MANGO: Multi-Agent Web Navigation via Global-View Optimization

Weixi Tong
Purdue University
tong172@purdue.edu

Yifeng Di
Purdue University
di5@purdue.edu

Tianyi Zhang
Purdue University
tianyi@purdue.edu

Abstract

Existing web agents typically initiate exploration from the root URL, which is inefficient for complex websites with deep hierarchical structures. Without a global view of the website's structure, agents frequently fall into navigation traps, explore irrelevant branches, or fail to reach target information within a limited budget. We propose MANGO, a multi-agent web navigation method that leverages the website structure to dynamically determine optimal starting points. We formulate URL selection as a multi-armed bandit problem and employ Thompson Sampling to adaptively allocate the navigation budget across candidate URLs. Furthermore, we introduce an episodic memory component to store navigation history, enabling the agent to learn from previous attempts. Experiments on WebVoyager demonstrate that MANGO achieves a success rate of 63.6% when using GPT-5-mini, outperforming the best baseline by 7.3%. Furthermore, on WebWalkerQA, MANGO attains a 52.5% success rate, surpassing the best baseline by 26.8%. We also demonstrate the generalizability of MANGO using both open-source and closed-source models as backbones. Our data and code are open-source and available at https://github.com/VichyTong/Mango.

1 Introduction

There has been growing interest in building web agents using large language models (LLMs) (Zhou et al., 2024c; Wang et al., 2023; Wu et al., 2025; Akkil et al., 2024; Yang et al., 2025). These agents typically start from the root URL (i.e., the homepage) of a website, navigate and parse web content, and interact with the browser through actions such as clicking buttons and typing into text fields.

Given the complexity of real-world websites, the root URL of a website may not be an optimal starting point. These agents have to traverse the website's structure from the top down, often exploring irrelevant sub-trees, falling into navigation traps, or wasting computation on generic intermediate pages. While prior work aims to enhance web navigation efficiency via better decision-making (Abuelsaad et al., 2024; Tan et al., 2025) or alignment (Kim et al., 2024; Yang et al., 2025), such methods are still limited by partial observations at each step without a global view of the website's structure.

To address this limitation, we propose MANGO, a **M**ulti-**A**gent **N**avigation method with **G**lobal-view **O**ptimization for web automation. Given a target website, MANGO first constructs the global structure of the website through a lightweight crawl and a site-specific keyword search. Building upon this structure, MANGO further identifies a candidate set of URLs that are most relevant to the user query. Given a limited budget, MANGO dynamically selects the next URL to visit via Thompson Sampling. After each navigation attempt, MANGO reflects on the navigation trajectory and updates the probabilistic distribution of Thompson Sampling. It also stores both the reflection and the trajectory in an episodic memory to prevent redundant visits.

MANGO is evaluated on two web navigation benchmarks: WebVoyager (He et al., 2024) and WebWalkerQA (Wu et al., 2025). Following prior works on web navigation (Wu et al., 2025; Yang et al., 2025), we use success rate (SR) and action count (AC) to measure the performance and efficiency of MANGO. We experiment with five LLMs as backbones for MANGO and compare it against two state-of-the-art (SOTA) methods, AgentOccam (Yang et al., 2025) and WebWalker (Wu et al., 2025), on both benchmarks. The results show that MANGO consistently achieves superior SR compared to SOTA baselines across all five backbone models, yielding absolute improvements of 3.1% to 7.3% on WebVoyager and 4.6% to 26.8% on WebWalkerQA. Regarding efficiency, MANGO maintains competitive or lower AC on the Qwen3 models. While it incurs higher AC when using GPT-5-

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 14209–14223

mini, this investment translates into performance gains, yielding a 7.3% absolute improvement on WebVoyager and a 26.8% absolute improvement on WebWalkerQA compared to the best baseline. Finally, ablation studies show that MANGO significantly outperforms variants based on random URL selection, search-based URL selection, and MCTS-guided navigation, which confirms the effectiveness of global view analysis and Thompson Sampling used in MANGO.

2 Related Work

2.1 Web Navigation Agents

Recent advances in LLMs have significantly accelerated research on web navigation agents (Ning et al., 2025). One line of research focuses on improving agents' perception of the browser environment (Deng et al., 2023; Zheng et al., 2024), e.g., by summarizing long HTML documents into task-relevant snippets to extract salient environmental information (Gur et al., 2024). Another line of research focuses on planning and decision-making, where complex tasks are decomposed into a sequence of sub-tasks that are executed step by step (Li et al., 2023; Abuelsaad et al., 2024; Tan et al., 2025). A third line of research aims to refine action execution through better grounding in the browser environment, e.g., aligning natural language actions with correct UI elements (Lin et al., 2025; Kim et al., 2024).

However, these methods share a common limitation: they typically initiate navigation from the root URL (i.e., the homepage), leading to inefficient exploration on large and complex websites. In contrast, our method explicitly constructs a global structural representation of the website to identify intent-related entry points for navigation.

2.2 Agentic Search Strategies

To enhance the multi-step reasoning capability of agents, different search strategies have been adopted to navigate the solution space of a task in different domains (Gan et al., 2025; Antoniades et al., 2025; Yu et al., 2025). For instance, Language Agent Tree Search (Zhou et al., 2024a) integrates Monte Carlo Tree Search (MCTS) with LLM-powered value functions and self-reflection to optimize decision-making in the generation process. In web navigation, WebPilot (Zhang et al., 2024) adopts MCTS with a dual optimization strategy. Koh et al. (2025) adopts a best-first search algorithm to explore diverse interaction trajectories for improved task completion.

While effective, these methods still must discover the website structure incrementally from the homepage. Our method differs by pruning the search space before navigation begins. Instead of expanding a massive search tree from the root, we use the global structure of the website to identify candidate URLs and then use Thompson Sampling to allocate the computational budget efficiently. As shown in Section 4.3, MANGO consistently outperforms the MCTS variant, achieving absolute improvements ranging from 5.2% to 17.1%.

2.3 Agentic Memory

Agentic memory is an effective way to transform stateless LLMs into adaptive agents by enabling the autonomous retention, organization, and retrieval of experiential knowledge for long-horizon planning and continuous adaptation (Xu et al., 2025). Some web agents adopt short-term memory, which stores previous actions in a web navigation trajectory (Guan et al., 2024; Lai et al., 2024). Zheng et al. (2024) propose a method to store and retrieve web navigation trajectories of successfully completed tasks to guide the current navigation. Agashe et al. (2025) introduced Narrative Memory, which stores high-level summaries of past experiences and real-time information retrieved from the web.

Building on these foundations, we integrate an episodic memory module to prevent the agent from repeating unsuccessful actions. MANGO records the actions performed in each navigation trajectory and its reflection on the trajectory. During subsequent navigation to the same URL, the stored information is provided to the navigation agent to prevent it from repeating incorrect actions.

3 Method

Figure 1 provides an overview of MANGO. Given the user query q and the root URL u_r, MANGO first constructs and analyzes the website structure to identify a candidate set of URLs. Under a limited navigation budget, MANGO models URL selection as a Multi-Armed Bandit (MAB) problem and applies Thompson sampling to adaptively prioritize promising URLs during navigation. For each selected URL, the web navigation agent interacts with the browser to navigate starting from that URL. After each navigation attempt, the reflection agent evaluates the navigation trajectory and pro-

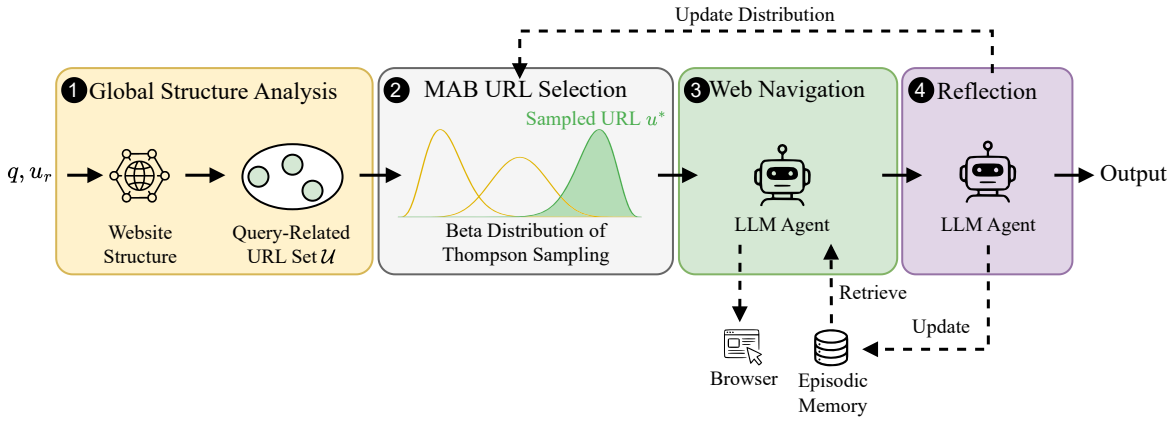


Figure 1: Overview of the web navigation agent system.

vides feedback to update the bandit posterior for the next navigation attempt.

3.1 Global Structure Analysis

Existing web navigation agents typically begin their exploration from the root URL (Gur et al., 2024; Zhou et al., 2024a). While effective for small-scale sites, this strategy becomes inefficient for large-scale websites with thousands of pages, as it forces the agent to incrementally traverse the enormous search space. To address this limitation, MANGO constructs and analyzes the global hierarchical structure of a website to identify a set of query-related URLs as starting points, denoted as $\mathcal{U}$, prior to navigation.

Specifically, given the root URL u_r, MANGO first performs lightweight web crawling via breadth-first search (BFS) to collect a set of reachable URLs starting from u_r. During crawling, MANGO filters out URLs that point to non-HTML files (e.g., images and videos) as well as URLs that lead to external websites, retaining only pages within the same domain. To ensure the process completes within a reasonable time, we set a limit τ on the maximum number of pages to crawl. To ensure simplicity and cost efficiency, MANGO employs the BM25 ranking algorithm (Robertson et al., 2009) to score these URLs based on the semantic similarity of their content to the query q. The top 10 URLs are added to $\mathcal{U}$. We describe the implementation details of this web crawling process in Appendix A.1.

However, some websites contain a vast number of webpages that cannot be fully reached through crawling alone due to time and depth constraints. For example, arXiv has over 2.9M pages of research papers.[1] Consequently, crawling the entire site is practically infeasible. In such cases, relying solely on web crawling is insufficient. As Google has already indexed most pages within a website's hierarchy, MANGO asks the LLM to recommend search keywords based on the user query (Prompt shown in Table 6). Then, it leverages the `site` operator of Google Search to retrieve additional relevant pages and add the top 10 results into $\mathcal{U}$.

3.2 URL Prioritization and Selection

After the global structure analysis component identifies the candidate URLs set $\mathcal{U}$, MANGO needs to prioritize these URLs for navigation under a limited navigation budget. Inspired by Chakrabarti et al. (2008), we model this task as a multi-armed bandit (MAB) problem with finite-lifetime arms. Unlike Chakrabarti et al. (2008), which assigns a fixed lifetime to each arm (e.g., an arm i is deactivated after being selected L_i times), the deactivation of an arm in our method is dynamically determined by the reflection agent based on the navigation history. We formalize this variant of the MAB problem as follows:

- **Arms**: The set of candidate URLs $\mathcal{U}$. Each URL u_i maintains a dynamic status, being either *Active* or *Exhausted*.
- **Selection Strategy**: A probabilistic strategy based on Thompson Sampling that selects the most promising URL from the set of currently active arms, denoted as $\mathcal{U}_{act}$.
- **State Transitions**: A dual-update logic that modifies both the probability distribution and the active status of the arm based on navigation outcomes.

[1] https://arxiv.org/stats/main

We initialize the Beta distribution parameters (α_u, β_u) for each arm based on the relevance between the web content of a URL and the user query. Specifically, we reuse the BM25 relevance scores calculated by the global structure analysis component to initialize these two parameters. Let λ_u be the BM25 score of the content of URL u, we normalize λ_u to ρ_u:

$$\rho_u = \frac{\lambda_u - \min_{u' \in \mathcal{U}_r}(\lambda_{u'})}{\max_{u' \in \mathcal{U}_r}(\lambda_{u'}) - \min_{u' \in \mathcal{U}_r}(\lambda_{u'}) + \epsilon}$$

Here, ϵ is a small constant for numerical stability. Then, we initialize α_u, β_u as follows:

$$\alpha_u^{(0)} = 1 + \kappa \cdot \rho_u, \quad \beta_u^{(0)} = 1 + \kappa \cdot (1 - \rho_u)$$

Here, κ is the weight parameter of the relevance.

At each step t, the agent selects a URL u^* to visit by sampling a value θ_u from the posterior Beta distribution of each *Active* arm and choosing the maximum:

$$u^* = \arg\max_{u \in \mathcal{U}_{act}} \left\{ \theta_u^{(t)} : \theta_u^{(t)} \sim \text{Beta}(\alpha_u^{(t)}, \beta_u^{(t)}) \right\}$$

After each navigation, the reflection agent outputs a status based on the navigation trajectory (detailed in Section 3.4). We assign a Bernoulli reward $r \in \{0, 1\}$ for each type of status. Given the reward value $r^{(t)}$ of the status, MANGO updates the distribution parameters as follows:

$$\alpha_u^{(t+1)} = \alpha_u^{(t)} + r^{(t)}, \beta_u^{(t+1)} = \beta_u^{(t)} + (1 - r^{(t)}) \quad (1)$$

Furthermore, if the status indicates the navigation of the URL reached a dead end, MANGO will mark this URL as exhausted so it will not be considered in future URL selection and navigation.

3.3 Web Navigation Agent

The input to the web navigation agent consists of the user query q and the selected URL u^*, which is chosen by Thompson Sampling. The web navigation agent is constrained with a navigation budget b to limit the maximum actions to perform. If the URL has been previously visited, MANGO retrieves the navigation trajectory and reflection summary of the previous visit from the episodic memory and appends them to the input of the navigation agent. The navigation agent then decides which actions to perform on the browser environment and receives browser observations in response iteratively.

Specifically, MANGO treats the browser environment as a plug-in component. This architectural choice allows the navigation agent to interact with different web browsing environments and action spaces. To ensure a fair comparison with baselines, we align our browser environment settings with those used by leading methods on each benchmark. For example, when evaluating on WebVoyager (He et al., 2024), we adopt the Playwright-based environment[2] used by AgentOccam. Similarly, for the WebWalkerQA benchmark (Wu et al., 2025), we utilize the Crawl4AI environment[3] employed by WebWalker. Adopting the same browsing environments as these SOTA methods makes it easier to run the experiments and establish a fair comparison with these web agents.

Finally, the web navigation agent may end in one of two states. First, it may have generated a response indicating completion of the task. However, there is a chance that the task is not yet fully completed, especially for tasks that require multiple actions, e.g., scraping multiple pieces of information from a webpage, filling in multiple text fields, etc. Second, the agent may exhaust its navigation budget b before completing the task. This case is more challenging. The agent may be following a promising path but is forced to terminate early due to budget constraints, or it may be pursuing an unpromising path that would ultimately lead to a dead end. To carefully evaluate the status of the navigation attempt, the web navigation agent shares its navigation history with the reflection agent and hands off control to it.

3.4 Reflection Agent

We introduce a reflection agent to further analyze the navigation trajectory and the final output of this navigation to determine the navigation attempt's status. If the web navigation agent indicates task completion, the reflection agent will assess whether the actions performed in the navigation trajectory and final output satisfy the user query. The prompt for this reflection is shown in Table 8 in Appendix C. If the answer is considered adequate by the reflection agent, MANGO terminates and outputs the final result. However, if the result is inadequate (e.g., the extracted information is partial or the interaction is incomplete), MANGO treats this as a promising path that requires further exploration. Consequently, it assigns a positive reward $r = 1$ to update the Beta distribution pa-

[2] https://github.com/microsoft/playwright-mcp
[3] https://github.com/unclecode/crawl4ai

rameters (Eq. 1), thereby increasing the probability of re-selecting this URL in future iterations.

In the case of budget exhaustion, the reflection agent will evaluate whether the current navigation trajectory remains promising and should be continued. The prompt is shown in Table 9 in Appendix C. If the reflection agent deems the page is relevant but requires more navigation to complete the task, MANGO assigns a reward $r = 1$ to maintain a high probability of selection. Otherwise, MANGO assigns a negative reward $r = 0$, which updates the Beta distribution to discourage the agent from revisiting unpromising URLs.

The resulting status is then stored in the episodic memory and returned to the Thompson sampling selector to update the probability distribution. Furthermore, MANGO stores the navigation trajectory, the final output of this navigation, and the reflection in the episodic memory. In the next round of navigation, if the same URL is visited again, this information will be retrieved to help the web navigation agent to make a more informed decision without repeating the actions on the same URL.

4 Experiment

4.1 Experimental Settings

Benchmarks We conduct experiments on two web navigation benchmarks, WebVoyager (He et al., 2024) and WebWalkerQA (Wu et al., 2025). WebVoyager consists of web navigation tasks on popular real-world websites, such as Amazon and Coursera. To evaluate MANGO on WebVoyager without the subjectivity introduced by human annotations, we follow AgentOccam (Yang et al., 2025) to use the 129 filtered QA tasks with golden answers. WebWalkerQA is a benchmark that contains 680 web navigation tasks across websites in four different domains, including conference, education, organization, and game. Furthermore, WebWalkerQA includes both single-source and multi-source QA tasks. Single-source tasks require deep exploration from the root URL to locate information on a single page, whereas multi-source tasks require integrating details from multiple distinct pages to answer a user query.

Comparison Baselines We compare MANGO with two prior methods: (1) AgentOccam (Yang et al., 2025), a web agent that refines the action and observation space to better align with the LLM's capabilities; and (2) WebWalker (Wu et al., 2025), a web agent that adopts an explore–critic paradigm. These two baselines are the best-performing methods on the WebVoyager and WebWalkerQA benchmarks, respectively.

Base LLMs We experiment with five different LLMs as the backbone of MANGO, including the GPT-5-mini model and Qwen3-{4, 8, 14, 32}B models. We access all models with their official API. For the GPT-5-mini model, we use the default parameter settings of the OpenAI API. For the Qwen3 models, we disable the thinking mode and set the temperature to 0.7 and top_p to 0.8 following the official best-practice guideline.

Implementation Details In our experiments, we limit both the navigation budget b and Thompson sampling iterations of MANGO to 10. We run all experiments in a single run.

4.2 Results and Analysis

Method	Qwen3 4B	Qwen3 8B	Qwen3 14B	Qwen3 32B	GPT-5 mini
WebWalker	14.73	12.40	12.40	14.73	16.28
AgentOccam	22.48	20.93	25.58	34.11	56.25
MANGO	**25.58**	**27.13**	**30.23**	**37.98**	**63.57**

Table 1: Comparison of the success rate (SR) of MANGO with baselines on WebVoyager.

In this section, we present a comprehensive analysis of MANGO's performance and efficiency on the WebVoyager and WebWalkerQA benchmarks.

Success Rate Table 1 shows the success rate of MANGO compared with WebWalker and AgentOccam on the WebVoyager benchmark. MANGO achieves the best performance across all settings. For instance, with GPT-5-mini as the backbone, MANGO achieves a 63.6% success rate, surpassing AgentOccam with a 7.3% absolute improvement and WebWalker with a 47.3% absolute improvement.

Table 2 presents the results on the WebWalkerQA benchmark. MANGO consistently outperforms WebWalker and AgentOccam in terms of overall success rate across all backbone models. For example, with the GPT-5-mini backbone, MANGO achieves an overall success rate of 52.5%, substantially outperforming WebWalker (25.7%) and AgentOccam (20.3%). Comparing the performance across different difficulty levels, we observe that MANGO exhibits better robustness than both baselines. While the success rate naturally declines

Model	Method	Single-source QA				Multi-source QA				Overall
		Easy	Medium	Hard	Overall	Easy	Medium	Hard	Overall	
Open-Sourced LLMs										
Qwen3-4B	WebWalker	28.75	16.43	5.83	15.59	16.25	10.00	4.17	9.41	12.50
	AgentOccam	7.50	2.86	1.67	3.53	11.25	4.29	3.33	5.59	4.56
	MANGO	21.25	26.43	17.50	22.06	11.25	12.14	12.50	12.06	**17.06**
Qwen3-8B	WebWalker	15.00	18.57	13.33	15.88	17.50	8.57	5.00	9.41	12.65
	AgentOccam	8.75	4.29	1.67	4.41	10.00	5.00	2.50	5.29	4.85
	MANGO	26.25	31.43	25.83	28.24	16.25	15.71	15.00	15.59	**21.91**
Qwen3-14B	WebWalker	35.00	20.71	10.83	20.59	15.00	7.86	5.83	8.82	14.71
	AgentOccam	11.25	5.00	1.67	5.29	8.75	5.71	8.33	7.35	6.32
	MANGO	35.00	37.14	25.83	32.65	20.00	19.29	19.17	19.41	**26.03**
Qwen3-32B	WebWalker	35.00	22.14	14.17	22.35	13.75	15.00	5.00	11.18	16.76
	AgentOccam	11.25	10.00	5.00	8.53	16.25	11.43	11.67	12.65	10.59
	MANGO	41.25	40.71	25.00	35.29	25.00	22.14	18.33	21.47	**28.38**
Closed-Sourced LLMs										
GPT-5-mini	WebWalker	35.00	32.86	21.67	29.41	33.75	22.14	14.17	22.06	25.74
	AgentOccam	26.25	20.00	13.33	19.12	30.00	18.57	19.17	21.47	20.29
	MANGO	63.75	64.29	54.17	60.59	43.75	50.71	37.50	44.41	**52.50**

Table 2: Comparison of the success rate (SR) of MANGO with baselines on WebWalkerQA.

Method	Qwen3				GPT-5 mini
	4B	8B	14B	32B	
WebVoyager					
WebWalker	9.11	7.00	7.44	9.32	7.38
AgentOccam	5.34	6.48	4.55	6.07	9.46
MANGO	6.21	7.83	6.26	5.35	14.18
WebWalkerQA					
WebWalker	8.71	9.62	7.92	8.62	10.38
AgentOccam	12.84	6.88	7.65	14.33	10.09
MANGO	5.92	10.89	7.75	7.49	19.13

Table 3: Action count comparison on WebVoyager and WebWalkerQA.

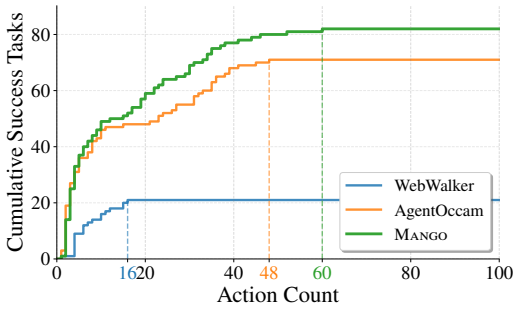
as the task difficulty level increases, MANGO maintains a significant lead over baselines on the most challenging queries.

Furthermore, MANGO demonstrates robust capabilities across both single-source QA and multi-source QA tasks. On single-source QA tasks, MANGO achieves a 31.2% absolute improvement over the best baseline when using GPT-5-mini. The performance gain is even more pronounced on the more challenging multi-source QA tasks—MANGO doubles the performance of both AgentOccam and WebWalker when using GPT-5-mini.

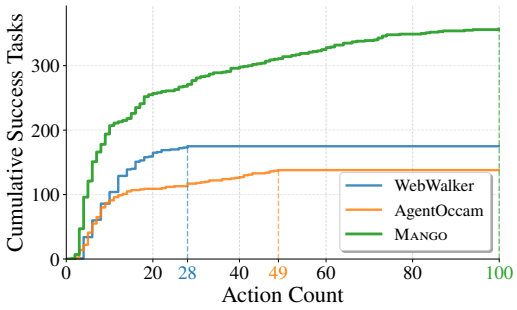
We also analyze the impact of the backbone model size using the Qwen3 family. As shown in Table 2, the performance of MANGO scales monotonically with model size, improving from 17.1% with Qwen3-4B to 28.4% with Qwen3-32B.

Action Count We count the total number of actions performed by MANGO and the baseline methods for each task to measure their efficiency. Table 3 reports the average action count of different methods on WebVoyager and WebWalkerQA. When using Qwen3 models, MANGO requires competitive or lower action counts while consistently obtaining higher success rates. Yet when using GPT-5-mini, MANGO takes more actions than both baselines on both benchmarks. An in-depth analysis reveals that this increase of actions is because MANGO solves more complex, long-horizon tasks when using GPT-5-mini. As illustrated in Figure 2, WebWalker and AgentOccam plateau early on both benchmarks. Specifically, on WebWalkerQA (Figure 2b), they fail to complete any more tasks after 28 and 49 action counts, respectively. In contrast, MANGO continues to accumulate successes well beyond 50 to 100 actions. This suggests that the higher average action count is driven by MANGO's ability to persevere and succeed on challenging tasks that require more actions to complete.

Finally, we analyze the impact of model size on the number of actions within the Qwen3 family. As shown in Table 3, the average action count remains relatively stable despite the significant improvement in success rates (e.g., from 17.1% with Qwen3-4B to 28.4% with Qwen3-32B on WebWalkerQA). This indicates that larger models in the Qwen3 family improve performance primarily through more accurate decision-making rather than by simply extending the exploration depth, thereby maintaining navigation efficiency even as



(a) WebVoyager



(b) WebWalkerQA

Figure 2: Cumulative number of successful tasks relative to the action count using GPT-5-mini on WebVoyager (Figure 2a) and WebWalkerQA (Figure 2b).

their capability to handle complex tasks increases.

4.3 Ablation Study

Method	Qwen3 4B	Qwen3 8B	Qwen3 14B	Qwen3 32B	GPT-5 mini
WebVoyager					
MANGO $_{\text{random}}$	17.83	20.93	21.71	27.13	56.59
MANGO $_{\text{google}}$	20.16	20.93	24.03	32.56	59.69
MANGO $_{\text{MCTS}}$	18.60	19.38	20.93	23.26	46.51
MANGO	**25.58**	**27.13**	**30.23**	**37.98**	**63.57**
WebWalkerQA					
MANGO $_{\text{random}}$	10.15	13.97	16.47	19.85	47.50
MANGO $_{\text{google}}$	15.59	18.38	23.09	25.88	49.41
MANGO $_{\text{MCTS}}$	11.91	13.53	15.74	16.47	42.21
MANGO	**17.06**	**21.91**	**26.03**	**28.38**	**52.50**

Table 4: Comparison of success rates (SR) between MANGO and three variants on WebVoyager and WebWalkerQA.

To validate the effectiveness of each component in MANGO, we conduct an ablation study by comparing MANGO with three variants.

We first evaluate the impact of the global structure analysis component by modifying the candidate URL set generation strategy with two variants: MANGO$_{\text{random}}$ and MANGO$_{\text{google}}$. MANGO$_{\text{random}}$ is a simple method that randomly selects a candidate URL set from the entire set of reachable pages collected during web crawling. MANGO$_{\text{google}}$ constructs the candidate set $\mathcal{U}$ solely using URLs retrieved by Google Search.

Second, we assess the contribution of our URL prioritization and selection algorithm by comparing MANGO with an agentic search strategy, Monte Carlo Tree Search (MCTS), as introduced in Section 2.3. We implement this variant inspired by both LATS (Zhou et al., 2024a), a general agent leveraging MCTS, and WebPilot (Zhang et al., 2024), a web agent leveraging a variant of MCTS. Since the implementation of WebPilot is not released publicly, we adopt the one-step simulation concept from it and retain the standard MCTS search for agents following LATS. We connect the MCTS navigation agent after our Global Structure Analysis component, initializing the state using the candidate URL set $\mathcal{U}$ and performing MCTS to navigate the pages. We describe the implementation details in Appendix A.2.

As shown in Table 4, MANGO consistently outperforms all variants across both benchmarks. When comparing URL selection strategies, MANGO$_{\text{random}}$ exhibits lower performance (e.g., 56.59% SR on WebVoyager with GPT-5-mini) compared to MANGO. While MANGO$_{\text{google}}$ performs better than the random baseline, it still lags behind the full method. This demonstrates that relying solely on Google Search is insufficient.

Furthermore, the comparison with MANGO$_{\text{MCTS}}$ reveals the superiority of our Thompson Sampling approach for budget-constrained web navigation. MANGO significantly outperforms MANGO$_{\text{MCTS}}$ across all settings. For instance, on WebVoyager with the GPT-5-mini backbone, MANGO achieves a success rate of 63.57% compared to 46.51% for MANGO$_{\text{MCTS}}$. This is primarily because MCTS requires a large number of interaction steps to expand the search tree and estimate state values, which becomes impractical under strict budget constraints. In contrast, Thompson Sampling can rapidly balance exploration and exploitation without simulation, allowing the agent to allocate its limited navigation budget more efficiently toward the most promising candidate URLs.

We further justify our design choices of the relevance scoring mechanism, the URL prioritization strategy, and the episodic memory component through additional ablation studies in Appendix B.

4.4 Sensitivity Analysis

We analyze the sensitivity of MANGO to five key hyperparameters: the navigation budget per URL

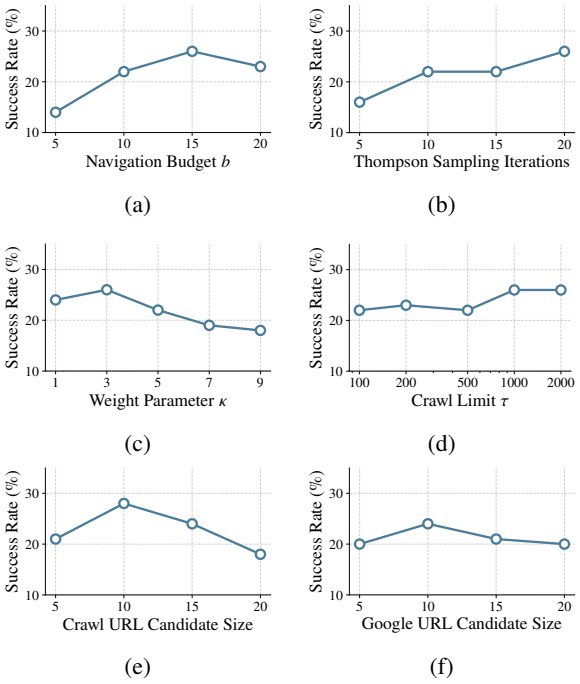


Figure 3: Sensitivity analysis of MANGO regarding key hyperparameters: the navigation budget per URL selection b, the number of Thompson Sampling iterations, the weight parameter κ, the crawl limit τ, and the candidate set sizes of Crawling and Google Search.

selection b, the number of Thompson Sampling iterations, the weight parameter κ, the crawl limit τ, and the candidate set size. We run MANGO in different budget settings on 100 tasks from WebWalkerQA using Qwen3-32B.

Navigation Budget b. As shown in Figure 3a, the success rate improves as b increases from 5 to 15. However, performance declines slightly when b is further increased to 20, suggesting that excessive actions on a single navigation attempt may cause the agent to confabulate or get lost in irrelevant paths.

Thompson Sampling Iterations. Figure 3b shows a positive correlation between the number of Thompson Sampling iterations and the success rate. This confirms that allocating more iterations allows the bandit to better balance exploration and exploitation across candidate URLs, leading to more effective navigation.

Weight Parameter κ. The parameter κ controls the strength of the initial BM25 relevance score when initializing the Beta distribution for Thompson Sampling. As shown in Figure 3c, the success rate peaks at 26% when κ=3. As κ increases to 7 and 9, performance drops to 19% and 18%, respectively. A high κ makes the initial prior too rigid, overpowering the reward signal from the reflection agent and preventing the bandit from dynamically adapting to navigation outcomes.

Crawl Limit τ. Figure 3d shows that performance steadily improves as τ increases from 100 (22% SR) to 1000 (26% SR). However, doubling τ to 2000 yields no further improvement. Therefore, τ=1000 serves as an optimal sweet spot, maximizing structural coverage without incurring the exponential time and computational costs associated with deeper crawling.

Candidate Set Size. Figures 3e and 3f show the effect of varying the number of candidates drawn from crawled URLs and Google search results, respectively. For both sources, Top-10 yields the best performance (28% and 24% SR). Since the navigation budget and Thompson Sampling iterations are capped at 10, an overly large combined candidate pool spreads the exploration budget too thin. Taking Top-10 from both sources thus strikes the ideal trade-off.

4.5 Failure Case Analysis

To understand the limitations of MANGO, we manually inspected 323 failure cases from the WebWalkerQA benchmark using GPT-5-mini as the backbone model. We identified five distinct error patterns:

- **Exceed Budget (52.4%):** This is the most common failure mode, where the agent exhausts the navigation budget before locating the target information. We identify two possible reasons for this issue. First, MANGO relies on lightweight crawling and search-based augmentation to construct a global view, which may provide incomplete coverage for very large websites with thousands of webpages. If the target information is buried deep in such websites, MANGO may not be able to find it with a limited budget. Second, the bandit-based URL selection depends on the quality of the initial candidate set, and errors in relevance estimation can result in suboptimal early choices that incur irreversible budget costs under strict action limits.
- **Locating Wrongly (24.6%):** This error occurs when the agent navigates to an incorrect

or irrelevant sub-page. MANGO may be misled by ambiguous links, resulting in navigation trajectories that deviate from the target information source.

- **Reasoning Error (15.4%):** In these cases, the agent successfully navigates to the correct page containing the target information but fails to generate the correct answer. For example, Task 140 asks for the student ID under a specific condition from the class of 2020, but MANGO outputs the student ID that satisfies the corresponding condition for the class of 2022. This indicates a failure in the underlying LLM's reading comprehension or reasoning capabilities, leading to hallucinated or incorrect extraction of details from the correct context.
- **Out-of-date Golden Answers (5.6%):** In a small portion of tasks in WebWalkerQA, the agent successfully retrieves the correct answer to the user query, but the answer is judged as incorrect because the golden answer is out-of-date. For example, Task 155 in WebWalkerQA asks how many fiscal years in a row Sony fulfilled the goal of using renewable electricity in China. MANGO navigates to the correct news page published in 2025 and answers "5 years" (2020-2024). However, since this benchmark was created in 2024, which was one year earlier than 2025, the golden answer was labeled as "4 years".
- **Reflection Error (2.0%):** The rarest error type involves the reflection agent incorrectly assessing the navigation state. Here, the reflection module prematurely classifies a partial or incorrect response as "adequate," causing the agent to terminate the navigation session before the user query is fully satisfied.

5 Conclusion

We propose MANGO, a web navigation framework that leverages global website structure to improve navigation efficiency under limited budgets. By identifying query-relevant entry points and modeling URL selection as a multi-armed bandit problem with Thompson Sampling to prioritize the URL, MANGO allocates exploration efforts more effectively. Experiments on WebVoyager and WebWalkerQA show that MANGO consistently outperforms best baselines across five backbone models.

6 Limitation

MANGO constructs a global view using lightweight crawling and search-based augmentation, which does not guarantee full coverage for large, dynamic, or deeply nested websites. In practice, many real-world websites (e.g., e-commerce platforms like Amazon) contain hundreds of thousands of pages, making exhaustive crawling both time-prohibitive and unnecessary. Therefore, MANGO deliberately constructs only a partial approximation of the site structure, focusing on representative and high-relevance pages. As a result, tasks with target information buried far down the hierarchy may still exceed the navigation budget. This indicates that partial structural visibility remains a bottleneck for long-horizon web navigation.

The bandit-based URL selection in MANGO depends on the quality of the initial candidate set generated during global structure analysis. Errors in relevance estimation or keyword generation can introduce suboptimal entry points, leading to early budget misallocation. While Thompson Sampling mitigates this over time, incorrect early decisions are costly under strict action limits.

Even when navigation succeeds, MANGO can fail due to reasoning errors, such as incorrect extraction or hallucinated details. These failures are orthogonal to navigation quality and cannot be addressed solely through improved exploration strategies. This limitation is shared by most current LLM-based web agents.

MANGO often achieves higher success rates by continuing exploration beyond where baseline agents plateau, which can increase action counts. While beneficial for solving long-horizon tasks, this behavior may be undesirable in latency-sensitive or cost-sensitive settings.

Acknowledgments

We sincerely thank the anonymous reviewers for their constructive feedback. This work was supported by NSF Grant ITE-2333736.

References

Tamer Abuelsaad, Deepak Akkil, Prasenjit Dey, Ashish Jagmohan, Aditya Vempaty, and Ravi Kokku. 2024. Agent-e: From autonomous web navigation to foundational design principles in agentic systems. *arXiv preprint arXiv:2407.13032*.

Saaket Agashe, Jiuzhou Han, Shuyu Gan, Jiachen Yang, Ang Li, and Xin Eric Wang. 2025. Agent s: An

open agentic framework that uses computers like a human. In *The Thirteenth International Conference on Learning Representations*.

Deepak Akkil, Tamer Abuelsaad, Prasenjit Dey, Ashish Jagmohan, Aditya Vempaty, and Ravi Kokku. 2024. Agent-e: From autonomous web navigation to foundational design principles in agentic systems. In *NeurIPS 2024 Workshop on Open-World Agents*.

Antonis Antoniades, Albert Örwall, Kexun Zhang, Yuxi Xie, Anirudh Goyal, and William Yang Wang. 2025. SWE-search: Enhancing software agents with monte carlo tree search and iterative refinement. In *The Thirteenth International Conference on Learning Representations*.

Deepayan Chakrabarti, Ravi Kumar, Filip Radlinski, and Eli Upfal. 2008. Mortal multi-armed bandits. In *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Bingzheng Gan, Yufan Zhao, Tianyi Zhang, Jing Huang, Li Yusu, Shu Xian Teo, Changwang Zhang, and Wei Shi. 2025. MASTER: A multi-agent system with LLM specialized MCTS. 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)*, pages 9409–9426, Albuquerque, New Mexico. Association for Computational Linguistics.

Yanchu Guan, Dong Wang, Zhixuan Chu, Shiyu Wang, Feiyue Ni, Ruihua Song, and Chenyi Zhuang. 2024. Intelligent agents with llm-based process automation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 5018–5027, New York, NY, USA. Association for Computing Machinery.

Izzeddin Gur, Hiroki Furuta, Austin V Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. 2024. A real-world webagent with planning, long context understanding, and program synthesis. In *The Twelfth International Conference on Learning Representations*.

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

Jaekyeom Kim, Dong-Ki Kim, Lajanugen Logeswaran, Sungryull Sohn, and Honglak Lee. 2024. Auto-intent: Automated intent discovery and self-exploration for large language model web agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16531–16541, Miami, Florida, USA. Association for Computational Linguistics.

Jing Yu Koh, Stephen Marcus McAleer, Daniel Fried, and Ruslan Salakhutdinov. 2025. Tree search for language model agents. *Transactions on Machine Learning Research*.

Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, and Jie Tang. 2024. Autowebglm: A large language model-based web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, page 5295–5306, New York, NY, USA. Association for Computing Machinery.

Tao Li, Gang Li, Zhiwei Deng, Bryan Wang, and Yang Li. 2023. A zero-shot language agent for computer control with structured reflection. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11261–11274, Singapore. Association for Computational Linguistics.

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

Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, and 1 others. 2025. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6140–6150.

Stephen Robertson, Hugo Zaragoza, and 1 others. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Ziluo Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, Ruyi An, Molei Qin, Chuqiao Zong, Longtao Zheng, YuJie Wu, Xiaoqiang Chai, Yifei Bi, Tianbao Xie, Pengjie Gu, and 9 others. 2025. Cradle: Empowering foundation agents towards general computer control. In *Forty-second International Conference on Machine Learning*.

Guanzhi Wang, Yuqi Zhang, Shuai Zhang, and 1 others. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. 2025. WebWalker: Benchmarking LLMs in web traversal. In

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10290–10305, Vienna, Austria. Association for Computational Linguistics.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. A-mem: Agentic memory for LLM agents. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Ke Yang, Yao Liu, Sapana Chaudhary, Rasool Fakoor, Pratik Chaudhari, George Karypis, and Huzefa Rangwala. 2025. Agentoccam: A simple yet strong baseline for LLM-based web agents. In *The Thirteenth International Conference on Learning Representations*.

Xiao Yu, Baolin Peng, Vineeth Vajipey, Hao Cheng, Michel Galley, Jianfeng Gao, and Zhou Yu. 2025. ExACT: Teaching AI agents to explore with reflective-MCTS and exploratory learning. In *The Thirteenth International Conference on Learning Representations*.

Yao Zhang, Zijian Ma, Yunpu Ma, and Volker Tresp. 2024. WebPilot: A Versatile and Autonomous Multi-Agent System for Web Task Execution with Strategic Exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. 2024. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *The Twelfth International Conference on Learning Representations*.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024a. Language agent tree search unifies reasoning acting and planning in language models.

Andy Zhou, Kai Yan, Michal Shlapentokh-Rothman, Haohan Wang, and Yu-Xiong Wang. 2024b. Language Agent Tree Search Unifies Reasoning Acting and Planning in Language Models. In *International Conference on Machine Learning*.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024c. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*.

A Implementation Details

A.1 Web Crawling

In this section, we describe our implementation details for the light-weight web crawling used in the global structure analysis component (Section 3.1). We implement the web crawling component using the open-source tool Crawl4AI [4]. We employ the `BFSDeepCrawlStrategy` to traverse the website structure and extract page content in markdown format. To ensure the process completes within a reasonable timeframe, we set the `max_pages` (i.e., τ in Section 3.1) parameter to 1000. Additionally, we set `include_external` to `False` and `exclude_all_images` to `True`. These constraints ensure the crawler remains focused on the target domain's hierarchy and filters out non-navigational assets such as image paths.

A.2 MANGO$_{\text{MCTS}}$

Following LATS (Zhou et al., 2024b), MANGO$_{\text{MCTS}}$ has the same six operations: selection, expansion, evaluation, simulation, backpropagation, and reflection. To avoid exceeding the context length of LLMs due to long page content, instead of expanding the currently selected node until a terminal state is reached as in LATS, we leverage the one-step simulation strategy from WebPilot. Specifically, we perform the action that has the highest score from the evaluation step and ask the LLM to generate a reflection based on the observation after performing this action. We set the maximum number of trajectories to sample, the number of times to prompt during expansion, and the number of times to prompt for state evaluation all to 10. As we set both the navigation budget b and the number of Thompson sampling iterations of MANGO to 10 (Section 4.1), we limit the maximum number of actions to 100 to ensure a fair comparison.

B Additional Experiments and Analyses

In this section, we present additional experiments and analyses to justify our design choices regarding relevance scoring, URL prioritization, and the episodic memory component. We conduct experiments on the same subset of the WebWalkerQA benchmark as Section 4.4, using GPT-5-mini as the backbone model.

B.1 Impact of Relevance Scoring Mechanisms

In the URL Prioritization and Selection component (Section 3.2), MANGO employs BM25 to score candidate URLs based on their relevance to the user query. While semantic embedding models could potentially capture richer contextual signals,

[4] https://github.com/unclecode/crawl4ai

Method	Single-source QA			Multi-source QA			Overall
	Easy	Medium	Hard	Easy	Medium	Hard	
MANGO$_{\text{Embedding}}$	50.0	77.8	53.3	75.0	44.0	44.4	53.0
MANGO$_{\text{Greedy}}$	25.0	72.2	66.7	75.0	40.0	44.4	51.0
MANGO$_{\text{No_Mem}}$	25.0	61.1	73.3	50.0	36.0	44.4	47.0
MANGO	50.0	72.2	60.0	75.0	48.0	44.4	**55.0**

Table 5: Additional ablation studies on MANGO, evaluated on a subset of WebWalkerQA with GPT-5-mini as the backbone. We compare MANGO with variants that (1) replace BM25 with semantic embeddings, (2) replace Thompson Sampling with a greedy ranking strategy, and (3) remove the episodic memory component.

computing embeddings for up to 1,000 crawled pages per task incurs substantial computational overhead. To validate our choice of BM25, we compare MANGO against a variant, MANGO$_{\text{Embedding}}$, which replaces BM25 with semantic similarity scores computed by the `KaLM-Embedding-V2.5` model[5]. As shown in Table 5, MANGO achieves a 55.0% overall success rate, slightly outperforming the embedding-based variant at 53.0%. We found that using a pre-trained embedding model sometimes struggled to distinguish page URLs that have subtle differences in wording, while BM25 is more sensitive to word differences. Nevertheless, fine-tuning an embedding model specifically for this task or performing semantic analysis on the page content instead of just the page URL would presumably lead to better performance, but would also lead to higher development cost or runtime cost.

B.2 Effectiveness of Thompson Sampling over Greedy Selection

To demonstrate that our MAB formulation provides meaningful benefits over simpler ranking approaches, we compare MANGO with a greedy variant, MANGO$_{\text{Greedy}}$, which ranks candidate URLs by their initial BM25 scores and visits them in descending order without dynamic updates from the reflection agent. As shown in Table 5, MANGO outperforms MANGO$_{\text{Greedy}}$ with a 4.0% absolute improvement in overall success rate. This confirms that Thompson Sampling effectively leverages feedback from the reflection agent to dynamically update the posterior distribution, enabling MANGO to adaptively prune unpromising URLs and reallocate the navigation budget toward more promising candidates.

[5] https://huggingface.co/KaLM-Embedding/KaLM-embedding-multilingual-mini-instruct-v2.5

B.3 Ablation on Episodic Memory

The episodic memory component in MANGO stores navigation trajectories and reflections, which are retrieved to inform subsequent navigation attempts on the same URL (Section 3.4). To isolate the contribution of this component, we evaluate a variant, MANGO$_{\text{No_Memory}}$, in which the episodic memory module is removed. As shown in Table 5, removing the episodic memory causes the overall success rate to drop from 55.0% to 47.0%, an 8.0% absolute decrease. This is consistent with our design motivation: without access to prior trajectories and reflections, the navigation agent is prone to repeating the same actions when revisiting a URL, leading to redundant exploration and premature budget exhaustion.

C Prompts

In this section, we present the full prompts of MANGO. Table 6 shows the prompt of generating keywords described in Section 3.1. Table 7 shows the prompt of the navigation agent (Section 3.3). Table 8 and 9 shows the prompts of the reflection agent (Section 3.4).

```
You are a search expert.

Transform the user's intent into a concise
search query composed of space-separated
keywords.
Output only the final query.
```

Table 6: Prompt for the Search Query Generator. It transforms complex user intents into concise keyword-based queries.

D Full Results

We report the full experimental results on WebVoyager, categorized by website type in Table 10. We also report the detailed results of the ablation study on WebVoyager and WebWalkerQA in Table 11 and Table 12, respectively.

```
You are a Web Navigation Agent.

You can call functions to visit websites as
needed.
You may also be provided with previous
navigation history, including function calls,
previous outputs, and reflections, to help
inform your decisions.
You may choose to continue navigating by
revisiting a previously visited URL or by
starting fresh from the root URL.
TASK INSTRUCTIONS
1. Use the browser functions to visit and
explore the target URL.
2. Read the page content thoroughly.
3. If you find new content that can answer the
user query, generate an answer based on the page
content. Otherwise, continue navigating to find
relevant information.

HANDOFF INSTRUCTIONS
Instead of outputting text, you must hand off
control to the appropriate reflection agent
based on your findings.

Case 1: Relevant Information Found
If you find new content that clearly answers the
user query:
   - Hand off to the success_reflection_agent.
   - Pass result and source (the specific URL)
to the handoff function.

Case 2: Stuck / Information Not Found
If you cannot find relevant information, reach a
dead end, determine that the page content is
entirely irrelevant, or cannot find new content
after thorough exploration:
   - Hand off to the failure_reflection_agent.
   - You do not need to provide content, but
ensure that you have explored the page
sufficiently.

TASK
User Query: {USER_QUERY}
Root URL: {ROOT_URL}
```

Table 7: Prompt for the Web Navigation Agent. This instruction guides the agent to explore pages using previous navigation history and hand off control to specific reflection agents based on success or failure.

```
You are a Navigation Decision Evaluator.

You are reviewing a navigation session in which
the agent has generated a response indicating
task completion. Your goal is to determine
whether the navigation actions and output fully
satisfy the user's query.

DECISION FRAMEWORK
A. adequate
   - The final output and navigation trajectory
provide a complete and comprehensive answer to
the User Query. The answer needs to cover all
questions of the User Query.
   - No further navigation is needed.

B. inadequate
   - The output is partial or relevant, but
does not fully answer the User Query.
   - Further navigation on following links is
likely to provide the missing information.

OUTPUT FORMAT
Output a JSON object.
{
   "status": "adequate" | "inadequate",
   "reason": "Explain why the current output is
sufficient or why we should continue.",
   "output": "The response generated by the
navigation agent.",
   "source": "The URL where the content was
extracted from."
}
```

Table 8: Prompt for the Reflection Agent (Task Completion Case). This prompt is triggered when the navigation agent indicates task completion to determine if the agent should stop or continue.

```
You are a Navigation Decision Evaluator.

The web navigation agent has exhausted its
navigation budget before completing the task.
Your goal is to analyze the navigation
trajectory and the final URL to decide if this
path is promising and should be continued, or if
it should be abandoned.

DECISION FRAMEWORK
A. feasible
   - The answer was not found yet, but the
current page is relevant to the User Query.
   - The stop might be due to budget
constraints, but the agent simply needs to visit
more links or navigate deeper on this site.
   - We should NOT give up on this path yet.

B. infeasible
   - The page is irrelevant, a dead end, or the
site is broken.
   - Repeated actions in the trajectory suggest
no answer exists here.
   - We should abandon this path.

OUTPUT FORMAT
Output a JSON object.

{
   "status": "feasible" | "infeasible",
   "reason": "Explain why the current trajectory
is promising vs. why it leads to a dead end."
}
```

Table 9: Prompt for the Reflection Agent (Budget Exhaustion Case). This prompt is triggered when the agent exhausts its navigation budget, helping determine if the current trajectory remains promising.

Website	Qwen3-4B			Qwen3-8B			Qwen3-14B			Qwen3-32B			GPT-5-mini		
	WW	AO	M	WW	AO	M	WW	AO	M	WW	AO	M	WW	AO	M
Allrecipes	25.0	0.0	100.0	50.0	25.0	25.0	25.0	0.0	50.0	25.0	0.0	75.0	25.0	75.0	75.0
Amazon	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0
Apple	14.3	0.0	28.6	14.3	14.3	14.3	28.6	14.3	14.3	28.6	28.6	28.6	28.6	57.1	57.1
ArXiv	18.8	18.8	12.5	18.8	25.0	18.8	18.8	25.0	25.0	12.5	37.5	37.5	12.5	53.3	56.2
BBC News	0.0	0.0	50.0	0.0	0.0	50.0	0.0	0.0	50.0	0.0	50.0	50.0	0.0	50.0	50.0
Booking	0.0	50.0	0.0	0.0	50.0	0.0	0.0	50.0	50.0	0.0	50.0	50.0	0.0	100.0	100.0
Cambridge Dictionary	22.2	0.0	44.4	22.2	0.0	55.6	22.2	0.0	66.7	22.2	0.0	88.9	44.4	11.1	88.9
Coursera	50.0	0.0	100.0	0.0	0.0	50.0	50.0	0.0	50.0	50.0	0.0	50.0	50.0	50.0	100.0
ESPN	30.0	20.0	30.0	30.0	10.0	40.0	40.0	30.0	50.0	20.0	40.0	50.0	40.0	50.0	70.0
Google Map	0.0	33.3	0.0	0.0	11.1	0.0	0.0	22.2	0.0	0.0	55.6	33.3	0.0	55.6	77.8
Google Search	0.0	0.0	37.5	0.0	6.2	50.0	0.0	12.5	50.0	6.2	12.5	50.0	0.0	56.2	68.8
Huggingface	11.8	23.5	17.6	5.9	5.9	23.5	11.8	17.6	11.8	17.6	17.6	11.8	17.6	41.2	41.2
Wolfram Alpha	17.6	47.1	17.6	11.8	47.1	20.6	2.9	50.0	23.5	14.7	58.8	26.5	11.8	73.5	61.8
Overall	14.7	22.5	**25.6**	12.4	20.9	**27.1**	12.4	25.6	**30.2**	14.7	34.1	**38.0**	16.3	56.2	**63.6**

Table 10: The full success rate (SR) results categorized by websites on WebVoyager. WW refers to WebWalker, AO refers to AgentOccam, and M refers to MANGO.

Website	Qwen3-4B				Qwen3-8B				Qwen3-14B				Qwen3-32B				GPT-5-mini			
	R	G	MCTS	M	R	G	MCTS	M	R	G	MCTS	M	R	G	MCTS	M	R	G	MCTS	M
Allrecipes	25.0	25.0	25.0	100.0	25.0	0.0	25.0	25.0	50.0	50.0	25.0	50.0	50.0	50.0	50.0	75.0	75.0	50.0	50.0	75.0
Amazon	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Apple	0.0	0.0	14.3	28.6	14.3	14.3	14.3	14.3	14.3	0.0	28.6	14.3	14.3	28.6	28.6	28.6	28.6	42.9	28.6	57.1
ArXiv	31.2	12.5	18.8	12.5	18.8	25.0	18.8	18.8	31.2	31.2	18.8	25.0	31.2	25.0	18.8	37.5	68.8	62.5	37.5	56.2
BBC News	50.0	50.0	0.0	50.0	0.0	0.0	0.0	50.0	0.0	0.0	0.0	50.0	0.0	50.0	0.0	50.0	100.0	50.0	0.0	50.0
Booking	0.0	0.0	50.0	0.0	50.0	50.0	50.0	0.0	0.0	0.0	50.0	50.0	0.0	50.0	50.0	50.0	100.0	100.0	50.0	100.0
Cambridge Dictionary	33.3	33.3	11.1	44.4	44.4	44.4	22.2	55.6	66.7	22.2	22.2	66.7	44.4	77.8	22.2	88.9	88.9	88.9	66.7	88.9
Coursera	50.0	100.0	0.0	100.0	50.0	50.0	0.0	50.0	50.0	0.0	0.0	50.0	50.0	100.0	0.0	50.0	100.0	100.0	100.0	100.0
ESPN	0.0	20.0	30.0	30.0	20.0	10.0	30.0	40.0	30.0	30.0	30.0	50.0	30.0	40.0	30.0	50.0	70.0	70.0	50.0	70.0
Google Map	0.0	0.0	0.0	0.0	11.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	44.4	0.0	33.3	55.6	66.7	55.6	77.8
Google Search	25.0	37.5	25.0	37.5	18.8	37.5	25.0	50.0	12.5	43.8	25.0	50.0	18.8	31.2	25.0	50.0	18.8	50.0	43.8	68.8
Huggingface	11.8	17.6	23.5	17.6	5.9	11.8	23.5	23.5	11.8	29.4	23.5	11.8	29.4	23.5	35.3	11.8	41.2	52.9	58.8	41.2
Wolfram Alpha	17.6	17.6	17.6	17.6	26.5	20.6	17.6	20.6	17.6	20.6	20.6	23.5	29.4	17.6	20.6	26.5	61.8	55.9	41.2	61.8
Overall	17.8	20.2	18.6	**25.6**	20.9	20.9	19.4	**27.1**	21.7	24.0	20.9	**30.2**	27.1	32.6	23.3	**38.0**	56.6	59.7	46.5	**63.6**

Table 11: Full comparison of the success rate (SR) of MANGO with its four variants on WebVoyager. R refers to $\text{MANGO}_{\text{Random}}$, G refers to $\text{MANGO}_{\text{Google}}$, MCTS refers to $\text{MANGO}_{\text{MCTS}}$, and M refers to MANGO.

Model	Method	Single-source QA				Multi-source QA				Overall
		Easy	Medium	Hard	Overall	Easy	Medium	Hard	Overall	
Open-Sourced LLMs										
Qwen3-4B	MANGO $_{\text{random}}$	18.75	10.71	7.50	11.47	13.75	7.14	7.50	8.82	10.15
	MANGO $_{\text{google}}$	16.25	25.00	10.83	17.94	12.50	12.86	14.17	13.24	15.59
	MANGO $_{\text{MCTS}}$	15.00	10.71	12.50	12.35	7.50	14.29	10.83	11.47	11.91
	MANGO	21.25	26.43	17.50	22.06	11.25	12.14	12.50	12.06	**17.06**
Qwen3-8B	MANGO $_{\text{random}}$	22.50	15.71	10.00	15.29	20.00	12.14	8.33	12.65	13.97
	MANGO $_{\text{google}}$	23.75	25.71	16.67	22.06	21.25	14.29	10.83	14.71	18.38
	MANGO $_{\text{MCTS}}$	17.50	12.86	12.50	13.82	10.00	16.43	11.67	13.24	13.53
	MANGO	26.25	31.43	25.83	28.24	16.25	15.71	15.00	15.59	**21.91**
Qwen3-14B	MANGO $_{\text{random}}$	27.50	25.71	10.00	20.59	21.25	11.43	10.83	13.53	17.06
	MANGO $_{\text{google}}$	32.50	35.71	17.50	28.53	18.75	20.71	13.33	17.65	23.09
	MANGO $_{\text{MCTS}}$	18.75	13.57	17.50	16.18	11.25	20.00	12.50	15.29	15.74
	MANGO	35.00	37.14	25.83	32.65	20.00	19.29	19.17	19.41	**26.03**
Qwen3-32B	MANGO $_{\text{random}}$	28.75	22.14	19.17	22.65	21.25	16.43	17.50	17.94	20.29
	MANGO $_{\text{google}}$	35.00	41.43	20.83	32.65	22.50	21.43	14.17	19.12	25.88
	MANGO $_{\text{MCTS}}$	18.75	14.29	18.33	16.76	11.25	22.14	12.50	16.18	16.47
	MANGO	41.25	40.71	25.00	35.29	25.00	22.14	18.33	21.47	**28.38**
Closed-Sourced LLMs										
GPT-5-mini	MANGO $_{\text{random}}$	57.50	60.71	46.67	55.00	45.00	38.57	38.33	40.00	47.50
	MANGO $_{\text{google}}$	53.75	60.00	50.83	55.29	46.25	40.00	45.83	43.53	49.41
	MANGO $_{\text{MCTS}}$	50.00	41.43	44.17	44.41	40.00	41.43	38.33	40.00	42.21
	MANGO	63.75	64.29	54.17	60.59	43.75	50.71	37.50	44.41	**52.50**

Table 12: Full comparison of the success rate (SR) of MANGO with four variants on WebWalkerQA.