Mobile-Agent-RAG: Driving Smart Multi-Agent Coordination with Contextual Knowledge Empowerment for Long-Horizon Mobile Automation

Yuxiang Zhou[1,2,4*], **Jichang Li**[2*], **Yanhao Zhang**[3], **Haonan Lu**[3], **Guanbin Li**[1,2,4†]

[1]School of Computer Science and Engineering, Sun Yat-sen University
[2]Pengcheng Laboratory
[3]OPPO AI Center, OPPO Inc., China
[4]Guangdong Key Laboratory of Big Data Analysis and Processing
zhouyx95@mail2.sysu.edu.cn, li.jichang@pcl.ac.cn, {zhangyanhao,luhaonan}@oppo.com, liguanbin@mail.sysu.edu.cn

Abstract

Mobile agents show immense potential, yet current state-of-the-art (SoTA) agents exhibit inadequate success rates on real-world, long-horizon, cross-application tasks. We attribute this bottleneck to the agents' excessive reliance on static, internal knowledge within MLLMs, which leads to two critical failure points: 1) strategic hallucinations in high-level planning and 2) operational errors during low-level execution on user interfaces (UI). The core insight of this paper is that high-level planning and low-level UI operations require fundamentally distinct types of knowledge. Planning demands high-level, strategy-oriented experiences, whereas operations necessitate low-level, precise instructions closely tied to specific app UIs. Motivated by these insights, we propose `Mobile-Agent-RAG`, a novel hierarchical multi-agent framework that innovatively integrates dual-level retrieval augmentation. At the planning stage, we introduce **Manager-RAG** to reduce strategic hallucinations by retrieving human-validated comprehensive task plans that provide high-level guidance. At the execution stage, we develop **Operator-RAG** to improve execution accuracy by retrieving the most precise low-level guidance for accurate atomic actions, aligned with the current app and subtask. To accurately deliver these knowledge types, we construct two specialized retrieval-oriented knowledge bases. Furthermore, we introduce **Mobile-Eval-RAG**, a challenging benchmark for evaluating such agents on realistic multi-app, long-horizon tasks. Extensive experiments demonstrate that `Mobile-Agent-RAG` significantly outperforms SoTA baselines, improving task completion rate by 11.0% and step efficiency by 10.2%, establishing a robust paradigm for context-aware, reliable multi-agent mobile automation.

Code&Supp. — https://github.com/george13zyx/MAR

Introduction

Mobile agents have demonstrated immense potential in intelligent automation, promising to autonomously accomplish complex user tasks by interacting with smartphone interfaces (You et al. 2024; Yan et al. 2023). However, current state-of-the-art (SoTA) agents still exhibit inadequate success rates when handling real-world, long-horizon, cross-application tasks (Bai et al. 2024). We attribute this bottleneck to a fundamental issue: the agents' excessive reliance on static, internal knowledge embedded in multimodal large language models (MLLMs) (Tang et al. 2025; Xie et al. 2025a). This reliance leads to two critical failure points: 1) strategic hallucinations in high-level planning requiring multi-step reasoning (Xie et al. 2025b; Ji et al. 2024), and 2) operational errors during low-level execution involving specific elements for user interfaces (UI) (Song et al. 2024; Guo et al. 2025).

To address these challenges, hierarchical frameworks, such as `Mobile-Agent-E` (Wang et al. 2025), have been introduced, which decouple high-level planning from low-level execution. Although this architecture improves task performance to some extent, both planning and execution modules still depend on the inherent reasoning capabilities of the models themselves (Hou et al. 2024). This dependence does not fundamentally resolve the hallucination issue caused by static knowledge, resulting in error accumulation during task execution. Retrieval-Augmented Generation (RAG) (Lewis et al. 2020) provides a new approach to mitigate this issue by dynamically retrieving information from external knowledge bases, which has shown success in domains (Yao et al. 2023; Feng et al. 2024; Zhu et al. 2024). However, the effective application of RAG in mobile automotion to specifically address the aforementioned issues at the planning and execution levels remains unexplored.

The core insight of this paper is that high-level planning and low-level UI operations require fundamentally distinct types of knowledge. Planning demands high-level, strategy-oriented experiences, whereas operations necessitate low-level, precise instructions closely tied to specific app UIs. Motivated by these insights, we propose `Mobile-Agent-RAG`, a novel hierarchical multi-agent framework explicitly designed for long-horizon, multi-app mobile automation tasks. Our framework innovatively integrates dual-level retrieval augmentation, significantly boosting agent performance through real-time, contextually relevant external knowledge. At the planning stage, we introduce **Manager-RAG**, retrieving human-validated comprehensive task plans to provide high-level guidance for long-term strategies, effectively reducing strategic halluci-

*Equal Contribution.
†Corresponding author is Guanbin Li.

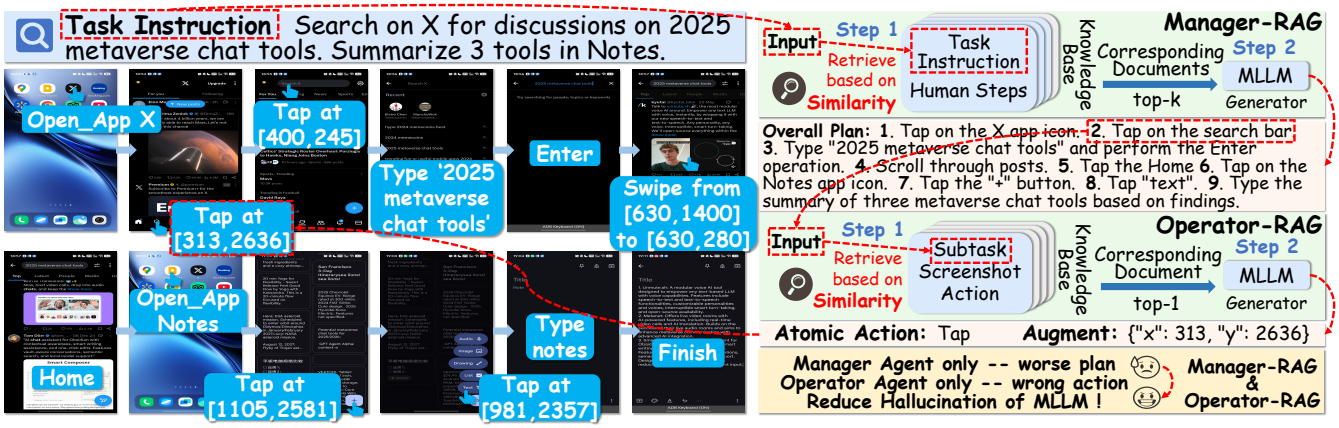

Figure 1: `Mobile-Agent-RAG` is a novel hierarchical multi-agent framework explicitly designed for long-horizon, multi-app mobile automation tasks. `Mobile-Agent-RAG`'s proposed **Manager-RAG** retrieves human-verified task demonstrations to guide high-level plans, while **Operator-RAG** retrieves UI-grounded examples for atomic action generation.

nations. At the execution stage, we develop **Operator-RAG**, which retrieves most precise low-level guidance for accurate atomic actions aligned with the current app and subtask, substantially improving execution accuracy.

To accurately deliver these knowledge types, we construct two specialized retrieval-oriented knowledge bases. The framework dynamically leverages these two core RAG components to systematically overcome the limitations of current agents. Furthermore, we introduce **Mobile-Eval-RAG**, a challenging benchmark featuring realistic multi-app, long-horizon tasks. Extensive experiments demonstrate that our approach significantly surpasses existing state-of-the-art baselines, e.g. `Mobile-Agent-E` (Wang et al. 2025), achieving notable improvements in task completion rates, operator accuracy, step efficiency, and success rates.

In summary, this paper makes the following contributions.

- We propose `Mobile-Agent-RAG`, a hierarchical multi-agent framework enpowered with contextually relevant external knowledge through dual-level retrieval augmentation, namely Manager-RAG and Operator-RAG, for robust long-horizon mobile automation.
- We design two retrieval-oriented knowledge bases, specialized to support Manager-RAG and Operator-RAG, for reducing agent hallucinations and execution errors, and release Mobile-Eval-RAG, a benchmark tailored for evaluating such retrieval-augmented mobile agents.
- Extensive experiments demonstrate that our proposed `Mobile-Agent-RAG` significantly outperforms SoTA baseline algorithms, improving task completion rate by 11.0% and step efficiency by 10.2%, establishing a robust paradigm for context-aware, reliable multi-agent mobile automation.

Related Work

Mobile Agents for UI-Based Task Automation Early mobile task automation focused on enhancing agents' context understanding for graphic user interfaces (UI) through tools and exploratory actions, e.g., `Mobile-Agent` (Wang et al. 2024b) and `AppAgent` (Zhang et al. 2025). These single-agent systems struggled with complex tasks, prompting the shift to multi-agent approaches (Xie et al. 2024). For instance, `M3A` (Rawles et al. 2024) improves task completion by employing multiple agents for collaborative planning and decision-making. `Mobile-Agent-v2` (Wang et al. 2024a) further decomposes tasks into planning, decision, and reflection agents. However, its planning component functions more as a tracker than a strategic planner, and decision-making spans multiple levels. Hence, MLLM-based agents like `DroidBot-GPT` (Wen et al. 2023) and `AutoDroid` (Wen et al. 2024) automate UI interactions by converting GUI states into natural language prompts. Additionally, `MobileGPT` (Lee et al. 2024) introduces a human-like memory system that modularizes tasks into reusable sub-modules via an "explore, select, derive, recall" model. Despite advancements, existing MLLM-based agents face hallucinations (Zheng et al. 2024; Li et al. 2025), undermining their reliability in multi-step mobile tasks. Therefore, we instead propose `Mobile-Agent-RAG`, a retrieval-augmented generation framework that integrates external knowledge to improve context accuracy and reduce hallucinations.

Retrieval-Augmented Agents *vs.* Memory-Driven Agents Retrieval-Augmented Generation (RAG) enhances MLLMs in knowledge-intensive tasks by combining parametric generation with non-parametric memory through semantic retrieval (Lewis et al. 2020). In agent-based systems, RAG facilitates dynamic access to external knowledge during inference, improving factual consistency and planning (Singh et al. 2025). This is evident in web-based agents like `WebGPT` (Nakano et al. 2021) and reasoning-action frameworks like `ReAct` (Yao et al. 2023). Meanwhile, memory-driven agents, such as `MemGPT` (Packer et al. 2023), support long-horizon tasks by retaining context across sessions, ensuring stable reasoning over time (Song et al. 2025). Recent mobile UI agents combine these techniques to ad-

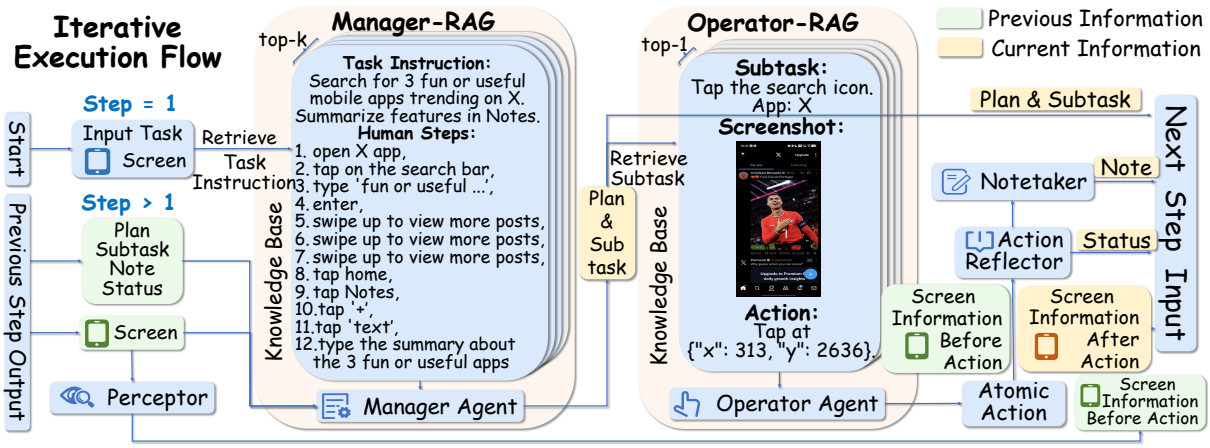

Figure 2: Framework overview of `Mobile-Agent-RAG`: A hierarchical multi-agent system empowered by dedicated knowledge providers, namely **Manager-RAG** and **Operatoer-RAG**, for enhanced high-level planning and precise atomic action generation. Its core operational loop is illustrated: Manager plans, Operator executes, with Perceptor, Action Reflector, and Notetaker providing comprehensive support for mobile task automation.

dress multi-step interactions. Representative works such as `AppAgent-v2` (Li et al. 2024) builds a structured knowledge base during offline exploration for retrieval-driven decision-making, while `AppAgentX` (Jiang et al. 2025) logs execution history to enhance adaptability. Although `Mobile-Agent-E` (Wang et al. 2025) improves task efficiency through agent-driven heuristic summarization, it remains susceptible to hallucinations and lacks explicit task abstraction. To overcome this challenge, our proposed `Mobile-Agent-RAG` integrates a hierarchical multi-agent framework, combining retrieval at both planning and execution levels. By utilizing both structured UI knowledge and external documents, this approach enhances context grounding, reduces hallucination risks, and supports reliable long-horizon mobile automation.

Methodology

This section details `Mobile-Agent-RAG`, a novel framework addressing long-horizon mobile automation by empowering a hierarchical multi-agent system with contextual knowledge. We first elaborate on its multi-agent architecture and execution flow. Subsequently, we detail how our proposed knowledge components enhance core agent decision-making, followed with their correpsonding knowledge base collection, providing a robust solution to challenges like hallucination. As illustrated in Figure 2, `Mobile-Agent-RAG` integrates our proposed dedicated knowledge providers, i.e. **Manager-RAG** and **Operator-RAG**, into the Manager and Operator agents, facilitating high-level task planning and low-level precise atomic actions. Notation definitions used in `Mobile-Agent-RAG` are provided in **Appendix A**.

Hierarchical Multi-Agent Architecture

`Mobile-Agent-RAG` is built upon a hierarchical multi-agent framework, inherited and extended from `Mobile-Agent-E` (Wang et al. 2025). This architecture is designed to achieve a clear separation between high-level planning and low-level execution in mobile task automation, ensuring robustness and adaptability across diverse mobile environments. The framework comprises a central decision-making loop facilitated by a **Manager Agent** and an **Operator Agent**, supported by several specialized modules including a **Perceptor**, an **Action Reflector**, and a **Notetaker**.

Core Agents: Manager and Operator The interaction between the Manager and Operator constitutes the backbone of task execution, coordinating high-level strategic planning and low-level fine-grained operations.

- **Manager Agent (M) empowered by `Manager-RAG`:** The Manager Agent is responsible for high-level strategic planning and subtask decomposition. At each timestep t, it devises the overall plan (P_t) to achieve the user's task instruction (I) and generates the next subtask (T_t^{app}) with the relevant application.

$$P_t, T_t^{\text{app}} = \begin{cases} M(I, D_{MR}, S_{t-1}), & \text{if } t = 1 \\ M(I, P_{t-1}, T_{t-1}^{\text{app}}, S_{t-1}, G_{t-1}, \\ N_{t-1}), & \text{if } t > 1 \text{ and } F_{t-1} = \text{False} \\ M(I, P_{t-1}, T_{t-1}^{\text{app}}, S_{t-1}, G_{t-1}, L^E_{[-k:]}, \\ N_{t-1}), & \text{if } t > 1 \text{ and } F_{t-1} = \text{True} \end{cases}$$

At the initial timestep ($t = 1$), the Manager leverages the input task instruction (I), a *retrieved Manager-RAG document* (D_{MR}), and the initial screenshot (S_{t-1}) to formulate the first overall plan (P_1) and subtask (T_1^{app}). For subsequent timesteps ($t > 1$), it refines its plan based on previous states. In case of consecutive errors ($F_{t-1} =$ True), it consults recent error logs ($L^E_{[-k:]}$) for error recovery. The Manager primarily focuses on abstract planning, thus avoiding fine-grained visual information (V_{t-1}) that might introduce noise for high-level decision-making.

Operator Agent (O) empowered by `Operator-RAG`: The Operator Agent translates the Manager's subtasks into concrete, executable atomic actions. It interacts directly with the mobile environment via Android Debug Bridge (ADB).

$$A_t, S_t = O(I, D_{OR}, P_t, T_t^{\text{app}}, S_{t-1}, V_{t-1}, G_{t-1}, L^E_{[-k:]}, L^A_{[-k:]}, N_{t-1})$$

The Operator generates an atomic action (A_t) based on the current context, including the input task instruction (I), a *retrieved Operator-RAG document* (D_{OR}), the overall plan (P_t), the current subtask (T_t^{app}), the screenshot before action (S_{t-1}), and crucially, the fine-grained visual information before action (V_{t-1}). The combination of S_{t-1} and V_{t-1} enables precise action parameter generation (e.g., specific tap coordinates). The atomic actions A_t are selected from a predefined set, including *Open_App*, *Tap*, *Swipe*, *Type*, *Enter*, *Back*, *Home*, and *Wait*. Refer to **Appendix B** for more details.

Supporting Modules: Perceptor, Action Reflector, and Notetaker These modules augment the core Manager-Operator loop by providing essential information and feedback, ensuring the system's awareness and adaptability within the mobile environment.

- **Perceptor (P): Fine-grained Visual Perception.** The Perceptor analyzes screenshots (S_t) to convert raw visual information into structured, fine-grained information (V_t). This information includes identified text, icons, their precise locations, and contextual descriptions, providing the crucial visual context for other agents. Refer to **Appendix C** for details.

$$V_t = P(S_t)$$

- **Action Reflector (R): Reflection on Action Outcome.** The Action Reflector evaluates the outcome of the Operator's most recent action (A_t), providing critical feedback to the system. It compares the UI states before (S_{t-1}, V_{t-1}) and after (S_t, V_t) the action, considering the current subtask (T_t^{app}) and global progress status (G_{t-1}). It classifies outcomes (e.g., successful, failed due to wrong page, failed due to no change) and updates the global progress status (G_t), action logs (L_t^A), and error logs (L_t^E). This feedback loop is vital for error detection and recovery.

$$\text{Outcome}, G_t, L_t^A, L_t^E = R(I, T_t^{\text{app}}, A_t, S_{t-1}, S_t, V_{t-1}, V_t, G_{t-1})$$

- **Notetaker (N): Information Aggregation.** The Notetaker maintains and updates task-critical information (N_t) throughout execution. It aggregates dynamic details (e.g., search results, extracted phone numbers) from the current state (S_t, V_t, G_t) and previous notes (N_{t-1}). This ensures persistent tracking of information across multiple steps, which is crucial for long-horizon tasks to maintain contextual awareness.

$$N_t = N(I, P_t, T_t^{\text{app}}, S_t, V_t, G_t, N_{t-1})$$

Iterative Execution Flow As illustrated in Figure 2, the system operates in a **dynamic iterative loop**, where agents and modules continuously interact to achieve the user's task instruction by processing information and executing actions. This process ensures adaptive responses to the mobile environment, evolving step-by-step toward task completion. The general flow at each timestep t is as follows:

1. **Perception:** The **Perceptor** module analyzes the current screenshot to extract fine-grained visual information.
2. **High-Level Planning:** The **Manager Agent** uses raw screenshot, along with the overall task and contextual knowledge, to refine the global plan and determine the next high-level subtask.
3. **Low-Level Action:** Based on the Manager Agent's subtask, the **Operator Agent** generates and executes a precise atomic UI action on the mobile device.
4. **Post-Action Perception:** Immediately after an action is performed, the **Perceptor** captures the updated fine-grained visual information for Action Reflector to reflect the changes.
5. **Reflection:** The **Action Reflector** evaluates the outcome of the executed action by comparing raw and fine-grained visual information before and after, providing critical feedback for progress tracking and error detection.
6. **Information Aggregation:** Concurrently, the **Notetaker** module continuously updates and aggregates essential dynamic information as notes, maintaining a persistent and comprehensive context for the ongoing task.

`Mobile-Agent-RAG` is built upon this hierarchical multi-agent architecture. To further enhance this framework, we introduce **Contextual Knowledge Empowerment**, a mechanism utilizing RAG to incorporate external, task-specific knowledge into both the Manager and Operator agents. By mitigating the limitations associated with exclusive reliance on internal representations from MLLMs, our approach significantly improves `Mobile-Agent-RAG`'s capability to address complex and previously unseen mobile tasks. Detailed implementation is described in the subsequent section.

Contextual Knowledge Empowerment via RAG

This section details how external knowledge is retrieved and integrated to empower the Manager and Operator agents, forming the core of `Mobile-Agent-RAG`'s innovation. This process allows the agents to leverage a rich repository of past experiences and domain-specific insights, significantly enhancing their performance beyond what is achievable with internal MLLM knowledge alone.

Manager-RAG (MR) Manager-RAG aims to guide high-level planning, especially during the initial stages of a task.

- **Retrieval:** At the beginning of a task ($t = 1$), the Manager-RAG module takes the input task instruction (I) and retrieves the top-k most relevant (task instruction I_{MR}, human steps H_{MR}) documents (D_{MR}) from

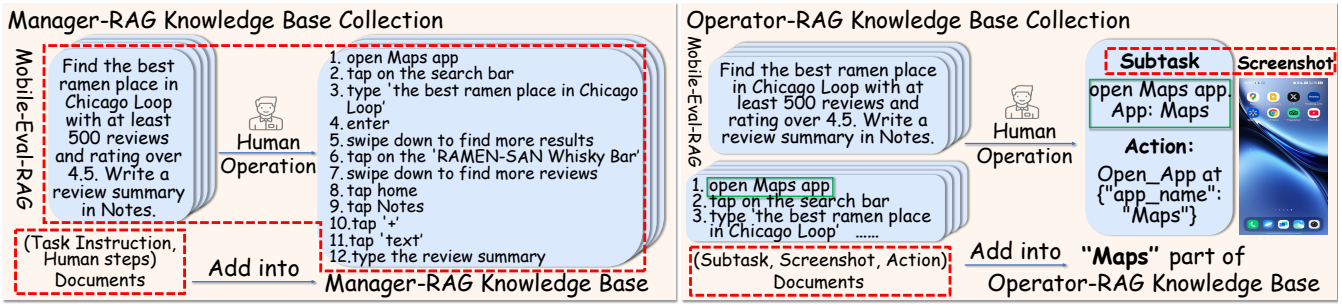

Figure 3: A flow of knowledge base collection for Manager-RAG and Operater-RAG.

Method	Framework Type	CR	OA	RA	Steps	Efficiency	SR
Single-App Task Execution							
AutoDroid	Single-Agent	24.4	59.7	–	30.0	0.81	0.0
Appagent (Auto)	Single-Agent	25.4	63.5	91.0	30.0	0.85	0.0
Appagent (Demo)	Single-Agent	29.2	73.7	92.4	30.0	0.97	0.0
Multi-App Task Execution							
Mobile-Agent	Single-Agent	33.7	60.3	–	29.0	1.16	12.0
Mobile-Agent-v2	Multi-Agent	38.5	61.9	92.5	23.5	1.64	44.0
Mobile-Agent-E	Multi-Agent	58.3	74.1	89.3	22.4	2.60	48.0
Mobile-Agent-E + Evo	Multi-Agent	61.2	77.2	91.0	21.8	2.81	56.0
Mobile-Agent-RAG (Ours)	Multi-Agent	**75.7**	**90.1**	**94.7**	**18.8**	**4.03**	**76.0**

Table 1: Comparison results of `Mobile-Agent-RAG` with previous SoTA algorithms for single-app and multi-app task execution with both single-agent and multi-agent frameworks. **Bold** indicates the best-performing results; the same applies hereafter.

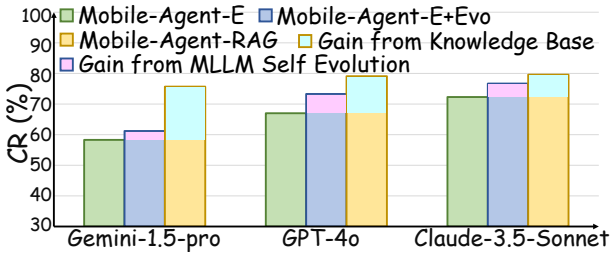

Figure 4: Comparison of performance gain sources for `Mobile-Agent-RAG`, `Mobile-Agent-E`, and `Mobile-Agent-E+Evo` across different MLLMs.

its manually curated knowledge base (K_{MR}). These documents are selected based on the semantic similarity between the input task instruction and indexed task instructions, as determined by the Contriever-MSMARCO model (Izacard et al. 2022). The detailed retrieval algorithm is provided in Algorithm 1 in **Appendix D**.

- **Generation:** The Manager-RAG module receives the overall task instruction I_{query} and the retrieved set of k relevant documents $\mathcal{R}_M$ from its retrieval mechanism. The MLLM generates the overall plan P as follows:

$$P = \text{PromptGen}_{\text{M}}(I_{\text{query}}, \mathcal{R}_M)$$

where $\text{PromptGen}_{\text{M}}(\cdot)$ is a function that formulates the input prompt for the MLLM by combining I_{query} with the retrieved examples in $\mathcal{R}_M$ as few-shot examples.

Method	CR	OA	RA	Steps	Effic.	SR
Simple Operation Tasks						
Mobile-Agent-E	63.4	81.8	89.7	20.3	3.12	80.0
Mobile-Agent-E + Evo	68.3	85.9	86.3	16.8	4.07	80.0
Mobile-Agent-RAG (Ours)	**78.0**	**91.1**	**97.3**	**14.6**	**5.34**	**90.0**
Complex Operation Tasks						
Mobile-Agent-E	54.9	69.7	89.1	23.8	2.31	26.7
Mobile-Agent-E + Evo	56.5	81.3	93.2	25.1	2.25	40.0
Mobile-Agent-RAG (Ours)	**74.2**	**89.6**	**93.5**	**21.6**	**4.30**	**66.7**

Table 2: Comparison resutls of `Mobile-Agent-RAG` with previous SoTA algorithms on tasks with varying complexity.

Operator-RAG (OR) Operator-RAG is responsible for retrieving app-specific knowledge to support the generation of atomic actions required to accomplish a given subtask.

- **Retrieval:** When the Operator is about to execute an atomic action, it sends the current subtask with the identified app name $T_{\text{query}}^{\text{app}}$ to Operator-RAG. The system then restricts retrieval to the app-specific knowledge base K_{OR}^{app} for (subtask T_{OR}, screenshot S_{OR}, action A_{OR}) document (D_{OR}). The subtask is embedded via the same embedding function $f(\cdot)$ (Contriever-MSMARCO) into a vector representation. The system performs retrieval within the app-specific embedding space, selecting the entry with the highest cosine similarity between the

Method	CR	OA	RA	Steps	Effic.	SR
Gemini-1.5-Pro						
Mobile-Agent-E	58.3	74.1	89.3	22.4	2.60	48.0
Mobile-Agent-E + Evo	61.2	77.2	91.0	21.8	2.81	56.0
Mobile-Agent-RAG (Ours)	**75.7**	**90.1**	**94.7**	**18.8**	**4.03**	**76.0**
GPT-4o						
Mobile-Agent-E	67.0	81.5	90.2	18.6	3.60	60.0
Mobile-Agent-E + Evo	73.3	85.6	92.4	**17.8**	4.12	76.0
Mobile-Agent-RAG (Ours)	**79.1**	**88.4**	**97.8**	18.4	**4.30**	**84.0**
Claude-3.5-Sonnet						
Mobile-Agent-E	72.3	91.0	94.8	17.3	4.18	60.0
Mobile-Agent-E + Evo	76.7	91.3	**95.6**	**17.1**	**4.49**	72.0
Mobile-Agent-RAG (Ours)	**79.6**	**92.7**	95.1	18.7	4.26	**84.0**

Table 3: Comparison results of `Mobile-Agent-RAG` with previous SoTA algorithms, evaluated by different MLLMs.

query and indexed subtasks. The detailed retrieval algorithm is provided in Algorithm 2 in **Appendix D**.

- **Generation:** For the Operator-RAG, the reasoning model receives the current subtask, a representation of the current screenshot, and the top-1 most relevant document $\mathcal{R}_O$ from its app-specific knowledge base. This detailed context allows the MLLM to accurately generate the correct atomic action needed to address the current subtask and the current screen state. The generation process can be expressed as:

$$A = \text{PromptGen}_{\text{O}}(T^{\text{app}}_{\text{query}}, S_{\text{current}}, \mathcal{R}_O)$$

where $\text{PromptGen}_{\text{O}}(\cdot)$ is a function that formulates the input prompt for the MLLM by integrating $T^{\text{app}}_{\text{query}}$, S_{current}, and the retrieved example $\mathcal{R}_O$.

Retrieval-Oriented Knowledge Base Collection

To enable retrieval-augmented reasoning for both high-level task planning and low-level action execution, we construct dedicated knowledge bases for Manager-RAG and Operator-RAG, as illustrated in Figure 3. By combining automated logging with human validation, our data collection pipeline produces high-quality, context-rich supervision signals that effectively ground model inference. For additional technical details, please refer to **Appendix E**.

Manager-RAG Knowledge Base (K_{MR}) This knowledge base is designed to support high-level task planning. Each entry in the knowledge base consists of a natural language task instruction paired with its corresponding human-annotated operation steps. These (task instruction, human steps) documents (D_{MR}), are directly extracted from the **Mobile-Eval-RAG** dataset *we construct below*. For each task in Mobile-Eval-RAG, the task instruction is treated as a retrieval query, and the human execution trace is simplified and structured into reference steps for downstream use by MLLMs. All data is manually collected and curated to ensure task-level coherence and accuracy.

Operator-RAG Knowledge Base (K^{app}_{OR}) This knowledge base is intended to support atomic action generation during low-level app interactions. Its construction follows a semi-automated process. During agent execution, we dynamically log three types of information: the current subtask being performed, the corresponding screenshot at that step, and the atomic action generated by the Operator. These (subtask, screenshot, action) documents (D_{OR}) are collected in real time and subsequently verified by human annotators. Invalid or low-quality entries are manually corrected or discarded, ensuring data quality and consistency. To avoid cross-app interference and enhance retrieval precision, we maintain separate retrieval libraries for each mobile application.

Experiments

This section develops the evaluation benchmark **Mobile-Eval-RAG** and designs experiments to thoroughly assess and analyze the performance, module roles, and generalization capability of `Mobile-Agent-RAG`. Due to space constraints, further details on the experiments, e.g. case study and more analysis, can be found in **Appendix F-M**.

Benchmark Dataset: Mobile-Eval-RAG

We introduce **Mobile-Eval-RAG**, a benchmark dataset of 50 diverse and challenging tasks designed specifically to evaluate Retrieval-Augmented Generation (RAG) performance in mobile agent systems. Unlike existing benchmarks such as Mobile-Eval-E (Wang et al. 2025), which have limited suitability for assessing RAG's generalization due to progressively increasing difficulty and insufficient task similarity, Mobile-Eval-RAG emphasizes cross-application coordination and long-horizon planning. Specifically, Mobile-Eval-RAG expands upon Mobile-Eval-E's five core categories—Information Searching, What's Trending, Restaurant Recommendation, Online Shopping, and Travel Planning—by introducing cross-application tasks averaging 16.9 steps across 2-3 applications. These tasks closely reflect realistic scenarios demanding substantial coordination and planning. Additionally, region-specific constraints of application have been removed to enhance general applicability. Queries are MLLM-generated and human-verified to ensure feasibility and consistency. Tasks are divided into simple operations (Information Searching, What's Trending), involving basic searches, and complex operations (Restaurant Recommendation, Online Shopping, Travel Planning), requiring detailed interactions and multi-step coordination, maintaining consistent difficulty levels within each category. In summary, Mobile-Eval-RAG effectively evaluates retrieval alongside long-horizon planning and inter-task coordination, providing a concise yet comprehensive platform for assessing RAG models in complex, multi-app environments.

Experimental Setups

Baselines To evaluate `Mobile-Agent-RAG`, we compare it with state-of-the-art open-source mobile agent frameworks, including `AutoDroid` (Wen et al. 2024), `AppAgent (Auto)` (Zhang et al. 2025), `AppAgent (Demo)` (Zhang et al. 2025), `Mobile-Agent` (Wang

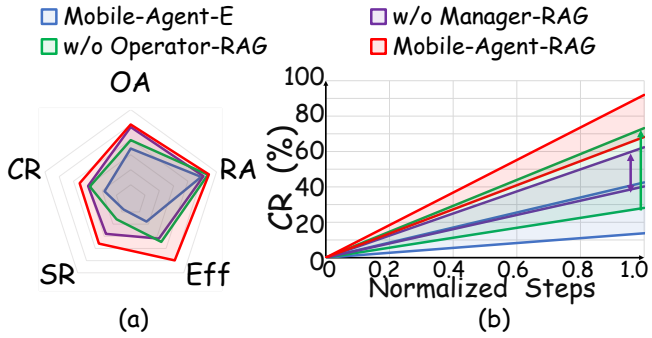

Figure 5: Ablation study results of Mobile-Agent-RAG, with (a) showing a radar chart comparing evaluation metrics across different ablation variants, and (b) presenting CR variations over 10 trials for each variant.

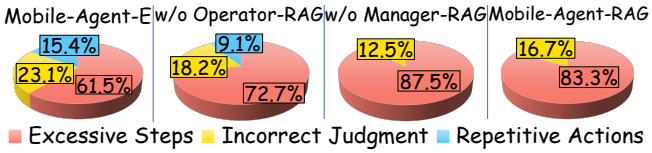

Figure 6: Error type distribution over three task failure modes corresponding to the three evaluation criteria of SR.

et al. 2024b), Mobile-Agent-v2 (Wang et al. 2024a), Mobile-Agent-E (Wang et al. 2025), and Mobile-Agent-E+Evo (Wang et al. 2025). These frameworks include both single-agent and multi-agent architectures, all utilizing MLLMs for mobile task automation. Specifically, Mobile-Agent-E+Evo enhances Mobile-Agent-E by integrating a "Self-Evolution" strategy, while AppAgent (Auto) and AppAgent (Demo) represent two variants proposed by (Zhang et al. 2025) for autonomous exploration and human-demonstration modes, respectively.

Reasoning Models We use several MLLMs as reasoning backbones in our framework, including **GPT-4o** (Hurst et al. 2024), **Claude-3.5-Sonnet** (Anthopic 2024), and **Gemini-1.5-Pro** (Team et al. 2024). Among them, *Gemini-1.5-Pro* achieves the best performance–cost balance, and we use it as the default backbone for efficiency and scalability.

Evaluation Metrics To assess Mobile-Agent-RAG's mobile agent performance on complex tasks, we first use the standard evaluation metrics from Mobile-Agent-E and Mobile-Agent-v2, including **Success Rate** (**SR**, %), **Completion Rate** (**CR**, %), **Operator Accuracy** (**OA**, %), and **Reflector Accuracy** (**RA**, %), and then newly add **Steps** and **Efficiency** to evaluate task execution efficiency; Refer to **Appendix G** for details.

Main Results

To evaluate Mobile-Agent-RAG, we first benchmarked it against SoTA single- and multi-app frameworks, as summarized in Table 1. Results show that Mobile-Agent-RAG consistently outperforms baselines, particularly in complex multi-app tasks, primarily due to accurate human-annotated knowledge rather than less reliable self-generated summaries. Furthermore, we assessed its robustness across tasks of varying complexity in Table 2. While methods like Mobile-Agent-E+Evo yield modest CR improvements of 7.7% on simple and 2.9% on complex tasks, our approach significantly boosts CR by 23.0% (simple) and 35.2% (complex) through effective knowledge augmentation. Furthermore, to confirm Mobile-Agent-RAG's generalization across multiple MLLMs, we conducted additional experiments shown in Table 3. Compared to Mobile-Agent-E, Mobile-Agent-RAG achieves a 23.6% CR gain on weaker models (Gemini-1.5-Pro) and a 5.8% advantage on stronger ones (GPT-4o & Claude-3.5-Sonnet). Finally, Figure 4 confirms that the RAG's benefit negatively correlates with model strength—providing greater compensation for limited reasoning in weaker MLLMs while continuing to improve performance on more capable models.

Ablation Analysis

We conducted an ablation study on Mobile-Agent-RAG to examine the roles of its two core modules, **Manager-RAG** and **Operator-RAG**. Results in Figures 5(a), 5(b) and 6 show that removing Operator-RAG significantly lowers OA, Efficiency, and SR by increasing repetitive and erroneous low-level actions. Conversely, removing Manager-RAG reduces maximum achievable CR, highlighting its essential role in long-horizon planning, although this benefit requires accurate execution provided by Operator-RAG. Further analysis illustrated by Figure 6 confirms their complementarity: Manager-RAG focuses on global task decomposition and provides a more concise strategy, while Operator-RAG ensures precise and context-aware task execution, reducing repetitive local errors. Together, they enable effective mobile automation for complex tasks.

Conclusions

We present Mobile-Agent-RAG, a hierarchical multi-agent framework that introduces dual-level retrieval-augmented generation (RAG) to address the core challenges of long-horizon, multi-app mobile automation. Our approach mitigates the limitations of conventional mobile agents, which often suffer from strategic hallucinations during high-level planning and operational errors in low-level execution due to static MLLM knowledge. By combining Manager-RAG for high-level, human-validated strategic planning and Operator-RAG for low-level, app-specific action grounding, our framework achieves robust and context-aware decision-making across both planning and execution layers. To support this design, we constructed two dedicated retrieval-oriented knowledge bases and released Mobile-Eval-RAG, a challenging benchmark dataset of realistic, long-horizon, multi-app tasks. Extensive experiments demonstrate that Mobile-Agent-RAG achieves significant gains in task completion rate, operator accuracy, and step efficiency over the state-of-the-art baselines, validating dual-level contextual knowledge empowerment for multi-agent coordination.

Acknowledgments

This work is supported in part by the National Key R&D Program of China (2024YFB3908503), and in part by the National Natural Science Foundation of China (62322608).

References

Anthopic, P. 2024. Introducing computer use, a new Claude 3.5 Sonnet, and Claude 3.5 Haiku.

Bai, Z.; Wang, P.; Xiao, T.; He, T.; Han, Z.; Zhang, Z.; and Shou, M. Z. 2024. Hallucination of multimodal large language models: A survey. *arXiv preprint arXiv:2404.18930*.

Feng, S.; Lu, H.; Jiang, J.; Xiong, T.; Huang, L.; Liang, Y.; Li, X.; Deng, Y.; and Aleti, A. 2024. Enabling Cost-Effective UI Automation Testing with Retrieval-Based LLMs: A Case Study in WeChat. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, 1973–1978.

Guo, L.; Liu, W.; Heng, Y. W.; Wang, Y.; et al. 2025. MAPLE: A Mobile Assistant with Persistent Finite State Machines for Recovery Reasoning. *arXiv preprint arXiv:2505.23596*.

Hou, X.; Yang, M.; Jiao, W.; Wang, X.; Tu, Z.; and Zhao, W. X. 2024. Coact: A global-local hierarchy for autonomous agent collaboration. *arXiv preprint arXiv:2406.13381*.

Hurst, A.; Lerer, A.; Goucher, A. P.; Perelman, A.; Ramesh, A.; Clark, A.; Ostrow, A.; Welihinda, A.; Hayes, A.; Radford, A.; et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Izacard, G.; Caron, M.; Hosseini, L.; Riedel, S.; Bojanowski, P.; Joulin, A.; and Grave, E. 2022. Unsupervised Dense Information Retrieval with Contrastive Learning. *Transactions on Machine Learning Research*.

Ji, Z.; Wu, D.; Ma, P.; Li, Z.; and Wang, S. 2024. Testing and understanding erroneous planning in llm agents through synthesized user inputs. *arXiv preprint arXiv:2404.17833*.

Jiang, W.; Zhuang, Y.; Song, C.; Yang, X.; Zhou, J. T.; and Zhang, C. 2025. Appagentx: Evolving gui agents as proficient smartphone users. *arXiv preprint arXiv:2503.02268*.

Lee, S.; Choi, J.; Lee, J.; Wasi, M. H.; Choi, H.; Ko, S.; Oh, S.; and Shin, I. 2024. Mobilegpt: Augmenting llm with human-like app memory for mobile task automation. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, 1119–1133.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474.

Li, J.; Zhang, J.; Jie, Z.; Ma, L.; and Li, G. 2025. Mitigating hallucination for large vision language model by inter-modality correlation calibration decoding. *arXiv preprint arXiv:2501.01926*.

Li, Y.; Zhang, C.; Yang, W.; Fu, B.; Cheng, P.; Chen, X.; Chen, L.; and Wei, Y. 2024. Appagent v2: Advanced agent for flexible mobile interactions. *arXiv preprint arXiv:2408.11824*.

Nakano, R.; Hilton, J.; Balaji, S.; Wu, J.; Ouyang, L.; Kim, C.; Hesse, C.; Jain, S.; Kosaraju, V.; Saunders, W.; et al. 2021. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*.

Packer, C.; Wooders, S.; Lin, K.; Fang, V.; Patil, S. G.; Stoica, I.; and Gonzalez, J. E. 2023. MemGPT: Towards LLMs as Operating Systems. *arXiv preprint arXiv:2310.08560*.

Rawles, C.; Clinckemaillie, S.; Chang, Y.; Waltz, J.; Lau, G.; Fair, M.; Li, A.; Bishop, W.; Li, W.; Campbell-Ajala, F.; et al. 2024. Androidworld: A dynamic benchmarking environment for autonomous agents. *arXiv preprint arXiv:2405.14573*.

Singh, A.; Ehtesham, A.; Kumar, S.; and Khoei, T. T. 2025. Agentic retrieval-augmented generation: A survey on agentic rag. *arXiv preprint arXiv:2501.09136*.

Song, X.; Chen, W.; Liu, Y.; Chen, W.; Li, G.; and Lin, L. 2025. Towards long-horizon vision-language navigation: Platform, benchmark and method. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 12078–12088.

Song, Y.; Bian, Y.; Tang, Y.; Ma, G.; and Cai, Z. 2024. Visiontasker: Mobile task automation using vision based ui understanding and llm task planning. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, 1–17.

Tang, F.; Xu, H.; Zhang, H.; Chen, S.; Wu, X.; Shen, Y.; Zhang, W.; Hou, G.; Tan, Z.; Yan, Y.; et al. 2025. A survey on (m) llm-based gui agents. *arXiv preprint arXiv:2504.13865*.

Team, G.; Georgiev, P.; Lei, V. I.; Burnell, R.; Bai, L.; Gulati, A.; Tanzer, G.; Vincent, D.; Pan, Z.; Wang, S.; et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Wang, J.; Xu, H.; Jia, H.; Zhang, X.; Yan, M.; Shen, W.; Zhang, J.; Huang, F.; and Sang, J. 2024a. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration. *arXiv preprint arXiv:2406.01014*.

Wang, J.; Xu, H.; Ye, J.; Yan, M.; Shen, W.; Zhang, J.; Huang, F.; and Sang, J. 2024b. Mobile-Agent: Autonomous Multi-Modal Mobile Device Agent with Visual Perception. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*.

Wang, Z.; Xu, H.; Wang, J.; Zhang, X.; Yan, M.; Zhang, J.; Huang, F.; and Ji, H. 2025. Mobile-Agent-E: Self-Evolving Mobile Assistant for Complex Tasks. *arXiv preprint arXiv:2501.11733*.

Wen, H.; Li, Y.; Liu, G.; Zhao, S.; Yu, T.; Li, T. J.-J.; Jiang, S.; Liu, Y.; Zhang, Y.; and Liu, Y. 2024. Autodroid: Llm-powered task automation in android. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, 543–557.

Wen, H.; Wang, H.; Liu, J.; and Li, Y. 2023. Droidbot-gpt: Gpt-powered ui automation for android. *arXiv preprint arXiv:2304.07061*.

Xie, B.; Shao, R.; Chen, G.; Zhou, K.; Li, Y.; Liu, J.; Zhang, M.; and Nie, L. 2025a. Gui-explorer: Autonomous exploration and mining of transition-aware knowledge for gui agent. *arXiv preprint arXiv:2505.16827*.

Xie, J.; Chen, Z.; Zhang, R.; Wan, X.; and Li, G. 2024. Large multimodal agents: A survey. *arXiv preprint arXiv:2402.15116*.

Xie, J.; Zhang, K.; Chen, J.; Yuan, S.; Zhang, K.; Zhang, Y.; Li, L.; and Xiao, Y. 2025b. Revealing the Barriers of Language Agents in Planning. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 1872–1888.

Yan, A.; Yang, Z.; Zhu, W.; Lin, K.; Li, L.; Wang, J.; Yang, J.; Zhong, Y.; McAuley, J.; Gao, J.; et al. 2023. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation. *arXiv preprint arXiv:2311.07562*.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

You, K.; Zhang, H.; Schoop, E.; Weers, F.; Swearngin, A.; Nichols, J.; Yang, Y.; and Gan, Z. 2024. Ferret-ui: Grounded mobile ui understanding with multimodal llms. In *European Conference on Computer Vision*, 240–255. Springer.

Zhang, C.; Yang, Z.; Liu, J.; Li, Y.; Han, Y.; Chen, X.; Huang, Z.; Fu, B.; and Yu, G. 2025. Appagent: Multimodal agents as smartphone users. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, 1–20.

Zheng, H.; Xu, T.; Sun, H.; Pu, S.; Chen, R.; and Sun, L. 2024. Thinking before looking: Improving multimodal llm reasoning via mitigating visual hallucination. *arXiv preprint arXiv:2411.12591*.

Zhu, Y.; Ou, Z.; Mou, X.; and Tang, J. 2024. Retrieval-augmented embodied agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17985–17995.