiao Liu, Xueqiao Sun, Siyi Cheng, Hao Yu, Hanyu Lai, Shudan Zhang, Dan Zhang, Jie Tang, and Yuxiao Dong. Androidlab: Training and systematic benchmarking of android autonomous agents, 2024. URL https://arxiv.org/abs/2410.24024.

Sherry Yang, Ofir Nachum, Yilun Du, Jason Wei, Pieter Abbeel, and Dale Schuurmans. Foundation models for decision making: Problems, methods, and opportunities, 2023. URL https://arxiv.org/abs/2303.04129.

Suyu Ye, Haojun Shi, Darren Shih, Hyokun Yun, Tanya Roosta, and Tianmin Shu. Realwebassist: A benchmark for long-horizon web assistance with real-world users, 2025a. URL https://arxiv.org/abs/2504.10445.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. Limo: Less is more for reasoning, 2025b. URL https://arxiv.org/abs/2502.03387.

Chaoyun Zhang, He Huang, Chiming Ni, Jian Mu, Si Qin, Shilin He, Lu Wang, Fangkai Yang, Pu Zhao, Chao Du, Liqun Li, Yu Kang, Zhao Jiang, Suzhen Zheng, Rujia Wang, Jiaxu Qian, Minghua Ma, Jian-Guang Lou, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Ufo2: The desktop agentos, 2025. URL `https://arxiv.org/abs/2504.14603`.

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. Agent-pro: Learning to evolve via policy-level reflection and optimization, 2024. URL `https://arxiv.org/abs/2402.17574`.

Xinlu Zhang, Yujie Lu, Weizhi Wang, An Yan, Jun Yan, Lianke Qin, Heng Wang, Xifeng Yan, William Yang Wang, and Linda Ruth Petzold. Gpt-4v(ision) as a generalist evaluator for vision-language tasks, 2023. URL `https://arxiv.org/abs/2311.01361`.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023. URL `https://webarena.dev`.

# A Ethics Statement

This work adheres to the ICLR Code of Ethics. In this study, no human subjects or animal experimentation was involved. All datasets used, were sourced in compliance with relevant usage guidelines, ensuring no violation of privacy. We have taken care to avoid any biases or discriminatory outcomes in our research process. No personally identifiable information was used, and no experiments were conducted that could raise privacy or security concerns. We are committed to maintaining transparency and integrity throughout the research process.

# B Reproducibility Statement

We have made every effort to ensure that the results presented in this paper are reproducible. All code and datasets have been made publicly available in an anonymous repository to facilitate replication and verification. The experimental setup, including training steps, model configurations, and hardware details, is described in detail in the paper.

We believe these measures will enable other researchers to reproduce our work and further advance the field.

# C LLM Usage

Large Language Models (LLMs) were used to aid in the writing and polishing of the manuscript. Specifically, we used an LLM to assist in refining the language, improving readability, and ensuring clarity in various sections of the paper. The model helped with tasks such as sentence rephrasing, grammar checking, and enhancing the overall flow of the text.

It is important to note that the LLM was not involved in the ideation, research methodology, or experimental design. All research concepts, ideas, and analyses were developed and conducted by the authors. The contributions of the LLM were solely focused on improving the linguistic quality of the paper, with no involvement in the scientific content or data analysis.

The authors take full responsibility for the content of the manuscript, including any text generated or polished by the LLM. We have ensured that the LLM-generated text adheres to ethical guidelines and does not contribute to plagiarism or scientific misconduct.

# D Details of NaturalGAIA

## D.1 Design principles of NaturalGAIA (Level-3)

Taking the Level-3 build as an example, the design guidelines we provide to Task builders are as follows:

**Part I: Task Structure and Tool Application**

- **Number and Identification of Atomic Tasks:** Each main task must be designed to contain six to seven clearly distinguishable atomic tasks. Each atomic task should be clearly labeled and distinguished in the task description document or related schematic diagram.
- **Multi-tool Collaboration Requirements:** Ensure that at least three different mobile applications (APPs) are called to complete each main task. In terms of application selection, applications on the Android platform should be selected.

**Part II: Core Characteristics of Atomic Task Results**

- **Textual Presentation of Results**: The final results of the constructed atomic tasks must be clearly presented in text form to support subsequent reliability assessments.
- **Uniqueness and Certainty**: The output results generated after the execution of each atomic task must be unique and clearly defined, and can be verified objectively and unambiguously, and there should not be multiple possible correct answers.
- **Exclusion of Timeliness Effects**: The results of atomic tasks should be designed to be time-invariant, aiming to avoid changes, failures, or deviations from expected standards due to the passage of time.
- **Processing procedures for time-sensitive information**: If the task scenario inevitably involves critical information with timeliness (for example, real-time stock prices, limited-time promotion information, dynamically updated news lists, etc.), the information must be solidified the moment it is captured by the atomic task. This processing requires that the information content be fully recorded in a local static file (e.g.: screenshots saved as PNG, text information saved as TXT/PDF, table data saved as XLSX, complex documents saved as DOCX, etc.). At the same time, the atomic task must provide specific time constraints (e.g.: query the price of Apple16 on Apple's official website on April 11, 2025). The local static file constitutes the deterministic and verifiable output result of this atomic task.

**Part III: Logical dependencies between atomic tasks**

- **Mandatory information flow dependencies**: Strict sequential dependencies must be established between the designed atomic tasks. Specifically, the output result z of the previous atomic task must be used as the core input information required for the execution of the next atomic task.
- **Failed step-by-step interruption mechanism**: If the previous atomic task fails to successfully obtain its expected correct result, all subsequent atomic tasks that rely on this result should not be able to continue or be completed correctly. The task process must be designed based on this standard.

**Part IV: Clarification of application tools and ground truth**

- **Application of atomic tasks**: In the detailed description of each atomic task, the name of the specific mobile APP expected to be used to complete the task must be clearly specified.
- **Detailed description of the path to obtain the ground truth**: The specific operation path or method used by each atomic task constructed to obtain its ground truth must be described in detail to facilitate verification and confirmation by the verifier.

**Part V: Selection criteria for applications**

- **Commonality priority principle**: When selecting applications for tasks, be sure to give priority to common applications with high user penetration and widespread use in the app store[12] (as of April 2025, ranked within the top 50).

Table 3: An example of a task definition from Stage 1. The table specifies the high-level task, its difficulty level, the step-by-step Causal Pathway for human execution, the corresponding ground truth answers, and the tools involved.

| | |
|---|---|
| **Task** | Search for Leonardo Dicaprio on Douban and tell me what movie the actor is starring in in 2023? Use IMDB to look up the movie's profile and tell me what era the story takes place in? |
| **Level** | 1 |
| **Steps(human)** | 1. Open the Douban app.<br>2. Type "Leonardo Dicaprio" in the search box and click Search.<br>3. In the search results, click on Leonardo Dicaprio's actor entry to enter his personal homepage.<br>4. View his film and television works list and find the works released in"2023".<br>5. Record the movie name of the 2023 work. (Ground Truth 1: **Killers of the Flower Moon**)<br>6. Open the IMDB app.<br>7. Type the movie name recorded from the Douban app (Killers of the Flower Moon) in the search box and click Search.<br>8. Click the corresponding movie entry in the search results to enter the movie details page.<br>9. Scroll down to the movie profile section.<br>10. Read, find and record the era information mentioned in the profile (Ground Truth 2: **1920s**). |
| **Ground Truth** | 1. Killers of the Flower Moon<br>2. 1920s |
| **Tools** | Douban app<br>IMDB APP |
| **Number of Atomic Tasks** | 2 |

### D.2 Details of task creation

To ensure the quality, feasibility, and determinism of each task, a rigorous three-stage process was designed and implemented for the construction of NaturalGAIA, executed by a team of four annotators. This multi-stage protocol ensures the robustness of the tasks and the reliability of their outcomes. The process is detailed as follows:

**Stage 1: Task Design and Causal Pathway Definition.** This initial phase was led by Annotator 1, who designed the Causal Pathway and its constituent atomic tasks. The design was governed by core principles mandating that the output (i.e., the answer) for each task node be unique, deterministic, and temporally stable. This foundation ensures the long-term validity of the benchmark. Annotator 1 also developed reference procedures to confirm the pathway's initial feasibility on the Android platform. An example of a complete task definition produced in this stage, detailing the task, its constituent steps (the Causal Pathway), and ground truth, is presented in Table 3.

**Stage 2: Independent Verification and Cross-Platform Screening.** Annotators 2 and 3 served as independent validators. They first followed the reference procedures to traverse the designed Causal Pathway on Android. Subsequently, they verified the pathway's applicability on PC (Windows) platforms. This stage rigorously evaluated the practical feasibility and output determinism of the tasks, ensuring cross-platform reproducibility. The verification results for the example task, confirming answer consistency across Android and Windows platforms, are shown in Table 4.

[1] https://play.google.com/store.
[2] https://m.app.mi.com/.

Table 4: Cross-platform verification for the example task, corresponding to Stage 2. This table demonstrates the task's reproducibility by confirming that identical ground truth answers are obtained on both Android and Windows platforms.

| **Operating System** | | **Android** | **Windows** |
|---|---|---|---|
| **Task** | | Search for Leonardo Dicaprio on Douban and tell me what movie the actor is starring in in 2023? Use IMDB to look up the movie's profile and tell me what era the story takes place in? | |
| **Level** | | 1 | |
| **Ground Truth** | | 1. Killers of the Flower Moon<br>2. 1920s | |
| **Tools** | | Douban app<br>IMDB APP | https://www.douban.com/<br>https://www.imdb.com/ |
| **Answer Matching** | **Atomic Task 1** | ✓ | ✓ |
| | **Atomic Task 2** | ✓ | ✓ |

**Stage 3: Blind Feasibility Assessment and Human Baseline Establishment.** This phase was conducted by Annotator 4 to assess task clarity and establish a human performance baseline. To ensure an unbiased evaluation, Annotator 4 was provided only with the high-level task description, without access to the reference solution or the intended Causal Pathway. This setup simulates the process of an agent discovering and navigating the pathway based solely on instructions, thereby providing a crucial human baseline for evaluating agent autonomy.

### D.3 Details of Data Statistics

Table 5 presents the detailed statistical characteristics of the NaturalGAIA benchmark at different difficulty levels. The table lists the total number of entries (Num) in each difficulty interval from level 1 to level 5 and the Overall case. For each difficulty level, we further quantified the average number (Avg) and distribution Range of Tools used by annotators and Atomic tasks)performed during the benchmark construction process. In NaturalGAIA, there is a positive correlation between the difficulty level of the task and the average number of tools and atomic tasks required, which indicates that tasks with higher difficulty usually correspond to more complex processing flows and operation sequences.

Table 5: Statistics of NaturalGAIA

| **Difficulty level** | **Num** | **Tools** | | **Atomic tasks** | |
|---|---|---|---|---|---|
| | | Avg | Range | Avg | Range |
| 1 | 133 | 1.5 | 1~2 | 1.9 | 1~2 |
| 2 | 92 | 2.3 | 2~4 | 3.7 | 3~4 |
| 3 | 51 | 4.4 | 3~7 | 5.7 | 3~7 |
| Overall | 276 | 2.6 | 1~7 | 3.1 | 1~7 |

### D.4 Task example of NaturalGAIA

Figure 3 presents concrete examples of tasks corresponding to the three difficulty levels. These examples illustrate the increasing requirements in terms of the number of atomic tasks, the diversity of applications involved, and the complexity of planning and execution across the levels.

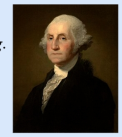
**Level-1**

**Question**: Search the movie "Everything Everywhere All at Once" on Douban to see who the Asian director is. Search IMDb to find out what movie he directed in 2016.

**Ground truth**: 1. Daniel Kwan; 2. Swiss Army Man

**Level-2**

**Question**: Use the file management app to open */Benchmark/Pic/1776.png*. In the jpg picture with the same name as the year, tell me who is in the picture? Which city in the United States is named after this person? Use Wikipedia to check the official name of this city. Use Google Map to view this city. Tell me which months are its peak tourist seasons? Use Google Map to check what the average temperature of this city is in these months?



*1776.png*

**Ground truth**: 1. George Washington; 2. Washington, D.C.; 3. July-October; 4. [July [24-31°C], August [23-30°C], September [19-27°C], October [13-21°C]]

**Level-3**

**Question**: Use Video Player to open /Benchmark/Video/song.avi and tell me which singer sings the song in the video? Use Apple Music to search which album this singer released in 2022? Which song in this album is named after a cocktail? Use Google APP to search the country where the main filming location of this song is located? Use Google APP to see what the capital of this country is? Use Wiki APP to search in which year the old city of this city was listed as a World Heritage Site by UNESCO?

*song.avi*

**Ground truth**: 1. Jay Chou; 2. The Greatest Show; 3. Mojito; 4. Cuba; 5. Havana; 6. 1982;

Figure 3: Task examples of different difficulty levels in NaturalGAIA

# E DETAILS OF EXECUTION TRAJECTORY

Each data entry in our collection is a JSON object designed to provide a comprehensive and transparent record of an agent's problem-solving process for a given task. The structure is organized with the following top-level keys:

- **`messages`**: This key holds an array that documents the agent's interaction flow. It begins with the initial, high-level task instruction (e.g., "Using Wikipedia, search for Jay Chou and tell me which album he released in 2000?"), followed by the agent's detailed, step-by-step execution trace.
- **`solution`**: This key contains a string with the ground-truth answer for the task, encapsulated within an <answer> tag (e.g., <answer>Jay</answer>). This allows for direct and automated evaluation of the task outcome.
- **`images`**: This key holds an array of strings, where each string is a file path to a screenshot captured during the interaction. These images provide the visual context for each step of the agent's trajectory.

The core of our dataset lies within the **execution trace**, which documents the agent's perception-reasoning-action loop for each step. A single step is structured as follows:

- **`think` Block**: This block encapsulates the agent's cognitive process before taking an action. It is crucial for understanding the agent's intent and planning. It contains:
  - **`<image>` Placeholder**: Marks where the corresponding pre-action screenshot is presented to the model.
  - **Thought**: A paragraph of text detailing the agent's high-level strategic thinking. It assesses the overall progress, recalls the main objective, and formulates a sub-goal for the current step.
  - **Action Thought**: A paragraph focused on low-level, tactical reasoning. Grounded in the visual information from the `<image>`, it identifies the specific UI element to interact with (e.g., a button or text field) and determines the **type of action** to perform on it (e.g., tap or type text).

- **`Action` Block**: This block contains a concise, human-readable natural language description of the action the agent decided to execute (e.g., "Tap on 'Search Wikipedia' to activate the keyboard.").
- **`Observation` Block**: This block documents the agent's reflection after the action has been executed. It includes:
  - **`<image>` Placeholder**: Marks where the post-action screenshot is presented, showing the result of the action.
  - **Observation**: A textual summary where the agent confirms the outcome of its action and updates its understanding of the task status, setting the stage for the next step.

This detailed, multi-faceted structure is explicitly designed to provide rich, process-level supervision signals, enabling models to learn not just what to do, but also why and how to do it based on visual evidence and logical reasoning. The trajectory data sample and visualization process are shown in the listings 1 to 3 and figs. 4 to 6 respectively.


```
{
    "images": [
        "image_17-44-23_1_1.jpg",
        "image_17-44-23_1_2.jpg",
        "image_17-44-23_1_3.jpg",
        "image_17-44-23_1_4.jpg",
        "image_17-44-23_1_5.jpg"
    ],
    "messages": [
        {
            "role": "user",
            "content": "Use Wikipedia to search for Jay Chou, and
                tell me which album he released in 2000?"
        },
        {
            "role": "assistant",
            "content": "===Step1===

<think>

<image_17-44-23_1_1.jpg>

**Thought**
The user wants to find the name of Jay Chou's album released in
    2000 on Wikipedia. I need to open the Wikipedia app, search
    for "Jay Chou", navigate to his discography or album section,
    and then find the album released in 2000.

**Action Thought**
The current subgoal is to open the Wikipedia app. Looking at the
    screen information, I can see "Wikipedia" at coordinates [664,
     1023]. This is the app I need to open.

**Action**
Open the Wikipedia app to start searching for "Jay Chou".

</think>

<Observation>

<image17-44-23_1_2.jpg >

**Observation**
1. Open the Wikipedia app.
```


Figure 4: Image from `image_17-44-23_1_1.jpg` to `image_17-44-23_1_5.jpg`

</Observation>

===Step2===

```
<think>

<image17-44-23_1_2.jpg >

**Thought**
The Wikipedia app is open. The next step is to search for "Jay
    Chou". I can see a search bar labeled "Search Wikipedia" at
    the top of the screen. I will tap on this search bar to
    initiate the search.

**Action Thought**
The current subgoal is to search for "Jay Chou". Looking at the
    screenshot, I see a search bar with the text "Search Wikipedia
    ". I need to tap on this search bar to activate the keyboard
    and then type "Jay Chou". The coordinates for "Search
    Wikipedia" are [323, 387].

**Action**
Tap on the "Search Wikipedia" bar to activate the keyboard and
    prepare for typing the search query.

</think>

<Observation>

<image_17-44-23_1_3.jpg >

**Observation**
1. Open the Wikipedia app. 2. Search for "Jay Chou".

</Observation>

===Step3===

<think>

<image_17-44-23_1_3.jpg >

**Thought**
The current screen shows the Wikipedia search interface, and "Jay
    Chou" is listed under "Recent searches". This means the search
     term has been entered, and I can proceed by tapping on "Jay
    Chou" to view the search results. This completes the "Search
    for 'Jay Chou'" subgoal. The next step is to navigate to the
    album/discography section.

**Action Thought**
The current screen shows the Wikipedia search interface. The user'
    s instruction is to search for "Jay Chou". I have already
    opened the Wikipedia app and tapped on the search bar. Now I
    need to type "Jay Chou" into the search bar. Looking at the
    screen information, "Jay Chou" is listed under "Recent
    searches". This means I don't need to type it, I can just tap
    on the existing "Jay Chou" entry.

**Action**
Tap on "Jay Chou" from the recent searches to perform the search.
```

```
</think>

<Observation>

<image_17-44-23_1_4.jpg >

**Observation**
1. Open the Wikipedia app. 2. Search for "Jay Chou". 3. Navigate
    to the album/discography section of the search result.

</Observation>

===Step4===

<think>

<image_17-44-23_1_4.jpg >

**Thought**
The current screen shows the search results for "Jay Chou". The
    goal is to find the album released in 2000. To do this, I need
     to navigate to the album/discography section. Looking at the
    options, "Jay Chou Discography" seems to be the most relevant
    link to find a list of his albums. "Jay (album)" is also an
    album, but it's a specific one, and I need to find the one
    released in 2000, so a discography list would be more
    comprehensive. Therefore, clicking on "Jay Chou Discography"
    is the next logical step.

**Action Thought**
The user wants to find the album released by Jay Chou in 2000. The
     current screen shows search results related to "Jay Chou".
    Among the results, "Jay Chou Discography" seems to be the most
     relevant link to find album information. I will tap on this
    link to navigate to the discography section.

**Action**
Tap on "Jay Chou Discography" to navigate to the discography
    section, which should contain information about his albums and
     their release dates.

</think>

<Observation>

<image_17-44-23_1_5.jpg >

**Observation**
1. Open the Wikipedia app. 2. Search for "Jay Chou". 3. Navigate
    to the album/discography section of the search result. 4.
    Identify the album released in 2000.

</Observation>"
        }
    ],
    "solution": "<answer> Jay </answer>"
}
```

Listing 1: Example of Execution Trajectory 1

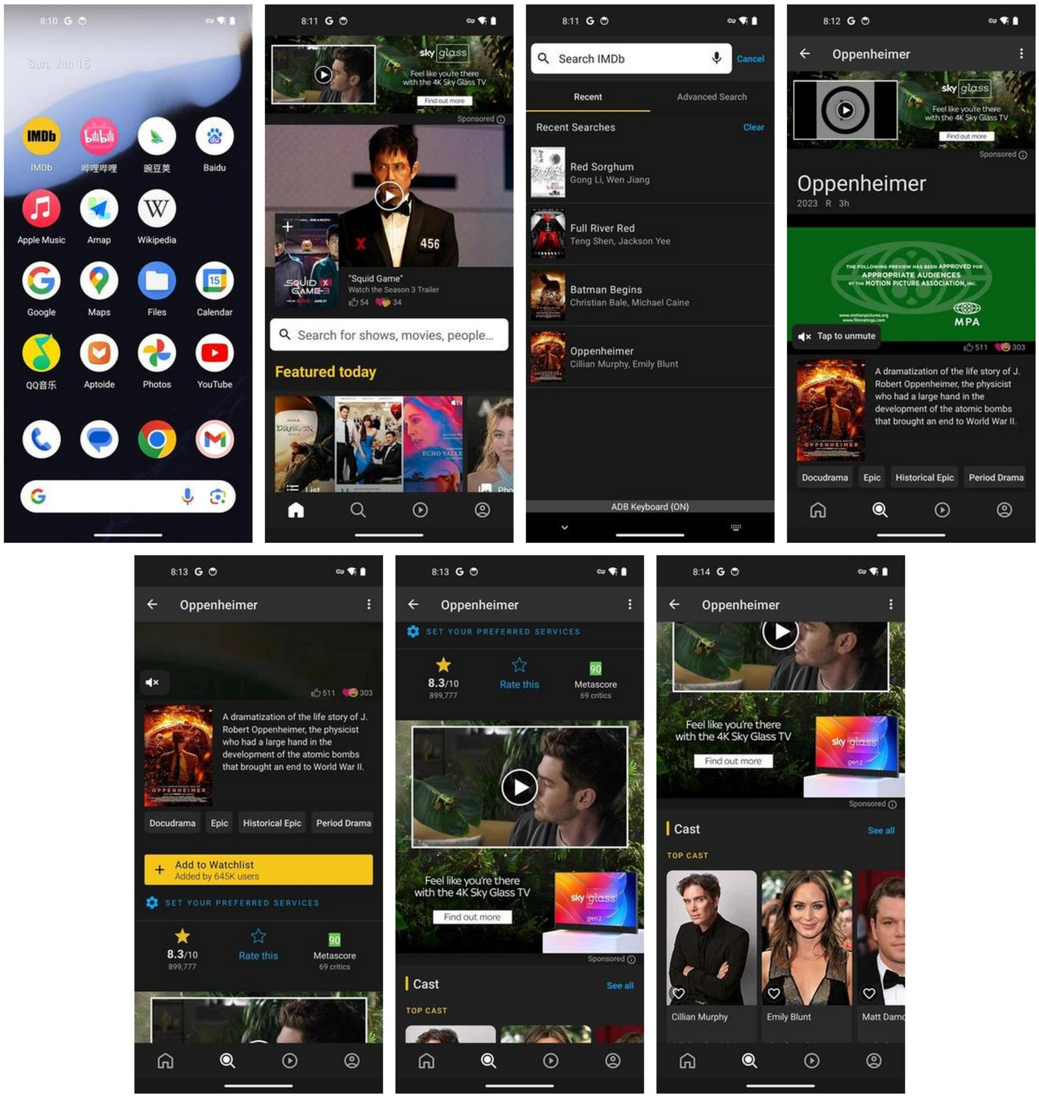



Figure 5: Image from `image_10-41_1_1.jpg` to `image_10-41_1_7.jpg`


```
{
    "images": [
        "image_-10-41_1_1.jpg",
        "image_-10-41_1_2.jpg",
        "image_-10-41_1_3.jpg",
        "image_-10-41_1_4.jpg",
        "image_-10-41_1_5.jpg",
        "image_-10-41_1_6.jpg",
        "image_-10-41_1_7.jpg"
    ],
    "messages": [
        {
            "role": "user",
            "content": "Please use the IMDb app to search for '
                Oppenheimer' and tell me who its first lead actor
                is?"
        },
        {
```

```
            "role": "assistant",
            "content": "===Step1===

<think>

<image_-10-41_1_1.jpg>

**Thought**
The user wants to search for the movie "Oppenheimer" in the IMDb
    app and find its first lead actor. This task can be broken
    down into the following steps: 1. Open the IMDb app. 2. Search
     for "Oppenheimer". 3. Navigate to the movie's details page.
    4. Identify the first lead actor.

**Action Thought**
The user wants to search for a movie in the IMDb app. The first
    step is to open the IMDb app. Looking at the screen
    information, I can see an icon labeled "IMDb" at coordinates
    [162, 570]. This is the app I need to open.

**Action**
Open the IMDb app to begin the search for the movie "Oppenheimer".

</think>

<Observation>

<image_-10-41_1_2.jpg>

**Observation**
1. Open the IMDb app. 2. Search for the movie "Oppenheimer".

</Observation>

===Step2===

<think>

<image_-10-41_1_2.jpg>

**Thought**
The current screen is the IMDb app's home screen. The previous
    subgoal was to open the IMDb app, which is already completed.
    The next step in the plan is to search for "Oppenheimer"
    within the IMDb app. I can see a search bar labeled "Search
    for shows, movies, people..." in the middle of the screen. I
    should tap on this search bar to initiate the search.

**Action Thought**
The current subgoal is to search for "Oppenheimer" within the IMDb
     app. The screenshot shows the IMDb app's home screen, and
    there is a search bar labeled "Search for shows, movies,
    people..." in the middle of the screen. I need to tap on this
    search bar to activate the keyboard and then type "Oppenheimer
    ".

**Action**
Tap on the search bar to activate the keyboard and prepare for
    typing "Oppenheimer".
```

```
</think>

<Observation>

<image_-10-41_1_3.jpg>

**Observation**
1. Open the IMDb app. 2. Search for the movie "Oppenheimer". 3.
   Type "Oppenheimer" into the search bar.

</Observation>

===Step3===

<think>

<image_-10-41_1_3.jpg>

**Thought**
The current screen shows the search results within the IMDb app.
   The movie "Oppenheimer" is already listed under "Recent
   Searches". Since the goal is to search for "Oppenheimer" and
   then find its first lead actor, the next logical step is to
   tap on the "Oppenheimer" search result to view its details.
   This will complete the "Search for 'Oppenheimer' within the
   IMDb app" subgoal and move towards the next one.

**Action Thought**
The current subgoal is to tap on the search result for "
   Oppenheimer" to view its details. Looking at the screen
   information, "Oppenheimer" is listed as a recent search result
    at coordinates [331, 1508]. I should tap on this to proceed.

**Action**
Tap on the "Oppenheimer" search result to view its details page.

</think>

<Observation>

<image_-10-41_1_4.jpg>

**Observation**
1. Open the IMDb app. 2. Search for the movie "Oppenheimer". 3.
   Type "Oppenheimer" into the search bar. 4. Tap on the search
   result for "Oppenheimer" to view its details.

</Observation>

===Step4===

<think>

<image_-10-41_1_4.jpg>

**Thought**
```

```
The current screen shows the movie details page for "Oppenheimer".
     The goal is to find the first actor listed. I need to scroll
    down to find the cast section.

**Action Thought**
The current screen shows the movie details page for "Oppenheimer".
     The goal is to identify the first actor listed. I need to
    scroll down to find the cast section. Since the current view
    only shows the movie description and genre tags, I need to
    swipe up to reveal more content, specifically the cast list.

**Action**
Swipe up to scroll down the page and reveal more content,
    including the cast section, to find the first actor.

</think>

<Observation>

<image_-10-41_1_5.jpg>

**Observation**
1. Open the IMDb app. 2. Search for the movie "Oppenheimer". 3.
    Type "Oppenheimer" into the search bar. 4. Tap on the search
    result for "Oppenheimer" to view its details. 5. Swipe up to
    scroll down the page and reveal more content, including the
    cast section.

</Observation>

===Step5===

<think>

<image_-10-41_1_5.jpg>

**Thought**
The current screen shows the movie details for "Oppenheimer". The
    goal is to find the first actor listed in the cast section.
    The "Important Notes" state that the current view does not
    show the cast section, so further scrolling is needed.
    Therefore, the next logical step is to scroll down to reveal
    the cast section.

**Action Thought**
The current subgoal is to identify the first actor listed in the
    cast section. Based on the screen information and the previous
     action, I have already swiped up to reveal more content.
    However, the cast section is still not visible in the current
    screenshot. Therefore, I need to swipe up again to scroll
    further down the page until the cast section appears.

**Action**
Swipe up again to scroll down the page and reveal more content,
    specifically the cast section, to find the first actor.

</think>

<Observation>
```

```
<image_-10-41_1_6.jpg>

**Observation**
1. Open the IMDb app. 2. Search for the movie "Oppenheimer". 3.
    Type "Oppenheimer" into the search bar. 4. Tap on the search
    result for "Oppenheimer" to view its details. 5. Swipe up to
    scroll down the page and reveal more content, including the
    cast section. 6. Identify the first actor listed in the cast
    section.

</Observation>

===Step6===

<think>

<image_-10-41_1_6.jpg>

**Thought**
The current screen shows the movie details page for "Oppenheimer".
     The "Cast" section is visible, and the goal is to identify
    the first actor listed. The image shows "TOP CAST" and
    partially visible images of actors. To identify the first
    actor, I need to look at the first visible actor's image and
    name. Since the image is only partially visible, I need to
    scroll to the right to fully see the first actor's image and
    name.

**Action Thought**
The current subgoal is to identify the first actor listed in the
    cast section. The screenshot shows the "Cast" section with "
    TOP CAST" visible. Below "TOP CAST", there are images of
    actors. The first image is partially visible. To identify the
    first actor, I need to see the name associated with the first
    image. Since the name is not visible, I need to scroll down
    further to reveal the full cast list or at least the names of
    the top cast members.

**Action**
Swipe up to scroll down the page and reveal the names of the
    actors in the "TOP CAST" section.

</think>

<Observation>

<image_-10-41_1_7.jpg>

**Observation**
1. Open the IMDb app. 2. Search for the movie "Oppenheimer". 3.
    Type "Oppenheimer" into the search bar. 4. Tap on the search
    result for "Oppenheimer" to view its details. 5. Identify the
    first actor listed in the cast section.

</Observation>"
         }
     ],
     "solution": "<answer> Cillian Murphy </answer>"
```

```
}
```

Listing 2: Example of Execution Trajectory 2

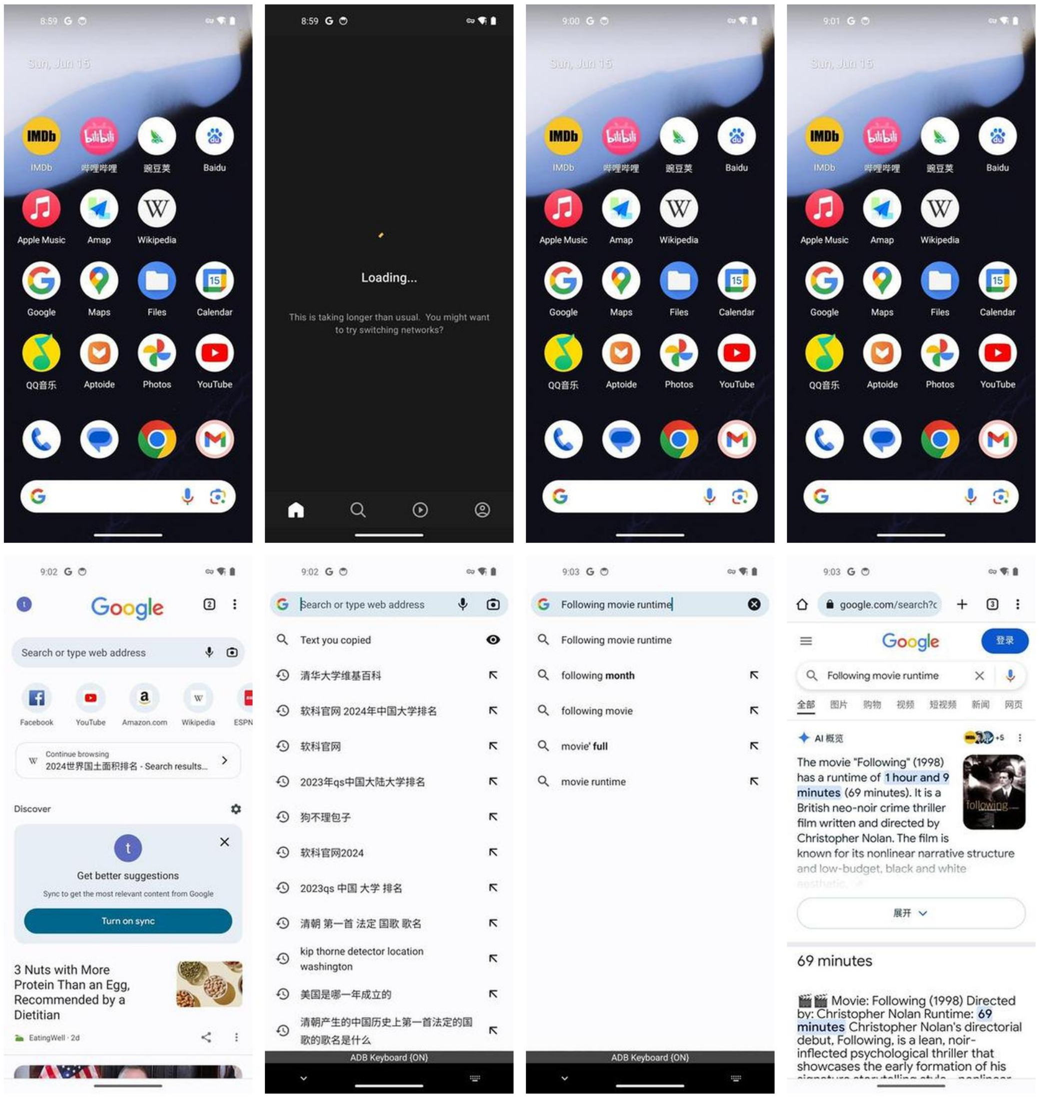



Figure 6: Image from `image_59-16_1_1.jpg` to `image_59-16_1_8.jpg`

```
{
    "images": [
        "image_16-59-16_1_1.jpg",
        "image_16-59-16_1_2.jpg",
        "image_16-59-16_1_3.jpg",
        "image_16-59-16_1_4.jpg",
        "image_16-59-16_1_5.jpg",
        "image_16-59-16_1_6.jpg",
        "image_16-59-16_1_7.jpg",
        "image_16-59-16_1_8.jpg"
    ],
    "messages": [
        {
```

```
            "role": "user",
            "content": "Please use IMDb to search for the movie '
               Following', and tell me what its runtime is?"
        },
        {
            "role": "assistant",
            "content": "===Step1===

<think>

<image_16-59-16_1_1.jpg>

**Thought**
The user wants to find the runtime of the movie "Following". I can
     use a search engine or a movie database app to find this
    information. Since there is an IMDb app on the home screen, it
     would be the most direct way to get movie information. If
    IMDb doesn't work, I can use Google or Baidu to search for it.

**Action Thought**
The user wants to search for movie information, and the overall
    plan is to use the IMDb app. The current subgoal is to open
    the IMDb app. Looking at the screen information, I can see an
    icon labeled "IMDb" at coordinates [162, 570]. This is the
    correct app to open.

**Action**
Open the IMDb app to start searching for the movie information.

</think>

<Observation>

<image_16-59-16_1_2.jpg>

**Observation**
1. Open the IMDb app.

</Observation>

===Step2===

<think>

<image_16-59-16_1_2.jpg>

**Thought**
The IMDb app is stuck on a loading screen, indicating a potential
    network issue. The message "This is taking longer than usual.
    You might want to try switching networks?" suggests that the
    app is having trouble connecting to the internet. Since the
    current plan relies on the IMDb app, and it's not functioning,
     I need to revise the plan to address this. The most
    straightforward way to search for movie information online is
    to use a web browser. Therefore, I will switch to using the
    Chrome browser to search for the movie's runtime.

**Action Thought**
```

```
The current subgoal is to open the Chrome browser. However, the
    previous action was to open the IMDb app, which is currently
    stuck on a loading screen. Since the IMDb app is not loading,
    it's better to close it and open Chrome instead of waiting. To
    do this, I need to go back to the home screen first, then
    open Chrome.

**Action**
Return to the home screen to close the stuck IMDb app and prepare
    to open the Chrome browser.

</think>

<Observation>

<image_16-59-16_1_3.jpg>

**Observation**
1. Open the IMDb app. 2. Open the Chrome browser.

</Observation>

===Step3===

<think>

<image_16-59-16_1_3.jpg>

**Thought**
The current plan is to open the Chrome browser. The previous
    attempt to use the IMDb app failed because it was stuck
    loading. The Chrome browser icon is visible on the screen. I
    need to tap on the Chrome icon to open it.

**Action Thought**
The current subgoal is to open the Chrome browser. Looking at the
    screenshot, I can see the Chrome icon at coordinates [663,
    1888]. I will use the 'Open_App' action to open Chrome.

**Action**
Open the Chrome browser to search for the movie runtime.

</think>

<Observation>

<image_16-59-16_1_4.jpg>

**Observation**
1. Open the IMDb app. 2. Open the Chrome browser.

</Observation>

===Step4===

<think>

<image_16-59-16_1_4.jpg>
```

```
**Thought**
The previous subgoal was to open the Chrome browser. The current
    screenshot shows the home screen with the Chrome browser icon
    visible. The IMDb app was loading slowly, so the plan was to
    switch to Chrome. I need to tap on the Chrome icon to open it.

**Action Thought**
The previous attempt to open Chrome failed. Looking at the
    screenshot, I can see the Chrome icon at coordinates [663,
    1888]. I will try to tap on this icon to open the Chrome
    browser.

**Action**
Tap the Chrome browser icon to open it.

</think>

<Observation>

<image_16-59-16_1_5.jpg>

**Observation**
1. Open the IMDb app. 2. Open the Chrome browser. 3. Search for
    the movie "Following" and its length.

</Observation>

===Step5===

<think>

<image_16-59-16_1_5.jpg>

**Thought**
The current status shows that the Chrome browser is open and
    displaying the Google search page. The previous subgoal was to
     open the Chrome browser, which has been completed. The next
    step in the plan is to search for "Following movie runtime". I
     need to type this into the search bar.

**Action Thought**
The current screen is the Chrome browser, and the goal is to
    search for "Following movie runtime". The search bar is
    visible with the text "Search or type web address". I need to
    tap on this search bar to activate the keyboard and then type
    the search query.

**Action**
Tap the search bar to activate the keyboard and prepare to type
    the search query.

</think>

<Observation>

<image_16-59-16_1_6.jpg>

**Observation**
```

```
1. Open the IMDb app. 2. Open the Chrome browser. 3. Search for
    the movie "Following" and its length. 4. The keyboard is
    activated.

</Observation>

===Step6===

<think>

<image_16-59-16_1_6.jpg>

**Thought**
The current status shows that the Chrome browser is open and the
    search bar is visible with the text "Search or type web
    address". The keyboard is activated. The next step is to type
    "Following movie runtime" into the search bar.

**Action Thought**
The current subgoal is to search for "Following movie runtime".
    The Chrome browser is open, and the keyboard is activated. The
     search bar is visible and ready for input. Therefore, the
    next logical step is to type the search query "Following movie
     runtime" into the search bar.

**Action**
Type "Following movie runtime" into the search bar to search for
    the movie's runtime.

</think>

<Observation>

<image_16-59-16_1_7.jpg>

**Observation**
1. Open the IMDb app. 2. Open the Chrome browser. 3. Search for
    the movie "Following" and its length. 4. The keyboard is
    activated. 5. Typed "Following movie runtime" into the search
    bar.

</Observation>

===Step7===

<think>

<image_16-59-16_1_7.jpg>

**Thought**
The current status shows that "Following movie runtime" has been
    typed into the search bar and the first search suggestion is "
    Following movie runtime". To proceed with the search, I need
    to tap on the search button or the first search suggestion.
    Since the first suggestion is exactly what I typed, tapping on
     it will initiate the search.

**Action Thought**
```

```
The current screen shows the Google search page with the search
    query "Following movie runtime" already typed in the search
    bar. There are also search suggestions below. The first
    suggestion is "Following movie runtime", which is exactly what
     we want to search for. To initiate the search, I need to
    either tap on the search suggestion or press the Enter key.
    Tapping the first search suggestion is a more direct way to
    proceed.

**Action**
Tap the first search suggestion "Following movie runtime" to
    initiate the search.

</think>

<Observation>

<image_16-59-16_1_8.jpg>

**Observation**
1. Open the IMDb app. 2. Open the Chrome browser. 3. Search for
    the movie "Following" and its length. 4. The keyboard is
    activated. 5. Typed "Following movie runtime" into the search
    bar. 6. The search results for "Following movie runtime" are
    displayed, and the movie's runtime is visible.

</Observation>"
         }
     ],
     "solution": "<answer> 69 minutes </answer>"
}
```

Listing 3: Example of Execution Trajectory 3

## F Training Details

The model was trained using an iterative Reinforcement Fine-Tuning (RFT) methodology with `Qwen2.5-VL-7B-Instruct` as the base model. The training was conducted on a computing cluster equipped with four NVIDIA A100 GPUs, running `Ubuntu 20.04` and utilizing the `ms-swift` framework. The RFT process spanned three iterations. **This process was initiated with a preliminary version of the data, consisting of 278 human-verified trajectories.** In each iteration, the model generated 16 candidate trajectories for each sample from this progressively augmented dataset. These candidates were subsequently scored and filtered by a Process Reward Model (PRM, `qwen-max`), with only the high-quality trajectories being retained for the fine-tuning step. The model was then fine-tuned on this curated data for a single epoch using Low-Rank Adaptation (LoRA). Key hyperparameters for the LoRA training included a rank of 8, an alpha of 32, and application to all linear modules. A learning rate of $1 \times 10^{-4}$ was used with a cosine schedule and a warmup ratio of 0.05. The training was configured with a global batch size of 128 (achieved with a per-device batch size of 1 and 32 gradient accumulation steps) and a maximum sequence length of 16,384 tokens. The entire process was optimized using `bfloat16` mixed-precision, DeepSpeed ZeRO stage 2, and FlashAttention. The parameters of the visual encoder remained frozen throughout all training iterations.

## A Appendix

You may include other additional sections here.