

Long-Horizon-Terminal-Bench 2.0: Automating Diverse Task Construction and Measuring Sustained Agent Progress

Zongxia Li^{†1,2}, Zhongzhi Li^{†1,3}, Yucheng Shi^{†1}, Ruhan Wang^{1,4}, Junyao Yang^{1,5}, Chengsong Huang⁶, Zhichao Liu², Xiyang Wu², Anhao Li⁷, Yue Yu⁴, Ergan Shang⁸, Fuxiao Liu², Tong Zheng², Ninghao Liu⁹, Lichao Sun¹⁰, Haitao Mi¹, LeoweiLiang¹

¹Tencent HY LLM Frontier ²University of Maryland, College Park ³University of Georgia ⁴Indiana University ⁵National University of Singapore ⁶Washington University in St. Louis ⁷University of Minnesota, Twin Cities ⁸Carnegie Mellon University ⁹The Hong Kong Polytechnic University ¹⁰Lehigh University

[†] Core contribution.

zli12321@umd.edu, zongxiali@global.tencent.com, zl22754@uga.edu

Long-horizon terminal benchmarks measure whether an agent can sustain progress through extended workflows with process rewards, not only whether its final state passes a binary check. We present LONG-HORIZON-TERMINAL-BENCH 2.0 (LHTB 2.0), an extension of LONG-HORIZON-TERMINAL-BENCH that preserves its 46 original tasks while correcting task and evaluation-infrastructure defects and extending 31 tasks. LHTB 2.0 contains 77 tasks across 13 domains and eight task types. Different from prior benchmark, the extended tasks are produced through a systematic reconstruction-task pipeline. We start from a working scientific or engineering artifact, then use a parameterized reference implementation to generate multiple scenarios, reveal six as behavioral evidence, hold out three for evaluation, and use dense progress ladders to reward partial progress while preventing memorization of public outputs. We also extend the evaluation horizon from 90 minutes to three hours and record verifier checkpoints every 30 minutes, enabling direct measurement of sustained progress within a run. Across 1,145 scored model-task runs from 15 frontier models, GPT-6-astra leads with dense mean reward 0.69 and 37 tasks at $R \geq 0.95$, followed by GPT-5.6-sol with dense mean reward 0.67 and 32 tasks at the same threshold. On the 30 reconstruction tasks, binary verifier scores average 0.221, whereas dense ladder scores average 0.653, showing that outcome-only evaluation discards substantial intermediate progress. LHTB 2.0 provides a broader, hardened, and process-aware testbed for studying reliable agent execution over hours.

Project page: <https://zli12321.github.io/LHTB/>

Date: October 14, 2026

1 Introduction

AI agents built on large language models are increasingly expected to operate beyond short, self-contained interactions: they must use tools, maintain state, recover from errors, and coordinate many dependent steps in workflows that last minutes or hours. This shift creates two related evaluation challenges. First, a final pass/fail outcome reveals whether an agent completed a task, but not whether it made sustained progress, stalled, or regressed over time. Second, as agent capabilities expand into new domains, manually constructing realistic and reliable long-horizon tasks becomes increasingly difficult to scale, and also increasingly difficult for humans to annotate, but easy to measure agents. Evaluating the next generation of agents therefore requires both scalable task construction and process-aware measurements that track progress throughout extended trajectories.

LONG-HORIZON-TERMINAL-BENCH (LHTB) benchmark introduced 46 containerized tasks with dense, subtask-

based grading for these workflows [15]. It established that agents can make meaningful partial progress even when they do not finish, but it also exposed several limitations. First, manually curating long-horizon tasks is expensive and difficult to scale across domains [3, 23]. Second, most existing benchmarks score only the final outcome of each run, hiding whether an agent makes sustained progress, plateaus, or regresses over time [12]. Third, repeated evaluation revealed task-environment and verifier-integration defects that must be corrected before larger evaluations can be compared reliably.

We propose LONG-HORIZON-TERMINAL-BENCH 2.0 (LHTB 2.0) to address the limitations of the previous benchmark version. The benchmark retains the 46 original tasks, fixes task and harness defects, and adds 31 tasks, a total of 77-task suite spanning 13 domains and eight task types. The evaluation infrastructure now isolates checkpoint grading from the active agent, clears verifier artifacts between phases, audits successful trajectories for grader access, and retains only clean attempts. We also extend the experiment run to three-hour trials with verifier checkpoints every 30 minutes make it possible to measure how reward evolves within the same trajectory rather than comparing unrelated short and long runs.

The central technique for automating long-horizon and difficult task generation is through task reconstruction [13, 14]. Starting from a known working, we identify parameter axes, execute a reference oracle over a scenario, publish multiple input-output examples as behavioral evidence, and withhold multiple scenarios for grading. The agent receives the tools and evidence needed to infer the behavior, but not the reference implementation. This procedure contributes 30 new reconstruction tasks in science domains such as geospatial analysis, document processing, life sciences, chemistry and materials, physical simulation, signals and control, security, and related domains.

Our evaluation covers 1,145 scored model-task cells across 15 frontier models. On a dense all-77 basis, GPT-6-astra leads with mean reward 0.69 and 37 tasks at $R \geq 0.95$, followed by GPT-5.6-sol with mean reward 0.67 and 32 tasks at the same threshold. GPT-6-astra’s lead comes almost entirely from the reconstruction split; on the released workflows GPT-5.6-sol is slightly ahead. More importantly, the 30 reconstruction tasks average only 0.221 under binary verifier scoring but 0.653 under dense ladder scoring. The gap shows that binary evaluation frequently labels hours of useful progress as total failure. The checkpoints further show that 42% of solves arrive after the first hour, that a one-hour budget would reorder the middle of the leaderboard, and that the fastest models often claim completion long before they finish.

The new benchmark extension includes:

- **A broader and corrected long-horizon benchmark.** We extend LONG-HORIZON-TERMINAL-BENCH from 46 to 77 tasks, covering 13 domains and eight task types while preserving the original suite.
- **Verifying the feasibility of a systematic task-construction pipeline.** We turn parameterized real-world artifacts into reconstruction tasks with public behavioral evidence, hidden scenarios, anti-memorization checks, and dense progress ladders.
- **Process-aware long-horizon progress evaluation.** We increase the budget from 90 minutes to three hours and record 30-minute checkpoints, enabling direct analysis of sustained progress, late completion, and regression.
- **A hardened evaluation protocol.** We fix task and harness bugs, isolate verifier execution, audit grader access, and replace affected results with clean reruns.

2 From LHTB to LHTB 2.0

2.1 Task formulation

LONG-HORIZON-TERMINAL-BENCH 2.0 follows the Terminal-Bench and Harbor formulation [18]. Each task contains (1) a natural-language instruction, (2) a containerized terminal environment, (3) task and runtime configuration, and (4) a deterministic verifier or simulator. The instruction is the only specification shown to the agent. All relevant code, data, tools, and assets are available through the terminal, and the agent may

inspect files, edit programs, run commands, and evaluate intermediate outputs until it succeeds or reaches its time budget.

The benchmark is long-horizon by construction. A task requires continued work over many dependent steps rather than a single patch or answer: reproducing an experiment, repairing a scientific analysis pipeline, optimizing a system, solving a campaign of game levels, or inferring the behavior of an unknown artifact. Errors made early can propagate through later stages, and apparent completion must be checked against an end-to-end objective.

2.2 Benchmark extension and splits

LHTB 2.0 contains all 46 tasks from LONG-HORIZON-TERMINAL-BENCH and 31 additions. We preserve the original task identities and descriptions so that the extension remains traceable to the first benchmark, while repairing environment and verifier integration defects discovered during larger evaluations. The 77 tasks are organized into two complementary splits:

- **Released-workflow split (46 tasks).** The 46 original tasks and one additional optimization task cover audit and repair, optimization, gameplay, experiment reproduction, constraint puzzles, professional workflows, and engineering. Their graders directly expose dense subtask or continuous rewards.
- **Reconstruction split (31 tasks).** These tasks ask an agent to reproduce the behavior of a parameterized scientific or engineering artifact from public observations. Hidden scenarios test generalization, and a common six-stage ladder converts verifier evidence into dense progress rewards.

This split separates two sources of long-horizon difficulty. Released-workflow tasks require navigating a large, heterogeneous project whose intended objective is stated but difficult to complete. Reconstruction tasks make the implementation target itself partially latent: agents must infer behavior from evidence and then implement a solution that generalizes to withheld cases.

Dense reward and completion. We use the same normalized scale for both benchmark splits. For reconstruction tasks, the dense ladder score replaces the sparse final binary indicator in all cross-task results. We report mean R , a soft pass at $R \geq 0.95$, and a strict pass at $R = 1.0$. Mean reward captures partial progress; the two thresholds distinguish nearly complete solutions from exact completion.

2.3 Evaluation hardening

We extend each task run to three hours and invoke the verifier every 30 minutes to track progress. These checkpoints can reveal agent regressions, such as overwriting previously correct work, as well as evaluation-system defects, including stale verifier artifacts and hidden grader data leaking across phases—problems that a single end-of-run score would not expose. LHTB 2.0 therefore hardens both task execution and score collection. We fixed three harness/configuration defects that prevented tasks using separate verifier environments from producing rewards, isolated verifier phases from the active agent process tree, cleared verifier logs before resuming the agent, and audited successful trajectories for access to hidden fixtures or grader artifacts.

2.4 Domains

The benchmark spans 13 domains: APEX professional workflows [25]; earth, climate, and energy; interactive games; logic and constraint puzzles; multimodal and imaging analysis; research reproduction and machine learning; scientific computing and simulation; software and reverse engineering; systems, performance, and security; geospatial and cartographic workflows; document, OCR, and data engineering; life sciences, chemistry, and materials; and physical simulation, signals, and control. Figure 1 summarizes their distribution, while Appendix A provides the full 77-task inventory.



Figure 1 Task distribution across the 13 LHTB 2.0 domains. The 77 tasks combine the original benchmark domains with geospatial and cartographic workflows; document, OCR, and data engineering; life sciences, chemistry, and materials; and physical simulation, signals, and control.

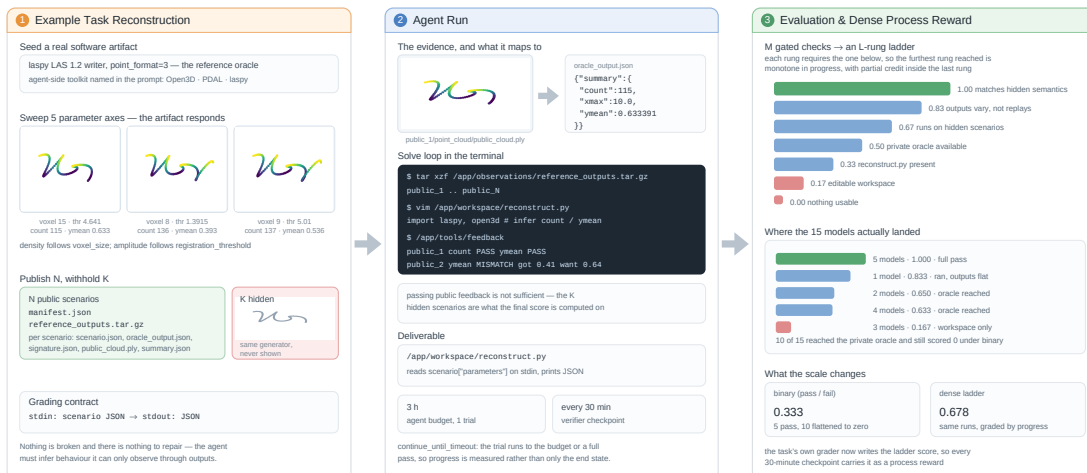
3 Task Construction Examples

The original LHTB tasks were individually designed around complete workflows. This produces realistic and varied environments, but it is costly to scale: every new task requires a plausible project, a reproducible solution, robust hidden tests, and a dense grader. LHTB 2.0 retains that design for the released-workflow split and introduces a complementary construction procedure that turns working domain artifacts into long-horizon reconstruction tasks.

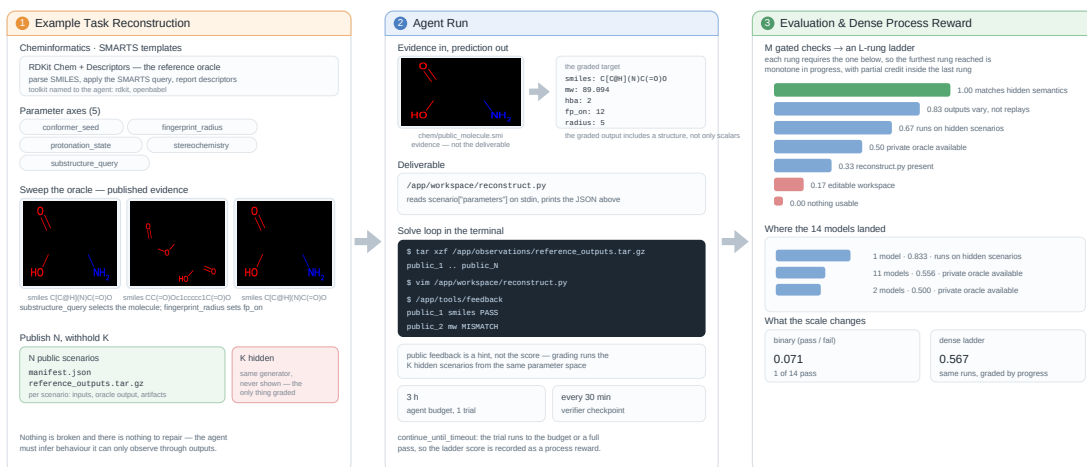
3.1 Construction examples

Figure 2 shows three instances of our task construction recipe. The visual artifacts are the actual public evidence supplied to agents. LiDAR point clouds expose changes in density and amplitude, molecular diagrams expose reaction and descriptor behavior, and acoustic waveforms expose changes in simulated impulse responses.

(a) LiDAR registration and point-cloud summaries



(b) Chemical reaction templates and molecular descriptors



(c) Acoustic beamforming and room simulation

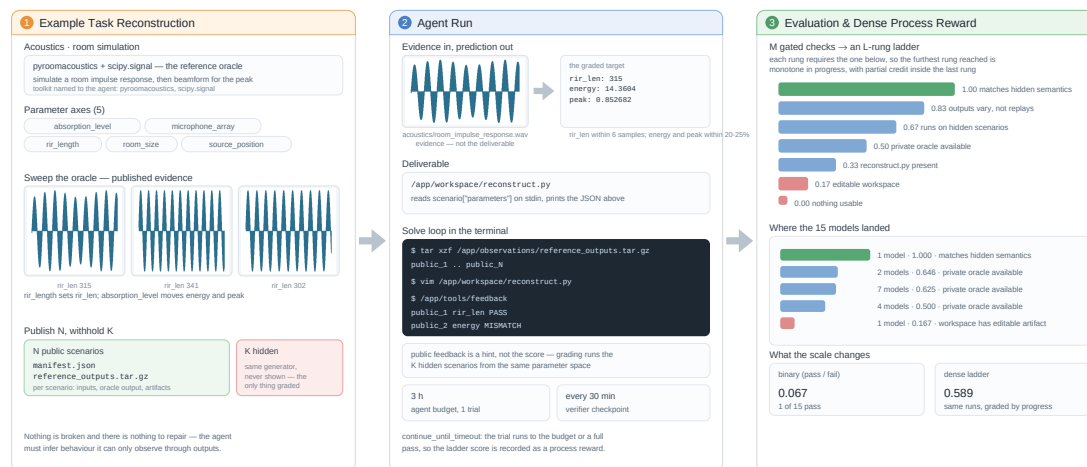


Figure 2 Construction of reconstruction tasks. A working reference artifact is swept over meaningful parameter axes. N scenarios are exposed as behavioral evidence and K are withheld for grading. The agent must implement a terminal program that reproduces behavior on unseen scenarios; public feedback guides debugging but does not reveal the hidden score. The same contract applies to point clouds (a), molecules (b), and waveforms (c).

Each task uses the same public/hidden split, terminal interface, anti-memorization gate, and checkpointed evaluation protocol.

3.2 Task types

The construction pipeline broadens the benchmark, where LHTB 2.0 contains eight task types:

Task type	Count	Long-horizon objective
Reconstruction	30	Infer and reproduce a reference artifact’s behavior from public observations.
Audit and repair	23	Diagnose and repair a broken analysis or audit pipeline against hidden cases.
Optimization	6	Iteratively improve a working implementation toward quality or performance targets.
Gameplay	5	Act over a long interactive campaign.
Reproduction	4	Reproduce a published experiment, benchmark, or scientific result.
Professional workflow	4	Complete a multi-stage professional matter with deterministic grading.
Constraint puzzle	3	Recover a legal solution through iterative interaction with a puzzle environment.
Engineering signoff	2	Drive a hardware or systems flow until all acceptance gates pass.

Table 1 Task types in LHTB 2.0. Reconstruction is the new automated construction family; the remaining types preserve the heterogeneous workflow structure of the original benchmark.

3.3 Quality control

Every task must satisfy three checks before inclusion. First, the reference solution must run reproducibly and achieve $R = 1.0$ on all hidden cases. Second, the starter materials should be incomplete enough that solving the task requires genuine work, rather than a trivial patch or simply copying from what is already visible. Third, we use DeepSeek-V4-Pro [4] to review alignment between each task’s instructions and verifier, ensuring that the instructions provide enough information to satisfy the grader without revealing hidden answers.

4 Experiments

Harness and agents. We evaluate models using the Harbor framework with the Terminus-2 agent harness [8], which interacts with each task environment through a single long-horizon terminal session.

Models. We benchmark GPT-6-astra [22], GPT-5.6-sol [21], DeepSeek V4 Pro [4], DeepSeek V4 Flash [4], Gemini 3.1 Pro [6], Gemini 3.8 Flash [7], GLM-5.3 [31], GLM-5.2 [30], Kimi K3 [10], Kimi K2.6 [20], MiniMax M3 [19], Qwen 3.8 Max [1], Doubao Seed 2.1 Pro [2], Hunyuan hy4 [24], and Grok-4.6 [26].¹

Metrics. For each model, we report the soft pass rate at $R \geq 0.95$, the strict pass rate (the fraction of tasks with normalized reward exactly 1.0), the mean normalized reward across all tasks, the number of episodes required per task, the elapsed runtime per task, the estimated dollar cost per task.

4.1 Main Results

We set the timeout limit for each task to be 3 hours for cost reasons. Figure 3 summarizes pass rate at $R \geq 0.95$ and $R \geq 1.0$, along with mean reward across all 77 tasks. Overall, the evaluated frontier models still struggle on long-horizon tasks, even under a dense-grading evaluation that gives credit for partial progress. GPT-6-astra is the strongest reported model, but still resolves only 48.1% of tasks (37/77) at $R \geq 0.95$.

¹All models run with xhigh reasoning.

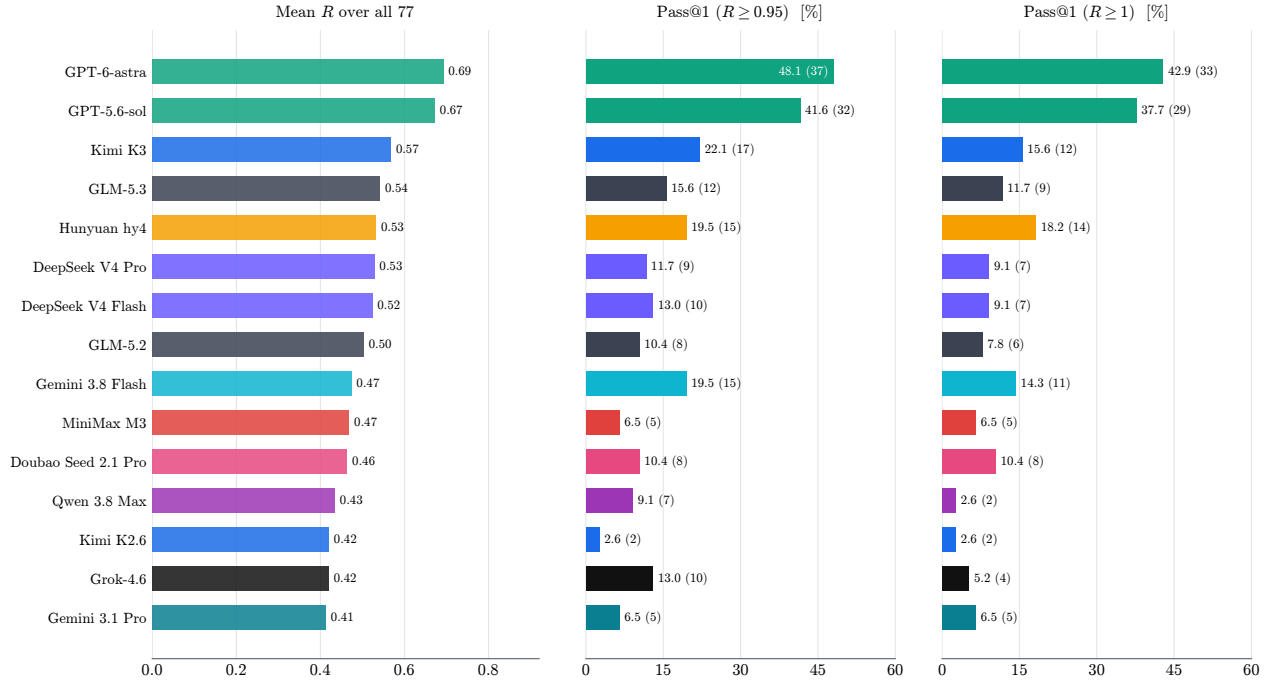


Figure 3 There exists a gap between solve rate and process reward, where models can achieve significant progresses on overall tasks but fail most of them. Pass rate is the fraction of the 77 tasks whose overall reward R meets a threshold τ , with $R \geq 1.0$ corresponding to full completion, while mean R captures partial progress across all tasks. GPT-6-astra achieves the strongest overall performance, resolving 48.1% of tasks at $R \geq 0.95$ and 42.9% at $R = 1.0$.

GPT-5.6-sol follows at 41.6% (32/77). Kimi K3 is a distant third at 22.1% (17/77). A second tier includes hy4 and Gemini 3.8 Flash at 19.5% (15/77) each, followed by GLM-5.3 at 15.6% (12/77). The gap between pass rate and mean reward (0.2-0.5) for several mid-tier systems indicates that many models make substantial partial progress on hard tasks without fully resolving them. The results suggest that long-horizon execution is not merely a local reasoning problem, but remains a central bottleneck even for top frontier models.

GPT-6-astra takes the lead on software reconstruction tasks. GPT-6-astra’s overall lead is concentrated in the reconstruction split. On the 30 reconstruction tasks it reaches a mean reward of 0.93 and solves 24 tasks at $R \geq 0.95$, ahead of GPT-5.6-sol (0.84, 18 tasks) and far ahead of every other model (≤ 0.71 , at most 9 tasks). On the 47 released-workflow tasks, GPT-5.6-sol is slightly ahead (0.57, 14 tasks), with GPT-6-astra second (0.54, 13 tasks) and DeepSeek V4 Flash close behind (0.53). GPT-6-astra is also more step-efficient on reconstruction tasks, solving them after a median of 37 agent steps and 26 minutes, against 67 steps and 36 minutes for GPT-5.6-sol. In the first 40 steps of runs that last that long, about one third of GPT-6-astra’s commands write or edit files and only 28% repeat an earlier command; for GPT-5.6-sol the corresponding numbers are 14% and 50%. On released tasks, that speed buys less: GPT-6-astra is close to its final reward after one hour (0.53 at 60 minutes, 0.56 at three hours), while GPT-5.6-sol keeps improving (0.53 to 0.58) and finishes ahead. Over all 77 tasks GPT-6-astra remains ahead throughout, although GPT-5.6-sol closes most of the gap by the end of the run (Section 4.3).

Open-source models, model performance broadly tracks release date . Earlier releases such as Kimi K2.6 (April 20, 2026) and GLM-5.2 (June 16, 2026) are surpassed by later releases such as Kimi K3 (July 16, 2026; open weights July 27), GLM-5.3 (August 14, 2026; open weights August 28), and Hunyuan hy4 preview (August 28, 2026).

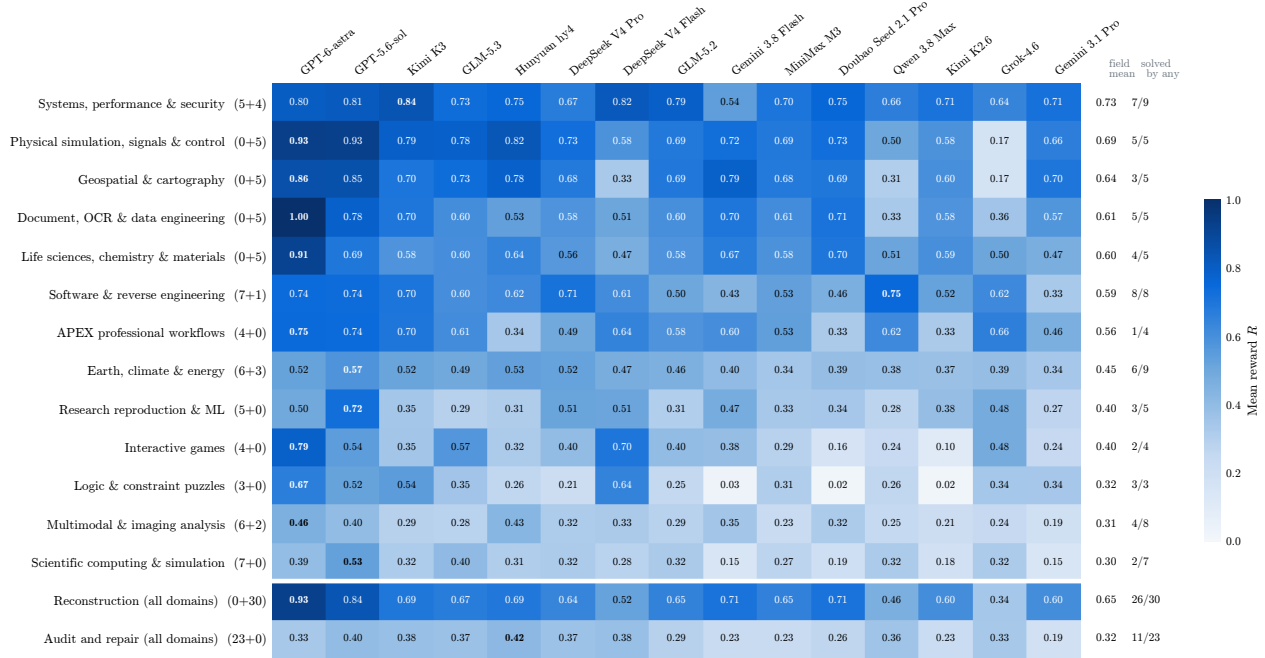


Figure 4 Mean reward by category and model. Rows are the 13 categories, sorted by the mean over all models, followed by the two most common task types pooled across domains; parentheses give the number of released (from LHTB-1.0) and reconstruction tasks (newly added). Each cell averages the dense reward over the row’s tasks, counting unscored tasks as 0, and the best model in each row is shown in bold. The right-hand columns give the mean over all models and the number of tasks that at least one model solves ($R \geq 0.95$).

4.2 Success and Failures

Figure 4 breaks mean reward down by category, task type, and model.

Software and systems tasks are the most tractable for every model. Systems, performance & security has the highest mean across models (0.73), every software & reverse-engineering task is solved by at least one model, and the most widely solved task, `riscv-core-debug`, is solved by 13 of the 15 models. GPT-6-astra is strong but not dominant: it ranks second on software & reverse engineering, just behind Qwen 3.8 Max (0.74 vs. 0.75), and fourth on systems, where Kimi K3 leads (0.84). Its clearest advantages are on interactive games (0.79 against a fleet mean of 0.40) and on document, OCR & data engineering, where it solves all five tasks.

Scientific tasks are among the hardest overall. Scientific computing & simulation (0.30) and multimodal & imaging analysis (0.31) are the two lowest-scoring categories. Of the 24 tasks that no model solves, 20 come from the released split and 12 are audit-and-repair tasks; on nine tasks, no model exceeds $R = 0.2$, and eight of those nine are audit-and-repair. The main difficulty is therefore sustained debugging and repair, not a single missing final step.

4.3 Solve Rate and Reward over Time and Steps

We follow each model through the three-hour budget in both wall-clock time and agent steps (Figure 5). A task counts as softly solved at the first checkpoint whose verifier reward reaches 0.95, which gives 189 solves over 1,143 runs and differs from the dense leaderboard by at most one task per model.

The three-hour budget changes the middle of the leaderboard. Across models, progress often continues well beyond the first hour: the median successful run solves its task after 57 minutes, with 42% of all solves appearing after the one-hour checkpoint and 21% after the two-hour checkpoint. The top stabilizes early.

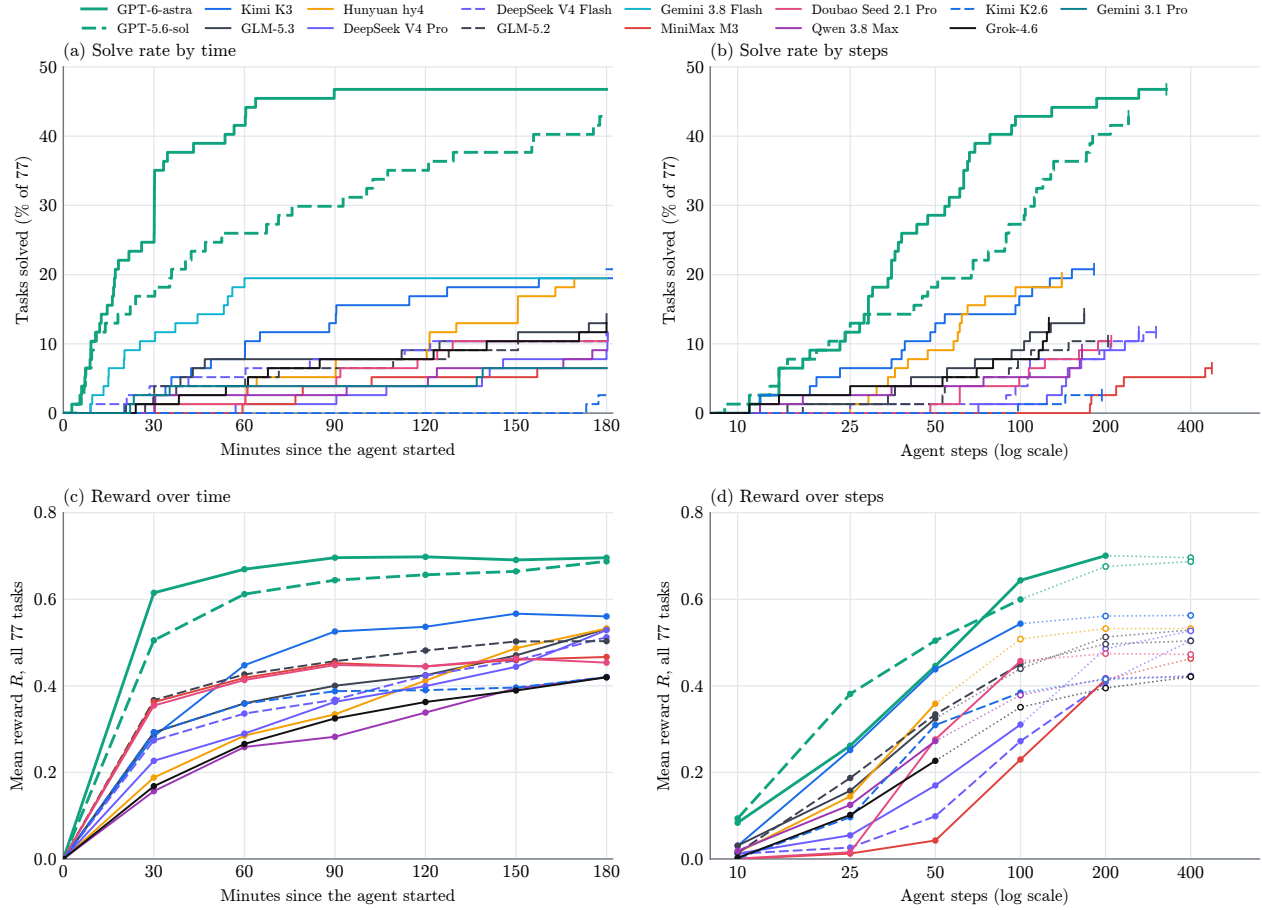


Figure 5 Solve rate and reward over the three-hour run. (a) Many solves arrive after the first hour. Share of the 77 tasks solved by each minute; a task counts as solved at the first checkpoint whose verifier reward reaches 0.95. (b) Late wall-clock solves can reflect slow steps. The same solve curves against agent steps; each curve ends at the median number of steps the model completes in three hours. (c) Mean reward keeps rising after the first hour. (d) More steps do not guarantee more reward. The same mean reward against agent steps. Hollow markers and dotted segments mark budgets that more than 20% of a model’s runs never reach.

GPT-6-astra solves 27 tasks within 30 minutes and 34 of its 36 solved tasks within the first hour, with a median solve time of 24 minutes. The middle does not. Hunyuan hy4 (median 121 minutes) and DeepSeek V4 Pro (141 minutes) solve many tasks late, so a one-hour evaluation would rank hy4 ninth instead of fourth and DeepSeek V4 Pro 13th instead of eighth.

Step efficiency and step latency tell different stories. Wall-clock time combines how many agent steps a task requires and how long each takes. Step latency alone varies by roughly $6\times$, from 23 seconds per step for MiniMax M3 to 141 seconds for Hunyuan hy4; across models, agents spend 48–76% of each run waiting for the model rather than interacting with the environment (Figure 6). This changes how late solves should be interpreted. Hy4, for example, needs a median of only 58 steps to solve a task, fewer than GPT-5.6-sol’s 68, yet it completes fewer than 80 steps within the three-hour budget. By contrast, the most step-hungry models are DeepSeek V4 Pro and MiniMax M3, which require medians of 147 and 218 steps to solve a task. Under an equal-step comparison, GPT-6-astra remains clearly best: it solves 22 tasks within 50 steps, compared with 14 for GPT-5.6-sol and at most 10 for any other model.

Fast models also overclaim completion. Completion claims are informative because the harness verifies them and resumes the run on failure. GPT-6-astra and GPT-5.6-sol make their first completion claim after only 7

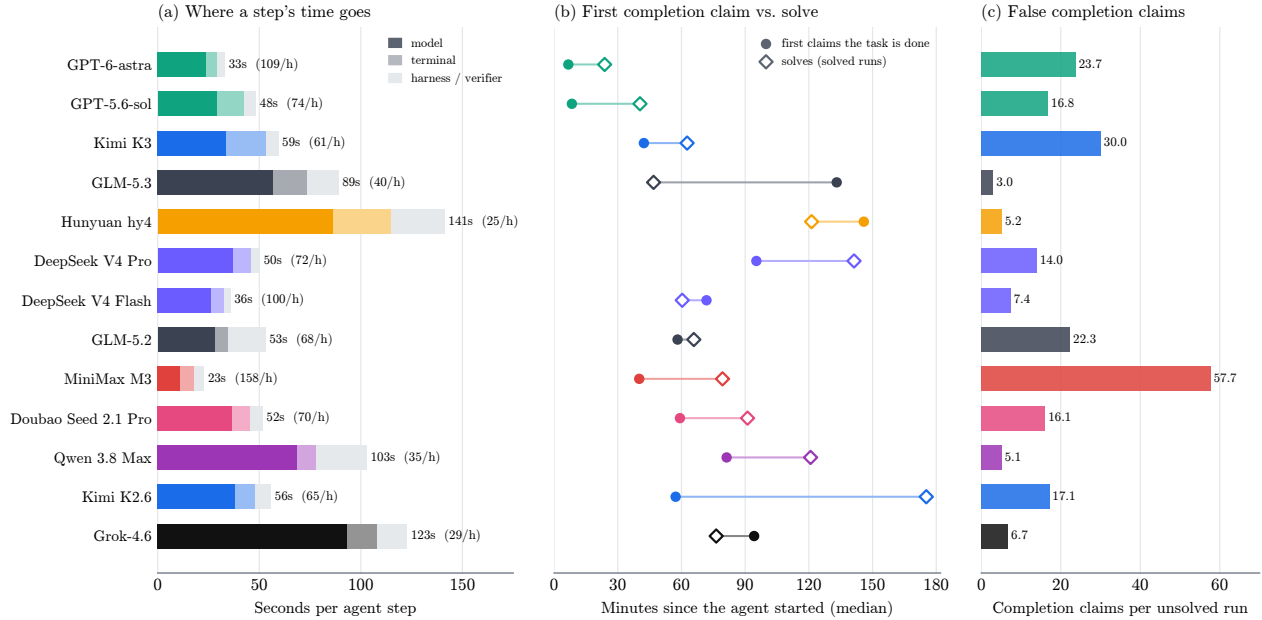


Figure 6 Wall-clock performance reflects both step speed and overconfident stopping. (a) Some models spend much of each step waiting for the model, so they take far fewer steps within three hours. (b) Fast models tend to claim completion earlier, while slower models often wait much longer before doing so. (c) Many early completion claims are false alarms, especially for the most aggressive models.

and 8 minutes on median, and on unsolved runs still claim completion 24 and 17 times on average; MiniMax M3 does so 58 times. By contrast, Hunyuan hy4, GLM-5.3, Grok-4.6, and DeepSeek V4 Pro often delay their first claim until 1.5–2.5 hours and average only 3–14 claims per unsolved run (Figure 6). Across the fleet, 92% of completion claims occur on runs that never pass verification.

4.4 Time and Process Reward Analysis

Process rewards let us measure partial progress during a long-horizon run, not just the final outcome. Figure 7 shows that many runs continue improving after the first checkpoint. Mean reward rises from 0.29 at 30 minutes to 0.43 by the end of the run; 267 of 693 runs gain more than 0.05 after 30 minutes, and 60 cross the soft-pass threshold only after that checkpoint. Even after 60 minutes, 181 runs still gain more than 0.05, and 38 later cross the threshold.

Regression could exist but is not the dominant pattern. We find 65 model–task trajectories whose final reward is more than 0.05 below either the 30 or 60-minute checkpoint; 55 continue declining later, and 54 end at the time limit. These cases are concentrated in games or puzzles, where later actions can damage state or overwrite useful outputs.

Most unsolved shows that unsolved runs still make progress. After removing successful runs, every model’s mean reward at 180 minutes remains above its 60-minute level, and the stronger models maintain higher process rewards throughout (Figure 8). Many failures therefore reflect incomplete long-horizon progress rather than immediate stagnation.

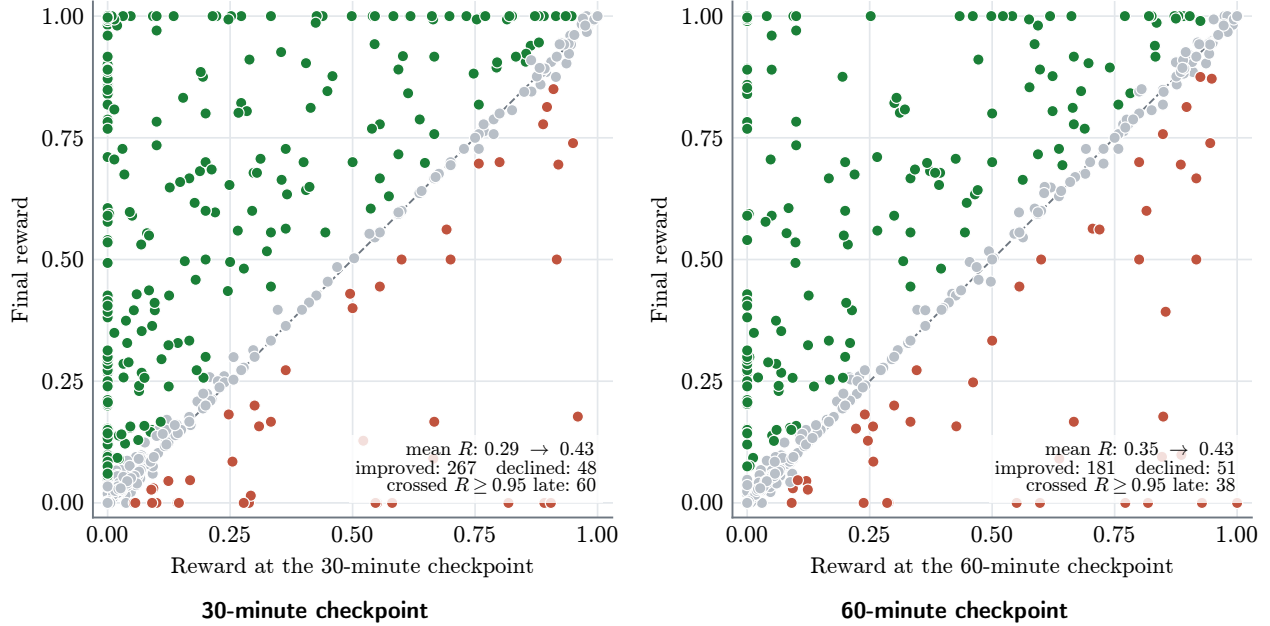


Figure 7 Many runs continue making meaningful progress well after 30 and 60 minutes. Each point is one scored task run: the x-axis shows reward at the 30-minute checkpoint (left) or 60-minute checkpoint (right), and the y-axis shows final reward. The dashed diagonal marks no change. Green points improve by more than 0.05, red points decline by more than 0.05, and gray points stay within that band. In many runs, reward continues to increase after the checkpoint, and some runs newly cross the soft-pass threshold only later.

5 Cost Analysis

Model	Tokens / task (M)	LLM calls / task	LLM time / task (min)	Cost / task (\$)
Hunyuan hy4	4.4	78.9	102.7	3.71
Doubao Seed 2.1 Pro	18.0	200.2	111.7	9.12
DeepSeek V4 Flash	45.5	290.1	127.0	10.07
GLM-5.2	27.9	214.0	93.0	15.43
Qwen 3.8 Max	8.8	113.3	119.0	17.67
DeepSeek V4 Pro	31.2	231.0	130.8	18.21
Kimi K2.6	19.5	191.6	123.2	18.70
GLM-5.3	20.5	137.3	110.8	18.83
MiniMax M3	71.0	452.5	84.3	21.44
Grok-4.6	12.8	133.3	132.8	25.91
Kimi K3	16.2	170.8	93.7	32.09
Gemini 3.1 Pro	13.3	500.2	9.5	32.93
Gemini 3.8 Flash	48.2	836.4	12.7	37.31
GPT-5.6-sol	33.5	175.0	80.1	67.60
GPT-6-astra	39.4	213.5	83.2	397.16
<i>Average</i>	27.3	262.5	94.3	48.41

Table 2 Cost varies sharply across models, and the strongest model is by far the most expensive. On average, models spend 262.5 LLM calls and 94.3 minutes per task, at an estimated \$48.4 per task. Costs are estimated from per-task token usage and public list prices. Tokens/task counts both input and output tokens, and Figure 9 uses the corresponding 77-task suite totals. [†] priced from the nearest listing rather than an exact match.

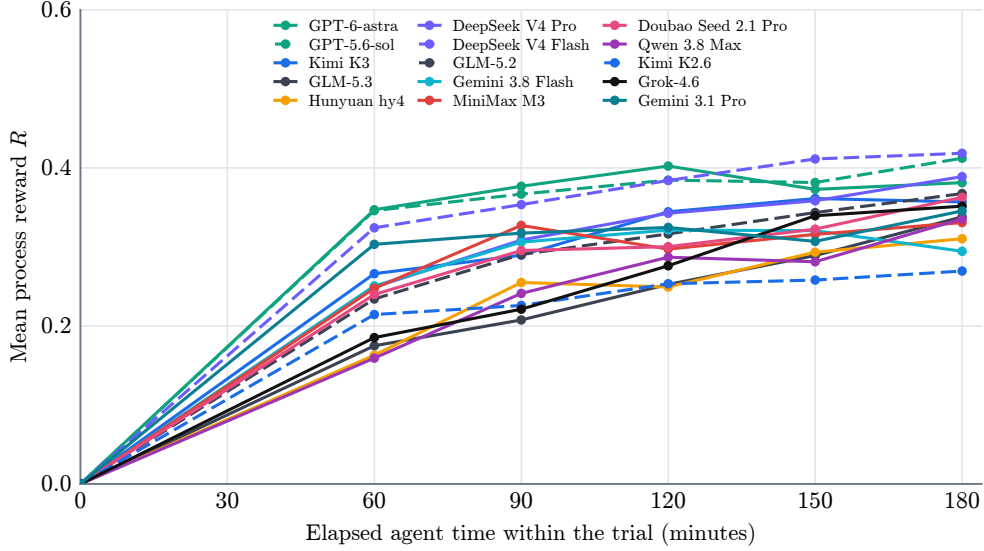


Figure 8 Even unsolved runs continue making progress for hours. Each curve includes only released-task runs whose final reward remains below 0.95. Curves start at (0, 0) by definition and omit the sparsely populated 30-minute interval. Within each time bin, checkpoints are first averaged per task and then across tasks, so every task receives equal weight regardless of how many checkpoints it produced.

We report the cost estimates reconstructed from per-task token usage and public list prices (Table 2), which Figure 9 summarizes with side-by-side Pareto cost–reward frontiers under strict and soft pass criteria. The estimated cost of running the full benchmark spans from about \$285 for Hunyuan hy4 to about \$30,581 for GPT-6-astra. Although GPT-6-astra achieves the highest pass rate, it does so at by far the highest total cost. GPT-5.6-sol forms a second tier of strong but substantially cheaper performance at roughly one sixth of that cost, while Kimi K3 sits further down the cost curve with much lower spend but also a noticeably lower resolution rate. Hunyuan hy4 anchors the low-cost end of the Pareto frontier, while GPT-5.6-sol and GPT-6-astra extend that frontier upward as progressively stronger but more expensive models. Under the soft $R \geq 0.95$ criterion, Kimi K3 joins the frontier because several of its near-misses are counted as passes rather than perfect completions.

Most spending goes to unsolved tasks and to the last hour. Figure 10 breaks spending down by outcome and by hour. Between 83% and 99% of every model’s spend goes to tasks it never solves; GPT-6-astra spends an average of \$664 on an unsolved task and \$101 on a solved one. This is due to that every step resends a longer history, so hours one, two, and three account for 18%, 31%, and 51% of benchmark-wide spend, while solves are front-loaded (58%, 21%, and 21%). A solve found in the third hour therefore costs about eight times as much as one found in the first (\$708 against \$90).² Per solved task, costs range from \$19 for Hunyuan hy4 to \$838 for GPT-6-astra.

6 Model-wise Model Trajectory Analysis

We parse every shell command in the 1,135 runs that kept a full trajectory and compare how models work out the tasks rather than how well they do, where run is described by its mix of command types.

Different models use similar toolsets, but not the same commands. When two different models attempt the same task, their trajectories share only 3% of exact commands and 18% of tool signatures on average, even though their overall mixes of programs are much more similar (cosine similarity 0.63). In other words, the

²Our estimates apply no prompt-cache discount, which would cheapen the resent history, so this ratio is an upper bound.

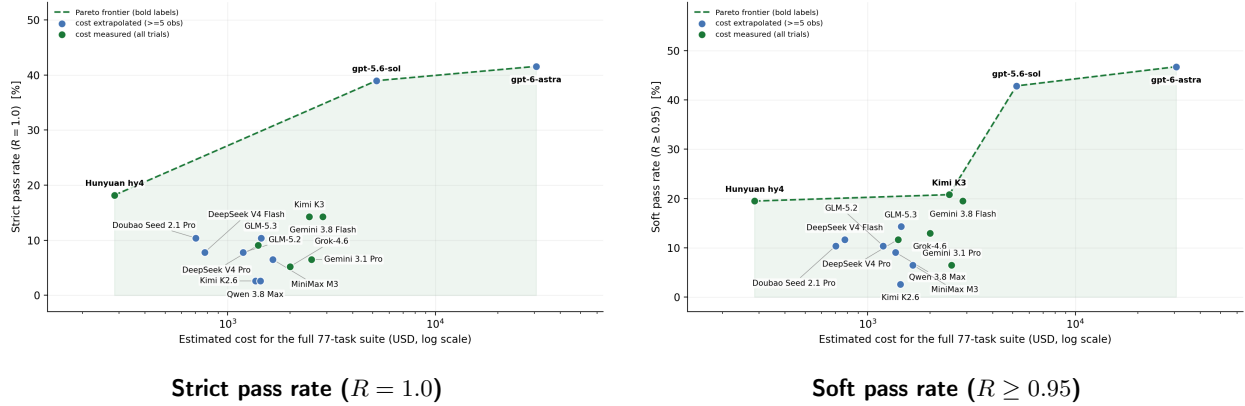


Figure 9 Better long-horizon performance comes at sharply higher cost. Estimated cost of the full 77-task suite (log scale) against strict pass rate ($R = 1.0$, left) and soft pass rate ($R \geq 0.95$, right). The dashed line traces the Pareto frontier, and bold labels mark models on that frontier. Softening the pass criterion adds Kimi K3 to the frontier. Hunyuan hy4 is the strongest low-cost model, while GPT-6-astra is the strongest overall at the highest cost.

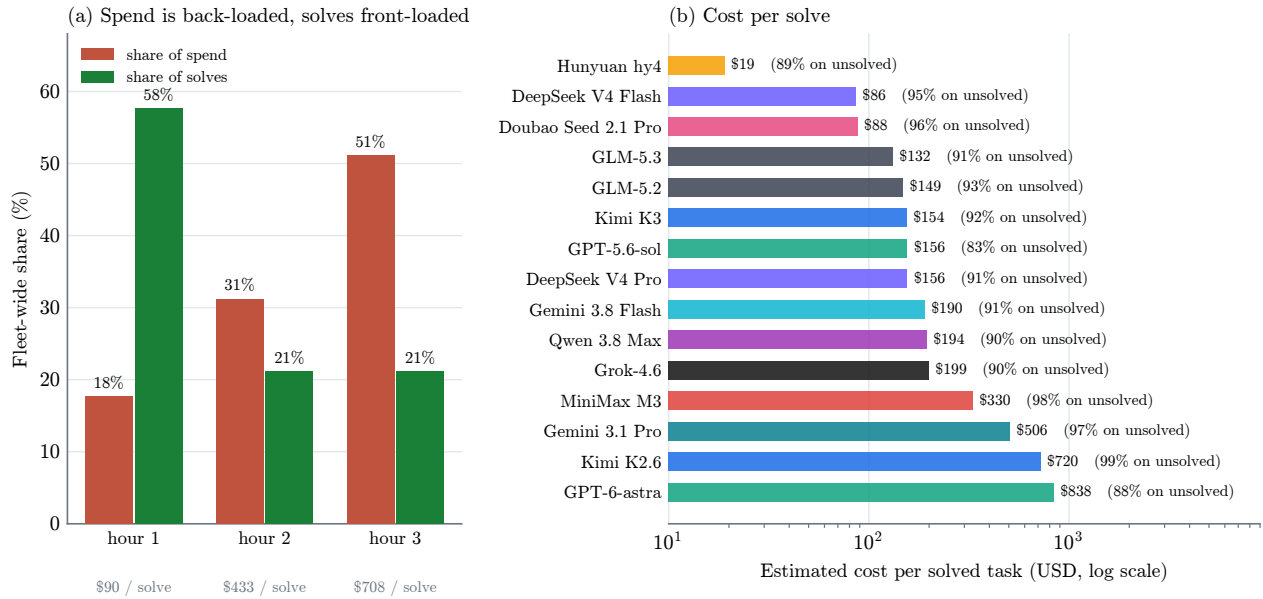


Figure 10 Most spending goes to the last hour and to tasks that never solve. (a) Solves are front-loaded, but spending is back-loaded. This panel shows the fleet-wide share of estimated spend and of solves in each hour of the run; below each pair, the spend in that hour divided by the solves found in it. (b) Most model spending goes to tasks that never solve. This panel shows estimated cost per solved task for each model (log scale), with the share of the model's spend that goes to tasks it never solves. Costs use public list prices without prompt-cache discounts.

task largely determines which tools are used, but the specific command sequence still depends strongly on the model.

Simple command statistics separate successful runs from unsuccessful ones, but not stronger failures from weaker failures. Across all runs, trajectories that spend more commands editing files tend to score higher, while trajectories with long repeated-command streaks tend to score lower (Figure 11b). These statistics mainly help distinguish runs that eventually solve the task from runs that do not. However, among the 800 runs that still fail after using up the full three-hour budget, they do not reliably indicate which runs made more progress than others.

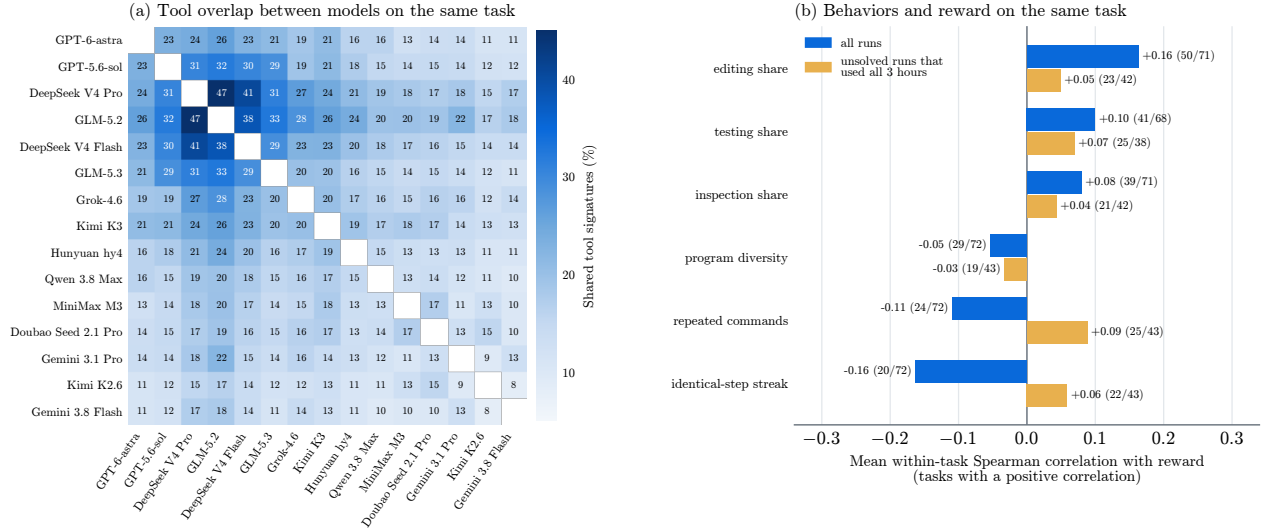


Figure 11 Models use similar toolsets, but stronger runs edit more and repeat less. (a) The task constrains the toolset, but models still take different paths. Share of tool signatures (program plus targeted script or subcommand) that two models’ runs on the same task have in common, averaged over tasks; models are ordered by similarity. (b) Higher-reward runs tend to edit, test, and inspect more, while repeated-command loops correlate with lower reward; program diversity matters little. Spearman correlation of each command statistic with reward across models on the same task, averaged over tasks; the label counts tasks with a positive correlation. The two Gemini submissions, which ran under a different harness, are excluded.

7 Related Work

Long-horizon terminal tasks and trajectories. Long-horizon terminal benchmarks move beyond isolated code generation to tasks that require agents to inspect files, invoke tools, use software, and recover from errors over many steps. Benchmarks such as Terminal-Bench 2.1, Terminal Bench 4.0, Long-Horizon-Terminal-Bench, and LongCLI-Bench progressively raise the difficulty and horizon of these evaluations [5, 18]. As task horizons grow, reliable completion becomes harder [11], and performance depends not only on the model but also on the harness that provides tools, manages context, and handles execution feedback [9, 29]. At the same time, longer and more realistic tasks are harder to support with high-quality reference solutions and agent trajectories, because collecting and validating them becomes increasingly expensive and time-consuming [3, 13]. Progress in long-horizon evaluation therefore depends not only on constructing harder and more diverse tasks, but also on building the solution and trajectory resources needed to understand how agents succeed or fail on them [12, 16].

Long-horizon task evaluation and training. For long horizon tasks, an unsuccessful final state may still contain completed sub-tasks and useful work. Dense, verifiable process rewards credit intermediate achievements, while repeated checkpoints show whether progress continues, stalls, or is lost [12, 15]. This distinguishes an agent that nearly finishes from one that makes little headway, and helps locate failures in planning, implementation, or feedback use [12]. Intermediate rewards also address sparse credit assignment during reinforcement learning by supplying feedback before a long rollout ends [17, 27, 28]. Recent work on early reward prediction reports gains in coding-agent RL while reducing rollout cost [28], showing the important of process rewards on long-horizon tasks.

8 Conclusion

LONG-HORIZON-TERMINAL-BENCH 2.0 extends the original LHTB to 77 terminal tasks spanning 13 domains and is designed to evaluate long-horizon execution. Across 15 frontier models evaluated with a shared

terminus-2 agent, the strongest model, GPT-6-astra, resolves only 48.1% of tasks (37/77) at $R \geq 0.95$, while only 192 of 1,145 scored runs (16.8%) pass overall. At the same time, outcome-only evaluation misses substantial progress: many runs continue improving over long horizons, process reward often rises steadily even on runs that never fully pass, and a large fraction of total cost is spent in the final hour of unsolved runs. These results suggest that the main challenge in long-horizon terminal tasks is not only local reasoning, but also sustaining productive work, preserving progress, and converting partial solutions into reliable completion over time.

References

- [1] Alibaba Cloud. Qwen3.8-Max Model Info. <https://www.alibabacloud.com/help/en/model-studio/qwen3-8-max>, 2026.
- [2] ByteDance Seed Team. Seed2.1 officially released: Advancing ai productivity. <https://seed.bytedance.com/en/blog/seed2-1-officially-released-advancing-ai-productivity>, June 2026. Official model release announcement (Doubao Seed 2.1 / Seed 2.1 Pro).
- [3] Zhaoyang Chu, Jiarui Hu, Xingyu Jiang, Pengyu Zou, Han Li, Chao Peng, Peter O’Hearn, Earl T. Barr, Mark Harman, Federica Sarro, and He Ye. Terminalworld: Benchmarking agents on real-world terminal tasks, 2026. URL <https://arxiv.org/abs/2605.22535>.
- [4] DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026. URL <https://arxiv.org/abs/2606.19348>.
- [5] Yukang Feng, Jianwen Sun, Zelai Yang, Jiaxin Ai, Chuanhao Li, Zizhen Li, Fanrui Zhang, Kang He, Rui Ma, Jifan Lin, et al. Longcli-bench: A preliminary benchmark and study for long-horizon agentic programming in command-line interfaces. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 29952–29963, 2026.
- [6] Google DeepMind. Gemini 3.1 pro model card. <https://deepmind.google/models/model-cards/gemini-3-1-pro/>, 2026.
- [7] Google DeepMind. Gemini 3.8 Flash Model Card. <https://deepmind.google/models/model-cards/gemini-3-8-flash/>, 2026.
- [8] Harbor Framework Team. Harbor: A framework for evaluating and optimizing agents and models in container environments, 2026. URL <https://doi.org/10.5281/zenodo.20953922>.
- [9] Haichuan Hu, Quanjun Zhang, Shengcheng Yu, Zhifei Chen, Tianyu Luo, Chunrong Fang, Zhenyu Chen, and Liang Xiao. Beyond the model: Demystifying harness effects in software engineering agents, 2026. URL <https://arxiv.org/abs/2609.32459>.
- [10] Kimi Team. Kimi K3: Open Frontier Intelligence, 2026. URL <https://arxiv.org/abs/2607.24653>.
- [11] Thomas Kwa, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, Sami Jawhar, Megan Kinniment, Nate Rush, Sydney Von Arx, Ryan Bloom, Thomas Broadley, Haoxing Du, Brian Goodrich, Nikola Jurkovic, Luke Harold Miles, Seraphina Nix, Tao Lin, Chris Painter, Neev Parikh, David Rein, Lucas Jun Koba Sato, Hjalmar Wijk, Daniel M. Ziegler, Elizabeth Barnes, and Lawrence Chan. Measuring ai ability to complete long software tasks, 2026. URL <https://arxiv.org/abs/2503.14499>.
- [12] Yiwei Li, Wanli Yang, Hexiang Tan, Xiangzhou Huang, Zhengyu Chen, Ziran Li, Borun Chen, Shanglin Lei, Huaisheng Zhu, Hao Tian, Fei Sun, Xunliang Cai, and Jingang Wang. Beyond final scores: A systematic evaluation of agents for long-horizon ai research and development, 2026. URL <https://arxiv.org/abs/2608.13417>.
- [13] Zhongzhi Li, Yucheng Shi, Zongxia Li, Ruhan Wang, Anhao Li, Zixun Huang, Junyao Yang, Lei Ke, Ninghao Liu, Haitao Mi, and Leowei Liang. Recursive synthesis for long-horizon terminal tasks, 2026. URL <https://arxiv.org/abs/2608.05466>.
- [14] Zhongzhi Li, Yucheng Shi, Zongxia Li, Junyao Yang, Ruhan Wang, Yu Wang, Jingyuan Huang, Jichao Yu, Ninghao Liu, Haitao Mi, and Leowei Liang. Self-supervised scaling of terminal environments for scientific domains, 2026. URL <https://arxiv.org/abs/2610.02710>.

- [15] Zongxia Li, Zhongzhi Li, Yucheng Shi, Ruhan Wang, Junyao Yang, Zhichao Liu, Xiyang Wu, Anhao Li, Yue Yu, Ninghao Liu, Lichao Sun, Haotao Mi, and Leowei Liang. Long-horizon-terminal-bench: Testing the limits of agents on long-horizon terminal tasks with dense reward-based grading, 2026. URL <https://arxiv.org/abs/2607.08964>.
- [16] Zongxia Li, Yucheng Shi, Zhongzhi Li, Junyao Yang, Ruhan Wang, Chengsong Huang, Fuxiao Liu, Haitao Mi, Jordan Boyd-Graber, and Leowei Liang. Scaling trajectories for complex tasks through recursive self-rewrite, 2026. URL <https://arxiv.org/abs/2610.02826>.
- [17] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let’s verify step by step, 2023. URL <https://arxiv.org/abs/2305.20050>.
- [18] Mike A Merrill, Alexander G Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E Kelly Buchanan, et al. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. *arXiv preprint arXiv:2601.11868*, 2026.
- [19] MiniMax. Minimax sparse attention, 2026. URL <https://arxiv.org/abs/2606.13392>.
- [20] Moonshot AI. Kimi k2.6: From code to creation, from one to many. <https://www.kimi.com/ai-models/kimi-k2-6>, 2026. Official Moonshot AI model release page (Kimi K2.6).
- [21] OpenAI. GPT-5.6 Sol Model. <https://developers.openai.com/api/docs/models/gpt-5.6-sol>, 2026.
- [22] OpenAI. GPT-6 Astra Model. <https://developers.openai.com/api/docs/models/gpt-6-astra>, 2026.
- [23] Xiaoshuai Song, Haofer Chang, Guanting Dong, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Envscaler: Scaling tool-interactive environments for llm agent via programmatic synthesis, 2026. URL <https://arxiv.org/abs/2601.05808>.
- [24] Tencent. Tencent Releases and Open-Sources Tencent Hy4 Preview. <https://www.tencent.com/tencent-releases-and-open-sources-tencent-hy4-preview/>, 2026.
- [25] Bertie Vidgen, Austin Mann, Abby Fennelly, John Wright Stanly, Lucas Rothman, Marco Burstein, Julien Benckek, David Ostrofsky, Anirudh Ravichandran, Debnil Sur, Neel Venugopal, Alannah Hsia, Isaac Robinson, Calix Huang, Olivia Varones, Daniyal Khan, Michael Haines, Austin Bridges, Jesse Boyle, Koby Twist, Zach Richards, Chirag Mahapatra, Brendan Foody, and Osvald Nitski. Apex-agents, 2026. URL <https://arxiv.org/abs/2601.14242>.
- [26] xAI. Introducing Grok 4.6. <https://x.ai/news/grok-4-6>, 2026.
- [27] Junyao Yang, Yucheng Shi, Zhongzhi Li, Ruhan Wang, Zongxia Li, Haitao Mi, and Leowei Liang. T1: Terminal agent reinforcement learning for long-horizon tasks, 2026. URL <https://arxiv.org/abs/2609.11042>.
- [28] Jihan Yao, Sihan Zeng, Shangbin Feng, Zhiyuan Fan, Banghua Zhu, and Yulia Tsvetkov. Before the rollout ends: Early terminal reward prediction for long-horizon coding agents, 2026. URL <https://arxiv.org/abs/2609.31995>.
- [29] Yilun Yao, Xinyu Tan, Chao-Hsuan Liu, Yaoming Li, Zhengyang Wang, Wenhan Yu, Zhewen Tan, Yuxuan Tian, Guangxiang Zhao, Lin Sun, Xiangzheng Zhang, and Tong Yang. Harness-bench: Measuring harness effects across models in realistic agent workflows, 2026. URL <https://arxiv.org/abs/2605.27922>.
- [30] Z.ai. GLM-5.2: Built for Long-Horizon Tasks. <https://z.ai/blog/glm-5.2>, 2026.
- [31] Z.ai. GLM-5.3: Frontier Coding with Emergent Cyber Capabilities. <https://z.ai/blog/glm-5.3>, 2026.

A Task Inventory

Table 3 lists all 77 benchmark tasks using the 13-domain taxonomy used throughout the paper. The first 46 rows preserve the original LHTB suite; the remaining 31 rows are the LHTB 2.0 additions, comprising 30 reconstruction tasks and one optimization task. Difficulty is derived from dense mean reward across model results in Figure 3: tasks with mean reward at least 0.5 are labeled *Easy*, and the rest are labeled *Hard*.

Table 3 Task list for Long-Horizon-Terminal-Bench 2.0. Each row shows a task ID, its paper-level category, a one-sentence description, and a coarse difficulty label derived from average task reward in Figure 3 (*Easy* if mean reward ≥ 0.5 , otherwise *Hard*).

Task ID	Category	Description	Difficulty
2048	Interactive games	Play 2048 turn-by-turn via a terminal game server, merging tiles to reach the highest tile, scored by replaying recorded moves in a fresh seeded engine.	Hard
alp-paper-reproduction	Research reproduction & ML	Reproduce a calibrated ALP floating-point compression experiment from the paper’s method and sample columns.	Hard
apex-ib244-matter	APEX professional workflows	Advise on taking EdTech firm KSchool private across 38 reactive stages, rebuilding LBO/DCF/comps models for exact figures.	Easy
apex-investment-banking-matter	APEX professional workflows	Work a private-equity take-private of Planet Fitness through 19 reactive stages, rebuilding an LBO model and answering exact figures.	Hard
apex-law433-matter	APEX professional workflows	Act as outside counsel across four legal matters and 70 reactive stages over ~600 documents, submitting deterministically-graded answers.	Easy
apex-management-consulting-matter	APEX professional workflows	Work two management-consulting client matters across 33 reactive stages, rebuilding business-case and cost-savings models for exact answers.	Hard
apex-openroad-ibex-signoff	Software & reverse engineering	Close RTL-to-GDSII signoff on the ibex CPU with OpenROAD/sky130 across reactive stages until timing, DRC, LVS, area, and power pass.	Hard
audio-visual-event-alignment	Multimodal & imaging analysis	Repair an audio-video sync audit that detects visual events, audio onsets, dropped frames, and per-clip synchronization offsets.	Hard
chess-mate	Interactive games	Play chess as White against a bundled superhuman Stockfish engine, scored on how long the agent survives before being checkmated.	Hard
climate-netcdf-extreme-event-audit	Earth, climate & energy	Repair a climate audit pipeline computing gridded heatwave, precipitation, drought, anomaly, and index metrics over NetCDF datasets.	Hard
commit0-multilib-tdd	Software & reverse engineering	Reimplement three whole Python libraries from stubs with no provided tests, writing your own suite, graded on hidden real test suites.	Hard
dicom-radiology-audit	Multimodal & imaging analysis	Audit synthetic DICOM CT series geometry, HU conversion, windowing, masks, and lesion measurements.	Hard
document-table-layout-reconstruction	Multimodal & imaging analysis	Repair a scanned-document pipeline that detects table grids, assigns noisy OCR tokens to cells, and emits CSV/JSON/Markdown tables.	Hard
duckdb-optimizer-closure	Systems, performance & security	Iteratively patch DuckDB’s query optimizer to speed up TPC-H while preserving correctness.	Hard

Task ID	Category	Description	Difficulty
epa-swmm-stormwater-regression-audit	Earth, climate & energy	Audit EPA SWMM stormwater hydraulic and hydrologic simulations against official solver outputs.	Hard
epidemic-inverse-control-audit	Scientific computing & simulation	Fit and control a multi-city, age-structured SEIR-H-ICU epidemic model.	Hard
foldseek-paper-reproduction	Research reproduction & ML	Reproduce a calibrated Foldseek-style structural protein search experiment from the Nature Biotechnology paper.	Hard
gdal-proj-raster-regression	Earth, climate & energy	Reproduce GDAL/PROJ raster reprojection, coordinate transform, point sampling, and zonal-statistics regression checks.	Hard
generals-bot-arena	Software & reverse engineering	Develop a bot for a Generals.io-style grid strategy game by iteratively playing matches and refining its strategy.	Easy
grammar-fuzz-coverage-hunt	Systems, performance & security	Iteratively build a grammar-guided fuzzer to maximize code coverage of target parsers.	Easy
great-expectations-audit	Software & reverse engineering	Build a Great Expectations-based multi-table data-quality and reconciliation audit pipeline.	Hard
langchain-version-migration	Software & reverse engineering	Migrate a legacy LangChain RAG/router app across a breaking API change with hard-fail compatibility gates.	Hard
materials-phase-diagram-audit	Scientific computing & simulation	Repair a materials phase-diagram audit reconstructing phase boundaries, invariant reactions, lever-rule fractions, and convex-hull stability.	Hard
matpower-opf-regression	Earth, climate & energy	Reproduce MATPOWER/PYPOWER power-flow and optimal-power-flow regression checks with grid-constraint diagnostics.	Hard
microscopy-cell-count-qc-audit	Multimodal & imaging analysis	Repair a microscopy QC audit that segments synthetic cell imagery, flags focus/artifact failures, and reports per-tile channel counts.	Hard
modflow6-groundwater-regression-audit	Earth, climate & energy	Audit USGS MODFLOW 6 groundwater simulations against official solver outputs.	Hard
nbody-accel-iterative	Scientific computing & simulation	Iteratively optimize an N-body gravitational simulation from $O(N^2)$ to $O(N \log N)$ with correctness verification at multiple scales.	Easy
nrel-pysam-hybrid-renewables-audit	Earth, climate & energy	Audit renewable generation and storage dispatch against NREL PySAM official PVWatts outputs.	Hard
opensees-seismic-structural-regression-audit	Scientific computing & simulation	Audit seismic structural-dynamics results against official OpenSeesPy finite-element solver outputs.	Hard
poc-exploit-craft	Systems, performance & security	Generate proof-of-concept inputs that trigger specific vulnerability classes using sanitizer feedback.	Easy
riscv-core-debug	Software & reverse engineering	Localize and fix undisclosed injected bugs in an out-of-order RISC-V SystemVerilog CPU until simulation, compliance, and formal checks pass.	Easy
robotics-slam-benchmark-repair	Research reproduction & ML	Repair a SLAM benchmark audit that composes SE(2) poses, evaluates loop closures, reconstructs landmark maps, and reports trajectory consistency.	Hard
rush_hour_campaign	Logic & constraint puzzles	Solve a four-stage 6x6 Rush Hour campaign by hand, submitting legal move routes under length caps.	Hard

Task ID	Category	Description	Difficulty
satellite-flood-change-detection-audit	Multimodal & imaging analysis	Repair a satellite flood change-detection audit that compares before/after imagery, masks clouds, and reports per-tile flood extent.	Hard
scientific-figure-data-reconstruction	Multimodal & imaging analysis	Repair a pipeline that reconstructs plotted series data from scientific figure images, captions, and partial CSV references.	Hard
snake_maze_campaign	Interactive games	Play a deterministic hard Snake variant, eating seeded apples and surviving growing obstacles as long as possible, scored by replaying the move log.	Hard
sokoban	Logic & constraint puzzles	Solve a ramp of Sokoban puzzles turn-by-turn through a game server, advancing as far as possible, scored by replaying moves in a clean engine.	Hard
spice-ephemeris-regression	Scientific computing & simulation	Reproduce NASA SPICE ephemeris, angle, and ground-station visibility regression checks from pinned kernels.	Hard
spot-scheduler-traces	Systems, performance & security	Implement and refine a cloud spot-instance scheduling policy that minimizes cost while meeting hard deadlines.	Easy
su2-airfoil-regression	Scientific computing & simulation	Reproduce a hardened SU2 aerodynamic regression matrix from official test-case references and solver history artifacts.	Hard
sudoku-recovery	Logic & constraint puzzles	Recover seven progressively corrupted hard variant-Sudoku puzzles turn-by-turn, detecting subtle corrupt givens via global constraint reasoning.	Hard
super-mario	Interactive games	Play Super Mario Bros turn-by-turn through a game server, clearing as many levels as possible, scored by replaying frame inputs in a fresh emulator.	Hard
tabular-data-feature-covshift	Research reproduction & ML	Recover a sparse high-dimensional signal from low-dimensional observations under covariate shift, with a leakage-audited honesty gate.	Hard
unison-paper-reproduction	Research reproduction & ML	Reproduce a calibrated UNISON fat-tree network simulation experiment from the paper’s method and settings.	Hard
unknown-config-semantics	Software & reverse engineering	Reverse-engineer an undocumented config format from a noisy spec and repair its normalization engine to pass hidden probes across five stages.	Hard
vector-db-iterative-build	Systems, performance & security	Build an approximate nearest-neighbor vector search service from scratch, iteratively optimizing recall and throughput.	Hard
acoustic_beamforming_peak_01	Physical simulation, signals & control	Reconstruct a room-acoustics beamforming pipeline so its peak-direction estimates match a reference simulator.	Hard
ajv_dynamic_ref_tree_05	Software & reverse engineering	Reconstruct AJV JSON-schema validation behaviour over dynamic \$ref trees from observed accept/reject traces.	Hard
astro_mosaic_reprojection_04	Multimodal & imaging analysis	Reconstruct an astronomical FITS mosaic reprojection and photometry pipeline from reference outputs.	Hard
chem_reaction_smarts_application_02	Life sciences, chemistry & materials	Reconstruct SMARTS reaction-template application so product sets match a reference cheminformatics engine.	Hard
circuit_buck_converter_ripple_01	Physical simulation, signals & control	Reconstruct a buck-converter circuit simulation reproducing reference ripple and transient measurements.	Hard

Task ID	Category	Description	Difficulty
climate_regrid_ conservation_check_ _06	Earth, climate & energy	Reconstruct a NetCDF climate regridding pipeline that preserves the reference conservation diagnostics.	Hard
duckdb_recursive_ hierarchy_report__03	Document, OCR & data engineering	Reconstruct a DuckDB recursive-hierarchy report matching reference aggregation output.	Hard
fea_plate_hole_ stress_concentration_ _01	Physical simulation, signals & control	Reconstruct a finite-element plate-with-hole analysis reproducing reference stress-concentration factors.	Hard
font_harfbuzz_ shaping_cluster__04	Document, OCR & data engineering	Reconstruct HarfBuzz text-shaping cluster and glyph-advance behaviour from observed shaping output.	Hard
gdal_antimeridian_ clip__01	Geospatial & cartography	Reconstruct GDAL antimeridian-crossing clip geometry matching reference vector output.	Hard
gdal_vrt_mosaic_ transform__03	Geospatial & cartography	Reconstruct a GDAL VRT mosaic and coordinate transform reproducing reference raster statistics.	Hard
genetic-convergence- testing	Scientific computing & simulation	Optimize a stochastic convergence-testing workflow against hidden statistical and runtime criteria.	Hard
geology_multiwell_ correlation_panel__ 04	Earth, climate & energy	Reconstruct a multi-well log correlation panel matching reference stratigraphic picks.	Hard
ghostscript_pdfa_ preflight_repair__06	Document, OCR & data engineering	Reconstruct a Ghostscript PDF/A preflight-and-repair pipeline matching reference compliance reports.	Hard
hydro_hydrograph_ response__06	Earth, climate & energy	Reconstruct a raster hydrology model reproducing reference hydrograph response curves.	Easy
lidar_icp_ registration_ summary__06	Geospatial & cartography	Reconstruct LiDAR ICP point-cloud registration reproducing reference alignment summaries.	Hard
map_label_priority_ collision__04	Geospatial & cartography	Reconstruct cartographic label placement and collision priority matching reference rendered labels.	Hard
map_road_class_ generalization__05	Geospatial & cartography	Reconstruct road-class cartographic generalization matching reference simplified geometry.	Hard
materials_xrd_peak_ shift__04	Life sciences, chemistry & materials	Reconstruct X-ray diffraction peak-shift analysis reproducing reference lattice measurements.	Hard
medical_slice_series_ reassembly__03	Multimodal & imaging analysis	Reconstruct medical slice-series reassembly and geometry recovery matching reference volumes.	Hard
nginx_location_ regex_tryfiles_ precedence__06	Systems, performance & security	Reconstruct nginx location/regex/try_files routing precedence from observed request traces.	Hard
opa_rbac_ namespace_ boundary__05	Systems, performance & security	Reconstruct OPA/Rego RBAC namespace-boundary policy decisions from observed allow/deny traces.	Hard
openssl_pkits_name_ constraints__05	Systems, performance & security	Reconstruct OpenSSL PKITS name-constraint certificate validation from reference verdicts.	Easy
pcap_http2_tls_ fingerprint__04	Systems, performance & security	Reconstruct HTTP/2 and TLS fingerprint extraction from packet captures matching reference fingerprints.	Hard
pdb_multimodel_ rmsd__02	Life sciences, chemistry & materials	Reconstruct multi-model PDB structural RMSD computation matching reference superposition values.	Hard
proteomics_ retention_time_ alignment__02	Life sciences, chemistry & materials	Reconstruct mass-spectrometry retention-time alignment matching reference peptide mappings.	Hard

Task ID	Category	Description	Difficulty
robotics_pick_reach_controller__01	Physical simulation, signals & control	Reconstruct a robotic pick-and-reach controller reproducing reference trajectories and success criteria.	Hard
samtools_mpileup_consensus__05	Life sciences, chemistry & materials	Reconstruct a samtools mpileup consensus-calling pipeline matching reference variant output.	Hard
seismo_cross_correlation_lag__01	Physical simulation, signals & control	Reconstruct seismic waveform cross-correlation lag estimation matching reference arrival offsets.	Easy
sqlite_wal_recovery_audit__02	Document, OCR & data engineering	Reconstruct a SQLite WAL recovery audit reproducing reference post-crash database state.	Hard
tesseract_table_image_ocr__06	Document, OCR & data engineering	Reconstruct a Tesseract table-image OCR pipeline matching reference extracted tables.	Hard