Cook Browser: Completing Work on the Live Web

Austin Worthington
Client Acquisition.io

trycook.ai/research

Abstract

We evaluated Cook Browser through the Cook desktop application on three browser-agent suites: Online-Mind2Web, BU Bench V1, and Odysseys. The accepted recorded results are 93.4%, 94.4%, and 86.25%, respectively, after the cohort adjustments described in this report. The work tested whether an agent can navigate live websites, retain the requested target, and return the result of a multi-step task. (Cook, 2026)

The research also produced concrete engineering improvements: stronger long-horizon persistence, explicit tab lifecycle management, correct final-answer extraction, and preservation of rich browser results. These mechanisms address failures in executing work, rather than only changes to the wording of the final response.

1 Introduction and scope

Cook completed 268 tasks in the recorded 300-task Online-Mind2Web cohort. Its adjusted score excludes 13 tasks classified as impossible; its full-cohort score is 89.3%. The accepted result merges a baseline with targeted reruns. It is a best-known recorded result, not a fresh single-pass estimate or an all-benchmark state-of-the-art claim.

Report scope: a retrospective of existing evaluation records. No new benchmark execution was performed for this publication. External results are attributed to their publishers and presented as context.

2 Architecture: closing the loop between intent and evidence

A browser agent works in a changing environment. A page can load slowly, a new tab can become active, or a button can produce an intermediate state rather than the requested result. The useful unit of work is therefore a continuing relationship between a person's intent, the current page, and evidence of progress. Our architectural emphasis is on preserving that relationship across a sequence of actions.

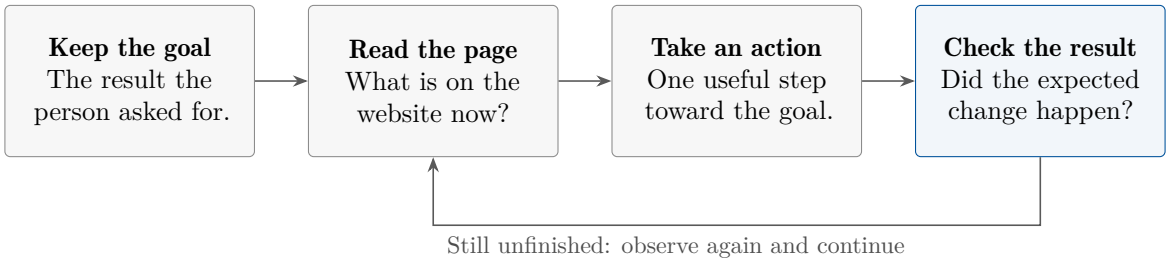


Figure 1: The browser task as a feedback loop. In plain English: keep the request in mind, look, act, and check. The diagram summarizes the architectural idea; it does not expose the execution policy.

2.1 Keep the intended target attached to the action

A browser can contain several similar pages at once. Selecting the right action is not enough if that action reaches the wrong tab or page. Requested-target binding makes the target part of the task's execution context, rather than leaving it as an assumption about whichever page happens to be active. The recorded benchmark work includes an explicit correction to this behavior. (Cook, 2026)

For a nontechnical analogy, imagine asking an assistant to update a particular customer's record while several customer records are open. The instruction "change the address" is only useful when the assistant also retains *whose* address is being changed. The browser equivalent is keeping the intended page associated with the request. This example illustrates the mechanism, not a separately scored benchmark task.

2.2 Let the work determine when a task is finished

Long tasks often contain legitimate stretches of investigation: comparing pages, following links, waiting for a state change, or collecting several pieces of evidence. Stopping because an interaction has become long can leave a task incomplete even when its remaining steps are feasible. One of the recorded improvements addressed premature winding down and increased persistence on long-horizon work. (Cook, 2026)

Persistence is useful when it remains tied to the original objective. Repeating an ineffective action indefinitely would merely turn a failure into a slower failure. Figure 1 expresses the design intent: another action should follow from an updated view of the task, and completion should be supported by the observed result. The benchmark record supports the importance of persistence as an engineering target; it does not isolate how many percentage points that change contributed.

2.3 Preserve what the browser actually returned

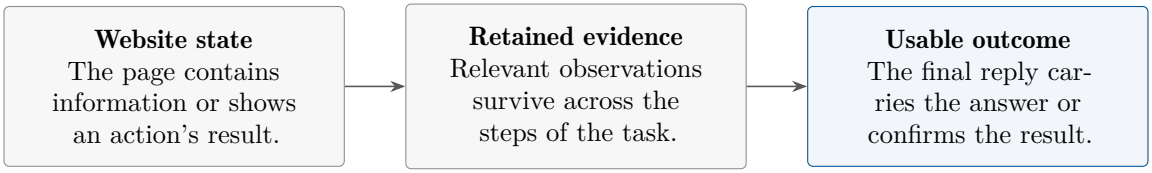
Browser observations are not always plain sentences. They can include structured page information, screenshots, and the result of an interaction. If this material is flattened, lost, or associated with the wrong step, later reasoning operates on an incomplete account of what happened. The research included fixes that preserved rich browser results and images throughout the task trail. (Cook, 2026)

The practical distinction is between *being told that a page changed* and *retaining the evidence of the change*. Evidence makes it possible to revisit a decision, understand an unexpected transition, and evaluate an action after the fact. A successful action whose result disappears between system components can be as unhelpful to the final answer as an action that never succeeded.

2.4 Treat browser resources as part of task reliability

Open pages consume resources and remain part of the agent's working environment. Unmanaged tabs can accumulate across tasks, increasing memory pressure and making subsequent behavior harder to interpret. The recorded work added explicit lifecycle management and cleanup for agent-created browser tabs. The relevant principle is ownership: the system should know which browser resources belong to its work and release those resources when appropriate. (Cook, 2026)

This is a systems contribution rather than a new reasoning capability. It does not teach the model a new fact. It helps preserve the conditions in which the model can use the capabilities it already has. A repeated evaluation is particularly sensitive to this distinction, because one task's leftover state can otherwise affect the next task.



A failure in any link can hide work that succeeded elsewhere. Better reasoning alone cannot repair missing evidence.

Figure 2: Why evidence preservation and final-answer handling matter. Completing an action, retaining its evidence, and delivering its result are distinct responsibilities.

2.5 Deliver the result, not just a trace of activity

A long task trail can contain intermediate findings, progress messages, and several candidate answers. The research corrected final-answer handling and preserved explicit answers during publishing workflows. These changes matter because the useful output is the answer or completed state requested by the person, not simply a record that tools were used. (Cook, 2026)

Together, the design choices address different failure points: target confusion, early termination, loss of observations, resource accumulation, and loss of the final result. They explain *how* a capable model can be made more effective inside a browser without assuming that the model alone is responsible for every change in the final score.

3 Evaluation Results

Benchmark	Accepted adjusted result	Full / strict result
Online-Mind2Web	268 / 287 = 93.4%	268 / 300 = 89.3%
BU Bench V1	17 / 18 = 94.4%	17 / 20 = 85.0%
Odysseys	34.5 / 40 = 86.25%	39 / 47 = 82.98%

Table 1: Cook Browser recorded results. Task and rubric denominators are distinct.

3.1 Online-Mind2Web

The cohort contains 300 tasks. The accepted merge records 268 passes, 19 failures, and 13 impossible tasks. Within the adjusted cohort, performance was 76/77 on easy tasks (98.7%), 125/136 on medium tasks (91.9%), and 67/74 on hard tasks (90.5%). The benchmark evaluates task execution on live websites. (Cook, 2026; Xue et al., 2025)

3.2 BU Bench V1

Cook ran a frozen 20-task subset. Two frozen reference answers were excluded after live-page evidence showed stale or incorrect expected values: a movie-chart overlap count and a crocodile-record count. Seventeen of the remaining 18 tasks passed. This subset is smaller than the public 100-task BU Bench V1 suite. (Cook, 2026; Browser Use, 2026)

3.3 Odysseys

Cook scored rubric points, allowing partial progress. Five tasks were excluded for unavailable future data, geographic restrictions, or anti-bot interruptions. The recorded adjusted total is 34.5/40; the strict record is 39/47. These totals are separate scoring views and must not be treated as binary task-success counts. (Cook, 2026; Jang et al., 2026)

Arithmetic: adjusted scores divide accepted points or passes by the eligible denominator. Full-cohort results keep the original task denominator; the strict Odysseys result uses the recorded

rubric total. Percentages are rounded only for display.

4 Experimental Methodology

4.1 Execution path

Tasks were executed through the real Cook desktop browser. Evaluation considered the browser trajectory, available observations, final answer, and the relevant task rubric. Recording the path matters because a short final response may omit a page state that the browser evidence establishes, while a confident response alone may not establish that an action occurred. (Cook, 2026)

This report describes the system's design principles and recorded evaluation outcomes. Internal prompts, action-selection policies, recovery rules, resource budgets, and implementation details are outside its scope.

4.2 Models and accepted results

BU Bench and Odysseys used GPT-5.6 Sol with high reasoning for the actor and the accepted automated judge. Using the same model family for both is a methodological limitation, rather than an independent human evaluation. Online-Mind2Web combined a Claude Sonnet 5 baseline with targeted Opus 5 reruns; the accepted result is a merged record, not a single uninterrupted run. These model and evaluation details describe the historical measurement, not a specification of the current product's serving configuration. (Cook, 2026)

4.3 Comparison with Aside

Aside publishes 99.0% on 300 Online-Mind2Web tasks, 93.0% on 100 BU Bench tasks, and 88.8% rubric-item success on 200 Odysseys tasks. Its Odysseys task-average rubric score is 86.5%; these two aggregations differ. Cook and Aside did not run in one controlled evaluation with identical actors, graders, cohorts, and website conditions. (Aside, 2026)

The research page therefore identifies the external bars as published references. Cook's smaller BU cohort, its Mind2Web reruns and exclusions, and the Odysseys denominator differences prevent a direct all-suite ranking. All chart values and notes are downloadable at trycook.ai/research/results.json.

5 Discussion and Limitations

5.1 Engineering findings

The benchmark work led to long-horizon persistence improvements, model-pin controls, tab cleanup, correct final-answer extraction, requested-target binding, preservation of explicit answers during publishing, and preservation of rich REPL results and images. The accepted record connects these changes to browser execution; it does not isolate a numerical contribution for each fix. The architecture discussion provides a mechanism-based explanation, while the tables provide aggregate outcome evidence. A controlled ablation would be required to turn each proposed mechanism into an individual causal score claim. (Cook, 2026)

5.2 Limitations and future work

Nineteen Online-Mind2Web failures remain; twelve are concentrated on apartments.com and petfinder.com. BU Bench retains a factual error on a sports-statistics task. Anti-bot interruptions, changing inventory, and geographic restrictions remain material to live-web evaluation. A fresh full-suite run with one fixed configuration is needed to estimate repeatability.

Next research step: pin complete manifests, actors, graders, budgets, and scoring rules; evaluate fresh runs and report every attempt. Keep development reruns distinct from confirmation runs, and publish full-cohort and adjusted scores together.

References

Cook. Accepted browser results and runbook, August 5, 2026. Cook Browser evaluation record and engineering research notes. Public chart summary: https://trycook.ai/research/results.json.

Xue et al. Online-Mind2Web benchmark. https://github.com/OSU-NLP-Group/Online-Mind2Web.

Browser Use. BU Bench V1. https://github.com/browser-use/benchmark.

Jang et al. Odysseys: Benchmarking Web Agents on Realistic Long Horizon Tasks. https://arxiv.org/abs/2604.24964.

Aside. Published benchmark results. Accessed September 5, 2026. https://github.com/at-inc/aside-benchmarks.

Data availability. Accepted aggregates and scoring notes are reproduced here. Full Cook trajectories are retained internally; this paper is not a public per-task reproduction package.