vSpeedUI: Turning Past GUI Experience into Fast Executable Plans

Xiaohan Zheng[1] , **Yihong Chen**[1] , **Haiquan Qiu**[1] , **Quanming Yao**[1,2,3]
[1]Department of Electronic Engineering, Tsinghua University, Beijing, China
[2]Beijing National Research Center for Information Science and Technology, Beijing, China
[3]State Key Laboratory of Space Network and Communications, Beijing, China
qyaoaa@tsinghua.edu.cn

Abstract

LLM-based mobile GUI agents usually invoke large models for nearly every micro-action, making real-device automation slow even when similar workflows have been completed before. We present v**SpeedUI**, a public real-device demo system that turns past GUI experience into fast executable plans. It organizes historical trajectories into an Executable Experience Graph (EXG), where UI states are connected by Semantic Step Summaries with explicit preconditions. At task initialization, vSpeedUI performs Global Look-ahead Planning to retrieve, validate, and rank candidate transitions into a pre-verified plan. During execution, the agent uses lightweight graph traversal with state localization, target adaptation, and fallback when needed. On HarmonyOS, vSpeedUI reduces LLM latency and total task time while maintaining strong success rates, showing a practical route toward data-efficient GUI automation. Code is available at: https://github.com/LARS-research/vSpeedUI.

1 Introduction

LLM-based mobile GUI agents automate smartphone tasks by mapping multimodal UI observations to executable actions, ranging from vision-grounded action prediction to structured text/graph-based element selection [Jiang *et al.*, 2025; Zhou *et al.*, 2025; Zhang *et al.*, 2025; Qin *et al.*, 2025; Ye *et al.*, 2025]. Their progress is commonly assessed in realistic interactive environments and robustness benchmarks [Zhou *et al.*, 2024; Rawles *et al.*, 2025; Chen *et al.*, 2025]. Despite optimizations in model-side efficiency [Yang *et al.*, 2025; Christianos *et al.*, 2025], RL-driven prediction [Lu *et al.*, 2026; Yan *et al.*, 2025; Hu *et al.*, 2025], and verifier-driven pipelines [Dai *et al.*, 2025], most systems still follow a step-wise perception-reasoning-action loop: they perceive the screen, invoke LLMs for the next micro-action, execute it, and repeat. Latency therefore accumulates with task length, limiting real-time deployment on mobile devices.

This work is motivated by data-efficient agentic learning [Wang *et al.*, 2026]: capable agents should not solve every task from scratch, but should organize structured experience, simplify repeated workflows, and reduce interaction-time reasoning cost. Mobile GUI automation is a natural testbed because users often repeat workflows across sessions, while the interface may still vary in layout, state, or task parameters. In this setting, past GUI trajectories should become executable experience rather than demonstrations placed in context.

We present vSpeedUI, a public real-device demo system that turns past GUI experience into fast executable plans. The system organizes historical trajectories into an Executable Experience Graph (EXG), where semantic transitions store step intents and state-level preconditions rather than brittle coordinates. At task initialization, Global Look-ahead Planning retrieves candidate transitions, batch-validates feasibility, and batch-ranks them into a pre-verified execution plan. During runtime, the agent follows this plan through lightweight graph traversal with state localization, target adaptation, and fallback when plan execution is infeasible. Our contributions are summarized as follows.

- We present vSpeedUI, a public real-device demo system that turns past GUI experience into fast executable plans for mobile GUI automation.
- We propose an experience-to-plan pipeline that organizes historical trajectories into executable transitions and uses Global Look-ahead Planning to pre-validate and rank candidate steps.
- Real-device experiments show that vSpeedUI substantially reduces total task time while preserving strong success rates under query variations and growing experience.

2 System Design

Figure 1 illustrates the framework of vSpeedUI. The system first converts recorded GUI trajectories into executable experience and then compiles a pre-verified plan at task initialization. Runtime execution follows the plan when grounding succeeds and falls back to the Base Agent otherwise, so acceleration is conditional rather than hard-constrained.

Executable Experience Graph. We organize historical trajectories into an *Executable Experience Graph (EXG)* $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ (Fig. 1d). Trajectories are collected by a Base Agent that interacts with the device and user to produce step sequences [Dai *et al.*, 2025]. Each node $v \in \mathcal{V}$ denotes a UI state and stores a UI tree/HTML snapshot plus a *State*

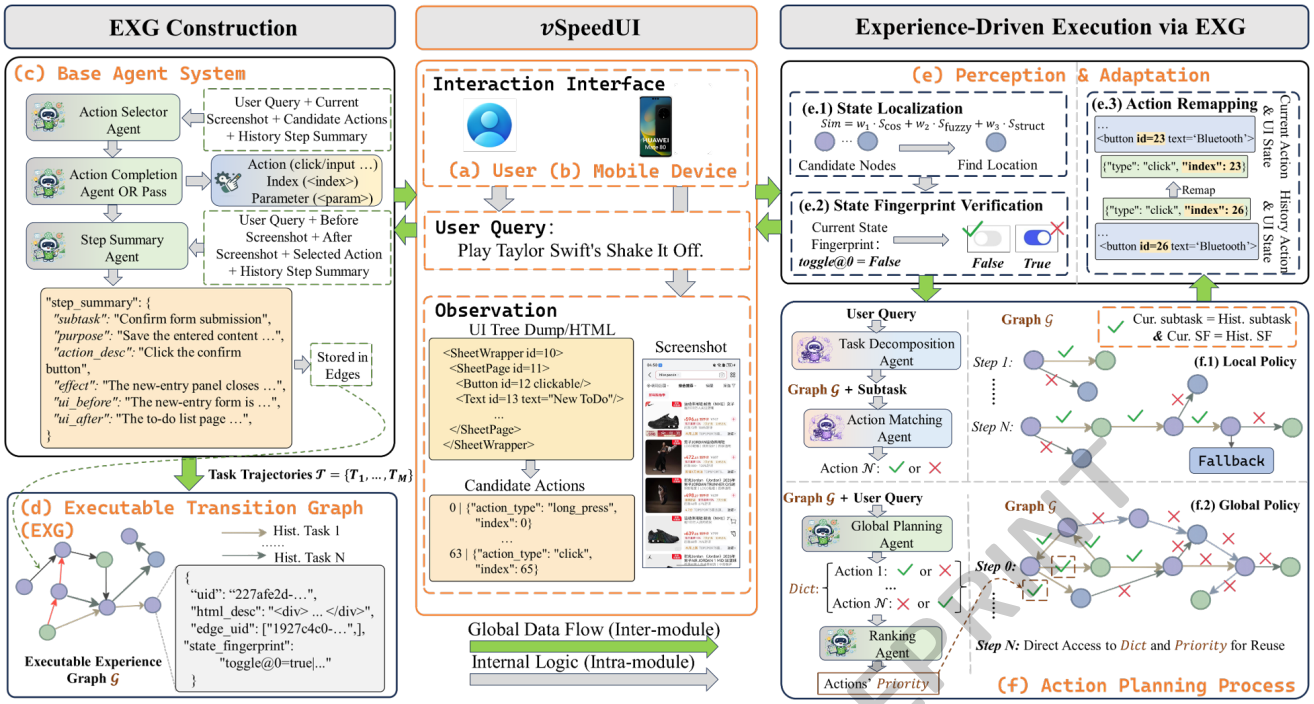

Figure 1: Overview. vSpeedUI converts past GUI trajectories into fast executable plans with EXG. **Left (EXG Construction):** historical trajectories are converted into executable experience transitions annotated with Semantic Step Summaries. **Middle (Environment):** collects user queries and device observations (UI dump tree/screenshot). **Right (Executable Plan Generation):** to make EXG-based plan execution reliable under UI changes, we perform state localization, state fingerprint verification, and context-aware target adaptation. The Global Policy executes Look-ahead Planning to batch-validate and batch-rank candidate steps at Step 0 to compile a pre-verified plan.

Fingerprint for task-relevant dynamic attributes. Each directed edge $e \in \mathcal{E}$ denotes a transition induced by an action and stores executable parameters, origin-task metadata, and a *Semantic Step Summary*. The summary abstracts raw events into `subtask`, a step intent for cross-task matching, and `ui_before`, state-level UI preconditions for valid execution. This representation makes reuse less brittle than coordinate replay: a stored transition can be re-grounded on a changed screen as long as the state and target remain semantically valid. When new trajectories are ingested, vSpeedUI merges matching nodes/edges and appends unseen branches, so experience coverage grows continuously with accumulated runs.

Plan Compilation and Safe Execution. At task start, Global Look-ahead Planning retrieves candidate EXG edges, batch-validates their alignment with the user goal, and ranks valid candidates into a pre-verified executable plan. During execution, State Localization maps the current UI to candidate EXG nodes via stable UI structure, State Fingerprint Verification checks task-critical micro-states (e.g., whether a switch is already toggled), and Context-Aware Target Adaptation remaps action targets under UI drift. Only when localization, preconditions, and target remapping all pass does vSpeedUI execute the planned transition. Otherwise it falls back to the Base Agent, preventing uncertain reuse from accumulating errors on unseen layouts or dynamic content.

Local vs. Global Policy. The Local Policy (Fig. 1 f.1) performs reactive step-wise matching, which is lightweight but still reasons during execution. The Global Policy (Fig. 1 f.2) shifts this reasoning to **Step 0**: it validates many candidate edges in parallel and compiles a pre-verified plan before acting. When multiple valid outgoing edges coexist at the same state, such as different app entry points on the home screen, a Ranking Agent orders candidates using the global goal and origin-task context. This resolves long-term intent at initialization and avoids the linear latency accumulation of repeated step-wise LLM calls.

3 Experiments

We evaluate the demo system along three questions: (i) can past GUI experience be compiled into fast executable plans? (ii) does pre-verified plan execution reduce latency on real devices? and (iii) does the system remain effective under query variations and larger experience collections?

Settings. We evaluate vSpeedUI on a Huawei Mate 80 running HarmonyOS 6, using 10 real-device mobile tasks spanning social media, music, e-commerce, settings, maps, and travel, with an average optimal trajectory length of 8.5 steps; each task is repeated for 15 trials. Baselines include UI-TARS [Qin *et al.*, 2025], Mobile-Agent-v3 [Ye *et al.*, 2025], and GPT-5-based vision/text agents. We report success rate (SR), LLM inference latency (IL), total time (TT), and plan execution rate (PER).

Method	SR (%)↑	IL (s/step)↓	TT (s/task)↓	PER (%)↑
UI-TARS (72B)	44.7	17.5	191.4	-
MA-v3 (32B)	50.7	27.7	276.9	-
GPT-5 (Vision)	47.3	18.1	189.1	-
MA-v3 (GPT-5)	54.0	56.4	515.0	-
GPT-5 (Text)	64.0	20.6	215.3	-
vSpeedUI (GPT-5)	93.3	21.4 /task	79.5	93.2

Notes. Total time accounts for system overhead, averaging 4.55s/step, which covers hardware-level operations such as screenshotting, UI dumping, and environment settling time. And method "MA-v3" is "Mobile-Agent-v3".

Table 1: Main Results.

Benchmarking Comparison. Table 1 shows that step-wise SOTA agents have moderate LLM inference latency, but their end-to-end SR remains below 64% and TT is still in the 190–515s range. In contrast, vSpeedUI (GPT-5) achieves 93.3% SR and reduces TT to 79.5s. The main gain comes from replacing repeated per-step reasoning with a pre-verified executable plan, which yields 93.2% PER.

Effectiveness of Experience-to-Plan Policies. Table 2 ablates reuse policies within vSpeedUI. Without experience-to-plan execution, None Policy reaches 76.7% SR but incurs 59.5s per-step LLM latency and 545.4s per task. Local Policy improves efficiency (140.5s) using lightweight step-wise reuse, while Global Policy further improves both reliability and speed. With GPT-5, Global Policy reaches 93.3% SR, 79.5s TT, and 93.2% PER by shifting LLM reasoning from repeated online calls to one-time plan compilation.

vSpeedUI Plan Policy	SR (%)↑	IL (s) ↓	TT (s/task)↓	PER (%)↑
None (GPT-5)	76.7	59.5 /step	545.4	–
Local (Qwen2.5-7B)	82.7	1.1 /step	140.5	72.9
Global (GPT-4)	86.0	11.7 /task	106.8	84.8
Global (GPT-5)	93.3	21.4 /task	79.5	93.2

Table 2: Efficiency under same-query execution.

Handling Query Variations. Table 3 evaluates four query perturbations: Synonym Paraphrasing (Para.), Step Addition/Deletion (Decr./Incr.), and Parameter Perturbation (Param.). Here, "Qwen2.5" denotes "Qwen2.5-7B-Instruct". Across both tasks, EXG-based plan execution remains robust: Local and Global policies retain high SR while substantially reducing TT versus the Base Agent. Global Policy is the most stable, achieving perfect SR on Memo and low TT under paraphrasing, deletion, and parameter perturbation. The main stress case is step increase on the harder XHS task, where PER drops and Local Policy SR degrades to 60.0%, yet Global Policy still preserves 85.7% SR with far lower TT than the Base Agent.

Scaling with More Experience. We evaluate how vSpeedUI scales when the EXG accumulates four diverse tasks (M=1 . . . 4): social media search (Xiaohongshu), task recording (Memo), music playback (QQ Music), and e-commerce filtering (JD.com). We test the agent on a new, long-horizon cross-app query (e.g., deciding whether to purchase Michael Jackson's *Thriller*). Fig. 2 shows that as M increases, Total Time drops from 1987s to 620s and Action Selection Accuracy (ASA) rises to 96.02%.

Method	Type	Memo (Simple) PER↑	Memo (Simple) SR↑	Memo (Simple) TT↓	XHS (Hard) PER↑	XHS (Hard) SR↑	XHS (Hard) TT↓
Local (Qwen2.5)	Para.	85.7	88.7	80.2	72.7	73.3	180.0
	Decr.	85.7	93.3	78.9	72.7	86.7	179.7
	Incr.	71.4	88.7	124.3	54.6	60.0	264.0
	Param.	85.7	93.3	76.8	63.6	73.3	234.4
Global (GPT-5)	Para.	100.0	100.0	44.3	84.6	85.7	135.0
	Decr.	100.0	100.0	44.0	76.9	92.9	171.7
	Incr.	71.4	100.0	128.8	69.2	85.7	207.6
	Param.	100.0	100.0	42.1	76.9	92.9	170.1
Base Agent	Mixed	n/a	93.3	334.6	n/a	73.3	603.0

Note: PER: Plan Execution Rate (%), **SR**: Success Rate (%), **TT**: Total Time (s).

Table 3: Robustness results under query perturbations.

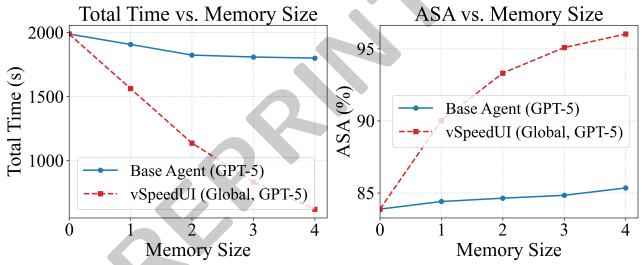

Figure 2: ASA and time vs. experience size M.

Demonstration. Our Demo Video illustrates vSpeedUI solving unseen, complex tasks by dynamically "stitching" transitions from different tasks, showcasing semantic recombination and acceleration over rote replay.

4 Conclusion

We presented vSpeedUI, a public real-device demo system that turns past GUI experience into fast executable plans. By organizing historical trajectories into executable experience transitions and validating candidate plans at task initialization, vSpeedUI reduces repeated step-wise LLM reasoning during mobile GUI automation. The demo shows that structured experience organization and plan execution can serve as practical mechanisms for data-efficient agentic learning [Wang *et al.*, 2026]: agents can become faster and more reliable not only by using stronger models, but also by better organizing and executing what they have already experienced.

Acknowledgments

This work is supported by Beijing Science and Technology Program (No.Z251100008125003).

Ethical Statement

There are no ethical issues associated with this work.

References

[Chen *et al.*, 2025] W. Chen, Z. Wang, L. Yang, S. Zhou, X. Tang, J. Bu, Y. Li, and W. Jiang. PG-Agent: An agent powered by page graph. In *ACM MM*, 2025.

[Christianos *et al.*, 2025] F. Christianos, G. Papoudakis, T. Coste, J. Hao, J. Wang, and K. Shao. Lightweight neural app control. In *ICLR*, 2025.

[Dai *et al.*, 2025] G. Dai, S. Jiang, T. Cao, Y. Li, Y. Yang, R. Tan, M. Li, and L. Qiu. Advancing mobile GUI agents: A verifier-driven approach to practical deployment. *arXiv*, 2025.

[Hu *et al.*, 2025] Z. Hu, S. Xiong, Y. Zhang, S.-K. Ng, A. T. Luu, B. An, S. Yan, and B. Hooi. Guiding VLM agents with process rewards at inference time for GUI navigation. *arXiv*, 2025.

[Jiang *et al.*, 2025] W. Jiang, Y. Zhuang, C. Song, X. Yang, J. T. Zhou, and C. Zhang. AppAgentX: Evolving GUI agents as proficient smartphone users. *arXiv*, 2025.

[Lu *et al.*, 2026] Z. Lu, Y. Chai, Y. Guo, X. Yin, L. Liu, H. Wang, H. Xiao, S. Ren, G. Xiong, and H. Li. UI-R1: Enhancing efficient action prediction of GUI agents by reinforcement learning. In *AAAI*, 2026.

[Qin *et al.*, 2025] Y. Qin, Y. Ye, J. Fang, H. Wang, S. Liang, S. Tian, J. Zhang, J. Li, Y. Li, S. Huang, W. Zhong, K. Li, J. Yang, Y. Miao, W. Lin, L. Liu, X. Jiang, Q. Ma, J. Li, X. Xiao, K. Cai, C. Li, Y. Zheng, C. Jin, C. Li, X. Zhou, M. Wang, H. Chen, Z. Li, H. Yang, H. Liu, F. Lin, T. Peng, X. Liu, and G. Shi. UI-TARS: Pioneering automated GUI interaction with native agents. *arXiv*, 2025.

[Rawles *et al.*, 2025] C. Rawles, S. Clinckemaillie, Y. Chang, J. Waltz, G. Lau, M. Fair, A. Li, W. Bishop, W. Li, F. Campbell-Ajala, D. Toyama, R. Berry, D. Tyamagundlu, T. Lillicrap, and O. Riva. AndroidWorld: A dynamic benchmarking environment for autonomous agents. In *ICLR*, 2025.

[Wang *et al.*, 2026] Y. Wang, Z. Luo, P. Zhao, Y. Cai, and Q. Yao. Beyond scaling: A survey on data-efficient agentic learning. In *IJCAI*, 2026.

[Yan *et al.*, 2025] H. Yan, Y. Shen, X. Huang, J. Wang, K. Tan, Z. Liang, H. Li, Z. Ge, O. Yoshie, S. Li, X. Zhang, and D. Jiang. GUI exploration lab: Enhancing screen navigation in agents via multi-turn reinforcement learning. In *NeurIPS*, 2025.

[Yang *et al.*, 2025] Z. Yang, Z.-Y. Dou, D. Feng, F. Huang, A. T. Nguyen, K. You, O. Attia, Y. Yang, M. Feng, H. Zhang, R. Ramrakhya, C. Jia, J. Nichols, A. Toshev, Y. Yang, and Z. Gan. Ferret-UI Lite: Lessons from building small on-device GUI agents. *arXiv*, 2025.

[Ye *et al.*, 2025] J. Ye, X. Zhang, H. Xu, H. Liu, J. Wang, Z. Zhu, Z. Zheng, F. Gao, J. Cao, Z. Lu, J. Liao, Q. Zheng, F. Huang, J. Zhou, and M. Yan. Mobile-Agent-v3: Fundamental agents for GUI automation. *arXiv*, 2025.

[Zhang *et al.*, 2025] Z. Zhang, Y. Lu, Y. Fu, Y. Huo, S. Yang, Y. Wu, H. Si, X. Cong, H. Chen, Y. Lin, J. Xie, W. Zhou, W. Xu, Y. Zhang, Z. Su, Z. Zhai, X. Liu, Y. Mei, J. Xu, H. Tian, C. Wang, C. Chen, Y. Yao, Z. Liu, and M. Sun. AgentCPM-GUI: Building mobile-use agents with reinforcement fine-tuning. In *EMNLP Demos*, 2025.

[Zhou *et al.*, 2024] S. Zhou, F. F. Xu, H. Zhu, X. Zhou, R. Lo, A. Sridhar, X. Cheng, T. Ou, Y. Bisk, D. Fried, U. Alon, and G. Neubig. WebArena: A realistic web environment for building autonomous agents. In *ICLR*, 2024.

[Zhou *et al.*, 2025] H. Zhou, X. Zhang, P. Tong, J. Zhang, L. Chen, Q. Kong, C. Cai, C. Liu, Y. Wang, J. Zhou, and S. Hoi. MAI-UI technical report: Real-world centric foundation GUI agents. *arXiv*, 2025.