CERTIFIED FEASIBILITY REPORTING UNDER PARTIAL OBSERVABILITY: DISTINGUISHING UNREACHABLE FROM UNRESOLVED IN TOOL-USING AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a mechanically certified evaluation of *feasibility reporting*: whether a tool-using agent can report what its observations establish about task reachability. The benchmark separates hidden-world reachability from the epistemic status induced by an agent's observation history, and it constructs paired worlds whose reachability differs while every permitted observation available to the agent agrees. A finite-system proposition identifies the regime in which such a pair cannot exist: deterministic monotone resource dynamics, complete Boolean requirement probes, and a budget sufficient to saturate the operation catalogue. The benchmark then breaks that completeness assumption with an opaque resource field while keeping the budget ample.

The resulting evaluation separates resource accounting from reasoning about observability. In a matched experiment that holds the model, prompt wording, report menu and family set fixed and moves only the budget and the observation channel, budget-horizon twins are reported correctly on 58/59 runs and opaque-field twins on 28/60 runs (46.7%; 95% CI [34.6%, 59.1%]): a 52-point separation between two conditions whose licensed report is the same word. An ample budget on its own disturbs nothing, with 60/60 where committing is licensed. The same ordering appears across three models in source blocks collected earlier under different prompt revisions (360/360 against 90/300), which we report as descriptive. A controller that preserves an unresolved observation scores 120/120 on the opaque condition, while the ablation that collapses unresolved into absent emits the wrong state on 120/120 of the same runs. The reporting distinction transfers to ten real-execution twin pairs, from 0/590 under the base menu to 589/590 under the disambiguated menu. These results establish feasibility identifiability as a separate evaluation target for tool-using agents.

1 INTRODUCTION

We introduce an executable benchmark for a question that task-completion scores leave implicit: *what has an agent established about whether a task is reachable?* A scientific or engineering agent can encounter an absent package, a hidden authorization state, an unverifiable unit convention, or a resource whose relevant field the environment does not expose. These cases can produce the same failed execution while supporting different reports.

The paper makes four contributions. **First**, it formalizes feasibility reporting over an information set, so hidden-world reachability and the agent's epistemic report are separate objects. **Second**, it gives a twin-world construction with executable certificates: one member is reachable, the other is unreachable, and every permitted observation history available to the agent is identical. **Third**, it states and proves a budget-saturation proposition that fixes when the benchmark's `unknown` state can legitimately exist: under monotone dynamics, complete probes, and a saturating budget it cannot, so a twin pair requires an incomplete observation interface. The proposition is an instrument for benchmark validity rather than a result about a new class of systems, and the opaque-field condition is the experiment it licenses. **Fourth**, it evaluates a concrete representation repair—preserving

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

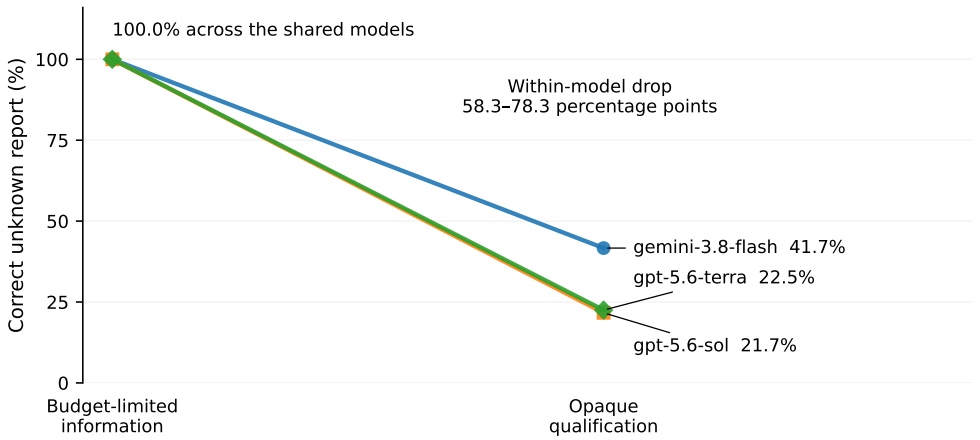


Figure 1: Model-level reporting profiles under the disambiguated menus. Each line joins the same model across budget-horizon and opaque-field conditions. Labels give correct-report rates and the range of within-model differences. These blocks were collected under different revisions of the prompt and show that the ordering reproduces across models; the estimate with the interface held fixed is in Section 5.1.

`unresolved` and preventing collapse into `absent`—and a report interface that separates artifact delivery from qualification.

The central empirical result is a source-of-uncertainty separation, estimated in a matched block that varies the budget and the observation channel while holding the model, the prompt wording, the report menu and the family set fixed. Reporting is at the ceiling when the agent's own budget is what stops it and far below the ceiling when a contract-relevant field is unreported, and an ample budget on its own changes nothing. Section 5 gives the cells and the reproduction across models; Section 6 tests the observation representation.

2 Related Work and Novelty Boundary

The new object in this paper is a mechanically certified *feasibility-identifiability* test for tool-using agents. It connects four established lines of work while assigning each a distinct role.

Feasibility awareness and abstention. FeasiGen evaluates whether tool-using agents recognize tasks made infeasible by masking critical tools (Cheng et al., 2026). AgentAbstain studies when agents should refrain from acting across executable sandbox scenarios (Liu et al., 2026). SSTA-32 distinguishes support states and evaluates overcommitment under limited support (Unlu, 2026). CAR-bench evaluates uncertainty handling, capability awareness, and disambiguation in multi-turn tool use (Kirmayr et al., 2026). Agentic Abstention treats stopping as a sequential decision, asking whether further interaction is likely to help across web, terminal, and question-answering environments (Luo et al., 2026). These studies establish that acting under insufficient capability or information is a measurable failure mode. Our benchmark supplies a certificate for a narrower question: whether two worlds with different reachability remain observationally indistinguishable to the agent.

Clarification and uncertainty-aware action. KnowNo calibrates LLM-based planners so that robots request help when uncertainty warrants it, using conformal prediction to control task-completion risk (Ren et al., 2023). Learning to Ask studies tool use under unclear instructions and measures whether agents recover missing user information instead of inventing it (Wang et al., 2025). AskToAct trains self-correcting clarification for ambiguous or incomplete tool requests (Zhang et al., 2025). These works treat uncertainty as a trigger for information acquisition or help-seeking. Our construction makes the availability of discriminating information itself an executable property of the environment.

Partial observability and diagnosability. Belief-state reasoning under partial observability predates LLM agents. Diagnosability planning for discrete-event systems explicitly reasons from an initial belief state and uses twin-plant constructions to test whether observationally equivalent behaviors can be distinguished (Ibrahim et al., 2017). We use that lineage as the formal precedent for indistinguishable worlds. Among foundation-model evaluations, Theory of Space asks whether models construct and revise a spatial belief from sequential partial observations by exploring (Zhang et al., 2026). That line asks whether an agent can reduce partial observability through exploration; we ask whether the remaining uncertainty is resolvable at all under the permitted observation interface, which is a property of the environment rather than of the policy. The contribution here is the benchmark object built on top of it: reachability certificates, observation-equivalence certificates, agent-facing report states, and executable scoring in resource-acquisition tasks.

Environment repair and scientific agents. EnvBench, SetupBench, Repo2Run, DeployBench, and ResearchEnvBench evaluate environment construction and research-artifact execution (Eliseeva et al., 2025; Arora et al., 2025; Hu et al., 2025; Wang et al., 2026a;b). ScienceAgentBench, SciCode, CORE-Bench, and AstaBench provide executable scientific tasks and programmatic evaluation (Chen et al., 2025; Tian et al., 2024; Siegel et al., 2024; Bragg et al., 2026). AndroidWorld scores interactive tasks from device state with per-task initialization and success checks (Rawles et al., 2025), and τ-bench scores tool-agent-user interaction by comparing the final database state against an annotated goal state while measuring reliability across repeated trials (Yao et al., 2025). We inherit executable verification, resource acquisition, and judge-free scoring from this literature, and we report repeated draws on the same family for the same reason τ-bench reports repeated trials. The new measurement asks whether the agent's final feasibility claim is licensed by its observation history.

3 Feasibility Reporting as an Epistemic State

Our formalization separates the state of a hidden world from what an agent can establish about that state.

Let W be the finite set of hidden worlds consistent with the task interface, A the permitted operation catalogue, C the original task contract, and B the run budget. Let R be the finite universe of resource facts named by the initial state, contract, and catalogue. A synthetic state is a subset of R; operations only add declared facts from R and never instantiate fresh identifiers. Hence the synthetic state space is finite, with at most $2^{|R|}$ resource states before budget and controller state are considered. For a world $w \in W$, define catalogue reachability

$$\rho(w) = \mathbf{1}\{\exists \sigma \in A^* : \sigma \text{ produces an artifact satisfying } C\},$$

where the reachability search removes the run-budget bound while retaining the declared catalogue and permissions. Thus $\rho(w)$ is an ontic property of the hidden world.

After an action-observation history h, let

$$I(h) = \{w \in W : w \text{ is observationally consistent with } h\}$$

be the agent's information set. The report target is

$$\Gamma(h) = \begin{cases} \texttt{ready}, & \rho(w) = 1 \ \forall w \in I(h), \\ \texttt{unreachable}, & \rho(w) = 0 \ \forall w \in I(h), \\ \texttt{unknown}, & \exists w_+, w_- \in I(h) : \rho(w_+) \neq \rho(w_-). \end{cases}$$

The report `unknown` denotes an information set containing worlds with different reachability.

3.1 Certified twin worlds

A certified twin pair (w_+, w_-) satisfies three conditions: $\rho(w_+) = 1$ by a replayed witness; $\rho(w_-) = 0$ by exhaustive catalogue search; and every affordable permitted action sequence induces the same agent-visible observation history in both members. The agent-visible projection is computational: each observation is reduced to its public action identifier, outcome kind, and normalized payload. Synthetic timing, process identifiers, random identifiers, and evaluator-only state are outside that projection. The third condition is checked by exhaustive lockstep enumeration or bisimulation over

the finite synthetic environment. The agent receives one member selected by a held-out draw and never receives the member identity.

The construction turns calibration into an executable property. If the observation history is shared by a reachable and an unreachable world, the information set contains both and $\Gamma(h) = \texttt{unknown}$ by definition.

3.2 BUDGET-SATURATION PROPOSITION

Catalogue saturation establishes reachability when resource dynamics and probes meet the following conditions. The statement is deliberately modest: for a finite monotone system with complete probes and a saturating budget, the conclusion is close to the definitions. Its role here is instrumental. It says that an instance admitting the report `unknown` must violate one of the three assumptions, which is what makes the opaque-field construction a manipulation of a named factor rather than an unexplained difficulty.

Proposition. Consider a deterministic finite resource system in which (i) operations have positive prerequisites and fixed additive effects, so execution preserves previously satisfied resource facts; (ii) every requirement has a complete Boolean probe whose answer exposes whether that requirement is satisfied; and (iii) the budget is sufficient to iterate every catalogue operation to a fixpoint and probe the goal. If two worlds are observationally indistinguishable under all affordable action sequences, then they agree on catalogue reachability.

Proof sketch. Repeatedly apply every enabled operation until no new resource fact is added. Because states are subsets of the finite resource universe R and each successful operation is monotone, this process reaches a fixpoint after finitely many state changes. Under the saturation budget the fixpoint is affordable, and complete Boolean probing reveals whether the goal requirement holds there. Any two worlds that disagree on reachability therefore differ in the final probe observation, contradicting observational indistinguishability. Appendix A gives the full argument and the executable bounded check used with the synthetic systems.

The proposition identifies the factor manipulated by the opaque-field condition: the budget remains sufficient to saturate the catalogue, while one contract-relevant resource field returns `unresolved`. The reachable and unreachable members then remain observationally equivalent because the discriminating field is absent from the observation channel.

4 BENCHMARK CONSTRUCTION

The benchmark operationalizes feasibility reporting with three independently checked objects: a recovery witness for reachable worlds, an exhaustive catalogue certificate for unreachable worlds, and an observation-equivalence certificate for unknown reports.

Each task specifies a typed contract over capability, version, scope, quality, provenance, and output semantics. Acquisition operations form a finite directed hypergraph with explicit preconditions, costs, outcomes, and postconditions. The frozen family set contains 120 task families spanning numerical simulation, statistical analysis, and scientific workflow tasks. Perturbations change resource state while preserving the original contract.

Budget-horizon twins. The first twin construction places the distinguishing action beyond the run budget. This condition tests whether the reporting interface separates *unaffordable* from *catalogue-unreachable*.

Opaque-field twins. The second construction gives the agent enough budget to probe requirements and exhaust the operation catalogue. A single contract-relevant field returns `unresolved` in both twin members. One member already holds a qualifying resource; the other holds a resource that differs only on the opaque field. Every operation that could replace the resource is disabled symmetrically, and bisimulation checks that every reachable continuation produces the same agent-visible observation.

Scoring. The evaluator maps structured terminal actions into the benchmark report vocabulary and scores task artifacts with deterministic tests. The mathematical $\Gamma(h)$ is the normative target over an information set; the scorer does not reconstruct all conceivable worlds at run time. Instead, benchmark admission supplies an evaluator-only witness, exhaustion certificate, and, for unknown instances, an observation-equivalence certificate that jointly determine the licensed report. Scientific scope is derived from execution receipts and typed resource provenance. No model judges another model's output. A restricted artifact can be delivered while qualification of the original contract remains unknown; Section 6 treats delivery and qualification as separate report dimensions.

4.1 Evaluation protocol

The live-agent comparison holds the public task contract, operation catalogue, family identity, and terminal report schema fixed within each paired condition. Models receive the same permission boundary and the same structured observations. The primary estimate is collected in matched sweeps under one pinned interface specification, with the prompt wording recorded in every row, so the cells compared in Section 5.1 differ in the manipulated factors and in nothing else the harness controls. The earlier aggregate blocks were collected under different revisions of the disambiguated prompt, which the rows of that period do not record, so their pooled contrast is reported as descriptive.

The analysis reports observed run counts and model-specific rates. A task family is the recurring problem unit. Run-level Wilson intervals in the appendix are descriptive summaries of the recorded counts.

The reporting vocabulary is tested with paired specificity conditions. A report-menu change that merely increases uncertainty responses would improve an unknown-only endpoint while damaging cases where commitment is warranted. The specificity evaluation includes catalogue-exhausted tasks, where `unreachable` is the licensed report, and fully verified tasks, where `ready` is licensed. We therefore interpret the menu intervention as a reporting-interface effect rather than evidence that the model has acquired a latent notion of epistemic uncertainty. Section 5.3 reports the paired counts for the three-valued menu, and Appendix B reports them for the four-valued intervention.

5 Results

The central result is a 52-point separation between uncertainty that follows from the agent's budget ledger and uncertainty that requires reasoning about an incomplete observation channel, measured with everything else about the interface held fixed.

5.1 Source of indistinguishability, with the interface held fixed

The primary estimate comes from a matched block: one model (`gemini-3.8-flash`), one prompt wording, one report menu, one submit vocabulary and one family set, with the run budget and the observation channel as the only factors that move. Budget-horizon twins are reported correctly on 58/59 runs and opaque-field twins on 28/60 runs (46.7%; 95% CI [34.6%, 59.1%]), a 52-point separation between two conditions whose licensed report is the same word. Tightening the budget on an opaque twin does not recover the loss (30/60), and an ample budget on its own disturbs nothing: where the catalogue is exhausted and `unreachable` is licensed, the same agent answers 60/60. The factor that carries the separation is therefore the observation channel rather than the budget.

Figure 2 maps the four cells by recorded condition budget, median interaction length and correct-report rate. Two cells license `unknown` and one licenses `unreachable`, so the cells are not interchangeable observations of one dependent variable; what the matched block removes is the possibility that a cell differs because of the day, wording or model it was run under.

The two conditions also differ in what they demand of the interaction. Budget-horizon success can be obtained from a rule over the agent's own resource accounting, and the recorded interaction is short. The opaque condition requires the agent to represent the fact that a contract-relevant field remains unresolved after the catalogue has been explored, and the recorded interaction is an order of magnitude longer.

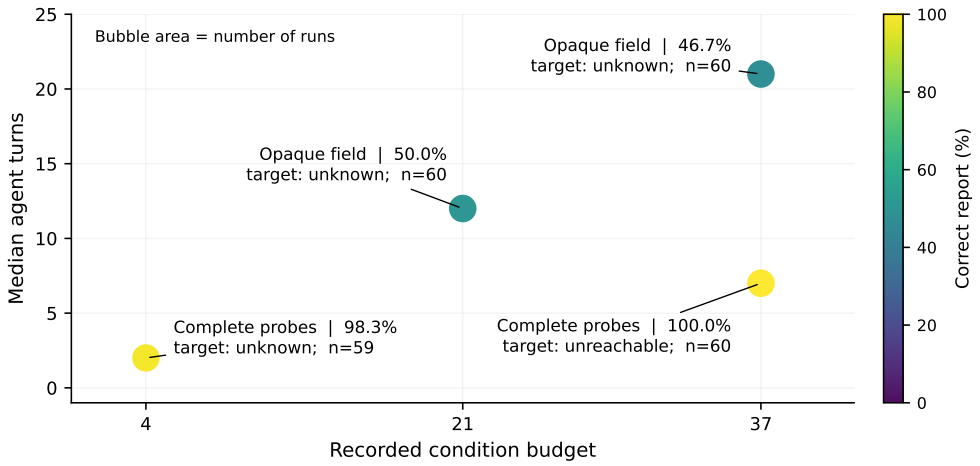


Figure 2: The matched cells for `gemini-3.8-flash`: one prompt wording, one menu and one family set, with the budget and the observation channel as the only factors that move. Position gives the recorded condition budget and median agent turns; bubble area gives the run count, and fill gives correct-report rate. Two cells license `unknown` and one licenses `unreachable`, so the cells share a scoring rule rather than a single dependent variable.

A trajectory-length sensitivity check goes in the opposite direction of a simple long-horizon account: within the two matched opaque cells, correct `unknown` reports occur on 19/62 runs at or below the pooled median of 13 turns and 39/58 above it. Turn count is post-treatment, so this split is descriptive rather than a causal adjustment; it only shows that the opaque errors are not concentrated in the longer trajectories.

5.2 Model-level structure

The same ordering appears in the earlier source blocks, which cover three models: budget-horizon reporting reaches the ceiling under the disambiguated menu on 360/360 runs, while opaque-field reporting stays at 90/300 (30.0%, [25.1%, 35.4%]) and is model-dependent, with no model's gap below 58 points. Those blocks were collected under different revisions of the disambiguated prompt, and the revision is not recorded in the older rows, so they are reported as a descriptive reproduction of the matched estimate rather than as a second measurement of it. The committed per-model rows are listed in Appendix B.

Figure 3 resolves the opaque-field outcomes into explicit uncertainty, catalogue-unreachability declarations, and artifact delivery, for every model measured under both menus. The models start in different places: one is near the unreachable corner and another near artifact delivery, so the same condition produces different wrong answers. The clarified menu moves all of them toward `unknown`. The appendix retains every recorded row, including structurally uninterpretable terminal actions in an `Other` column and source blocks for which a matching base-menu run was not collected; unpaired rows are not drawn in the simplex.

5.3 The menu does not merely induce abstention

The reporting gap is read under a menu that names `unknown`, so one competing account of the effect is label priming: naming the third state makes agents select it wherever they are unsure. That account predicts a loss on tasks where committing is licensed. The specificity condition measures exactly that loss. On catalogue-exhausted tasks, where `unreachable` is the correct report, three models answer 45/45 on 45 families under the base menu and 44/45, 44/45 and 39/45 under the disambiguated menu. The menu moves reporting on the twin conditions while commitment survives where commitment is warranted, which is the pattern a global shift toward abstention does not produce. Appendix B gives the per-model counts, including the residual `unknown` reports that the intervention does cost.

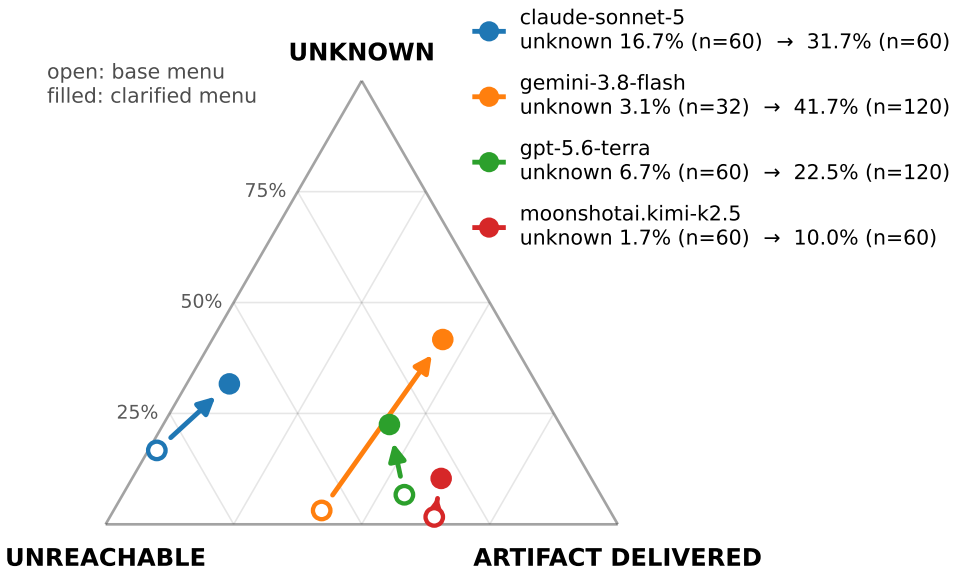


Figure 3: Terminal-report composition on opaque-field tasks. Triangle coordinates sum the shares of unknown reports, unreachable declarations, and artifact deliveries to unity. Arrows run from each model's base-menu composition (open marker) to its clarified-menu composition (filled). Shares are taken over the terminal actions the structural mapper classifies, with the denominator in the key; Appendix B keeps the unclassified ones. Artifact delivery combines original and restricted submissions and is not a task-success label; the arrows are group-level distributions, not task transitions.

5.4 Real execution

The reporting effect transfers from finite synthetic resource graphs to ten executable twin pairs with real process execution and deterministic scorers. Each pair keeps the construction and changes the reason the contract cannot be met: an absent analysis script, a distribution missing from the offline mirror, a distribution mirrored only at a version the contract excludes, a library that installs and does not expose the function the contract names, measurements in the wrong units, an absent provenance record, a checksum manifest that disagrees with the data it names, a series that covers too short an interval, data written in another locale's convention, and a workspace that will not accept the artifact. Every pair is admitted by execution before any paid run: reference witness, exhaustion of *every* ordering of the catalogue in the unreachable member, indistinguishability within the twin budget, and separation at the cheapest solving cost.

Across the committed transfer runs, the base menu yields 0/590 correct reports and the disambiguated menu yields 589/590. Every pair is measured on four models, at a replicate count that differs between the pairs the study began with and the ones added later; Appendix B gives the counts by pair and by model. Calls that returned a provider error are recorded as transport errors and excluded from the denominators rather than scored.

6 Observation Representation and Report Factorization

The intervention identifies where the opaque-field error enters the agent stack. The proposed controller preserves the observation value `unresolved`; the matched ablation maps the same observation to `absent`. On the opaque-field condition, the preserved representation yields 120/120 correct reports, while the collapsed representation emits the same catalogue-unreachable report on 120/120 runs. Appendix B gives the remaining scripted-arm counts.

The result locates the mechanism upstream of the final vocabulary: once an unresolved observation has been collapsed into absence, catalogue exhaustion supports the wrong epistemic conclusion. Preserving the third observation keeps both reachable and unreachable worlds in the information set.

An explicit uncertain-delivery target permits an artifact to be returned together with an `unknown` qualification report. The paired live-agent experiment evaluates this interface for `gemini-3.8-flash` and `gpt-5.6-terra`. A factorized delivery–qualification interface expresses the corresponding reporting semantics. Appendix B reports the paired outcomes together with specificity on conditions in which commitment is licensed.

7 Discussion

The evidence supports feasibility identifiability as a distinct property of tool-using agents. Budget-aware abstention and observation-aware calibration can produce different behavior under the same task family, operation catalogue, and terminal report schema. The opaque-field construction isolates that difference by making exhaustive action affordable while withholding one contract-relevant observation.

Implications for agent evaluation. A benchmark that records only task success or terminal abstention can combine two different information states: established unreachability and unresolved reachability. Reporting the observation history, its unresolved fields, and the epistemic feasibility state makes that distinction testable. The same representation is useful for deployed agents because it prevents an unavailable observation from being silently reinterpreted as a negative observation.

Scope and inherited material. Resource-state tracking, environment repair, belief-state reasoning, twin-plant diagnosability, abstention, clarification, executable scoring, and specification-preservation checks are established ingredients. This paper uses those ingredients to build a feasibility-reporting instrument with executable reachability and indistinguishability certificates. The catalogue bounds the reachability claim; open-world alternatives outside that catalogue are outside the evaluator's state space. The proposition assumes deterministic transitions over a finite fact universe, monotone additive effects, complete probes, and a saturation budget. Continuous state, stochastic transitions, irreversible resource deletion, and operations that create unbounded fresh objects require a different argument. The synthetic suite represents computational resources; autonomous wet-lab and clinical execution are outside the evaluated system. The real-execution transfer contains ten independently constructed twin pairs, which is wider than the three it began with and still far narrower than the task diversity of a deployment. Its equivalence check is weaker than the synthetic certificate: it enumerates bounded command sequences over an agent-visible projection and excludes stderr, timing, and other process metadata that could reveal world identity. The controller intervention tests the effect of preserving `unresolved` in this representation; it is a mechanism diagnostic, not a claim of algorithmic superiority over uncertainty-aware reinforcement-learning methods whose training objective and action interface differ from this benchmark. The evaluated models are frontier API models, and that is a scope condition rather than a sampling choice: two open-weight models run on the same condition reached a terminal report on 0 of 160 runs, spending 15% to 61% of their turns on verbs the schema does not admit (Appendix B.6). A rate cannot be read from a model that does not report, so this paper measures calibration among agents that hold the action interface. The matched block is one model on 60 of the 120 families, so it establishes the separation inside one interface rather than across the model population; the three-model blocks that reproduce the ordering differ in prompt revision and are read as descriptive. The matched cells also carry unequal budgets and two different licensed reports by construction, since a budget-horizon twin requires a binding budget and an exhausted catalogue requires an ample one. The ternary arrows summarize menu-level distributions with unequal denominators. Family-level transitions and sampling distributions require the underlying trajectories. The accompanying archive contains the certification pipeline, the scored rows, and the rendering code, and excludes the per-turn model transcripts.

8 Conclusion

Certified feasibility reporting separates what a hidden world permits from what an agent's observations establish. Twin-world certificates make `unknown` mechanically scoreable, the budget-saturation

proposition identifies the completeness assumptions that remove that epistemic state, and the opaque-field experiment shows that frontier tool-using agents can behave differently when uncertainty comes from the observation channel and when it comes from their own budget. Preserving unresolved observations and separating delivery from qualification provide concrete interfaces for carrying that distinction through an agent system.

USE OF LARGE LANGUAGE MODELS

Large language models are the object of study and were also used as tools for coding and editing. An LLM-based coding assistant contributed to implementation and analysis code, and language models contributed to drafting and editing prose. Experimental labels, scores, tables, and figures are produced by executable code over committed result files; no language model serves as a judge or grader. The authors verified the submitted content and take responsibility for it. This disclosure follows the ICLR 2027 requirement to state AI use in the paper.

REPRODUCIBILITY STATEMENT

The supplementary archive contains the objects that decide the reported scores: the family generator, the twin constructions, the reachability oracle, the observation-equivalence check, the evaluator, the real-execution qualification, and the deterministic control suite. It also contains the benchmark exported as inspectable data, the scored rows behind every table, and the code that regenerates the tables, figures, and anonymous PDF from them. The tables and figures are generated rather than transcribed. Per-turn model transcripts and API credentials are excluded, so reproducing the live-agent cells requires the reader's own provider access; every offline component, including benchmark admission and certification, runs without one.

REFERENCES

Avi Arora, Jinu Jang, and Roshanak Zilouchian Moghaddam. Setupbench: Assessing software engineering agents' ability to bootstrap development environments, 2025. arXiv:2507.09063.

Jonathan Bragg, Mike D'Arcy, Nishant Balepur, Dan Bareket, Bhavana Dalvi, Sergey Feldman, Dany Haddad, Jena D. Hwang, Peter Jansen, Varsha Kishore, Bodhisattwa Prasad Majumder, Aakanksha Naik, Sigal Rahamimov, Kyle Richardson, Amanpreet Singh, Harshit Surana, Aryeh Tiktinsky, Rosni Vasu, Guy Wiener, Chloe Anastasiades, Stefan Candra, Jason Dunkelberger, Dan Emery, Rob Evans, Malachi Hamada, Regan Huff, Rodney Kinney, Matt Latzke, Jaron Lochner, Ruben Lozano-Aguilera, Cecile Nguyen, Smita Rao, Amber Tanaka, Brooke Vlahos, Peter Clark, Doug Downey, Yoav Goldberg, Ashish Sabharwal, and Daniel S. Weld. Astabench: Rigorous benchmarking of ai agents with a scientific research suite. In *International Conference on Learning Representations*, 2026. arXiv:2510.21652.

Ziru Chen, Shijie Chen, Yuting Ning, Qianheng Zhang, Boshi Wang, Botao Yu, Yifei Li, Zeyi Liao, Chen Wei, Zitong Lu, Vishal Dey, Mingyi Xue, Frazier N. Baker, Benjamin Burns, Daniel Adu-Ampratwum, Xuhui Huang, Xia Ning, Song Gao, Yu Su, and Huan Sun. Scienceagentbench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, 2025.

Liang Cheng, Mingsheng Cai, Jiuming Jiang, and Luo Mai. Do agents know what they can't do? evaluating feasibility awareness in tool-using agents, 2026. arXiv:2605.28532.

Aleksandra Eliseeva, Alexander Kovrigin, Ilia Kholkin, Egor Bogomolov, and Yaroslav Zharov. Envbench: A benchmark for automated environment setup, 2025. arXiv:2503.14443.

Ruida Hu, Chao Peng, Xinchen Wang, Junjielong Xu, and Cuiyun Gao. Repo2run: Automated building executable environment for code repository at scale, 2025. arXiv:2502.13681.

Hassan Ibrahim, Philippe Dague, Alban Grastien, Lina Ye, and Laurent Simon. Diagnosability planning for controllable discrete event systems. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 2017. doi: 10.1609/aaai.v31i1.10694.

Johannes Kirmayr, Lukas Stappen, and Elisabeth Andre. Car-bench: Evaluating the consistency and limit-awareness of llm agents under real-world uncertainty. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 40599–40618, 2026. doi: 10.18653/v1/2026.acl-long.1886.

Xun Liu, Yi Evie Zhang, Vira Kasprova, Parisa Rabbani, Pardis Sadat Zahraei, Tianyu Zhang, Ali Ebrahimpour-Boroojeny, and Varun Chandrasekaran. Agentabstain: Do llm agents know when not to act?, 2026. arXiv:2607.10059.

Han Luo, Bingbing Wen, and Lucy Lu Wang. Agentic abstention: Do agents know when to stop instead of act?, 2026. arXiv:2606.28733.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. Androidworld: A dynamic benchmarking environment for autonomous agents. In *International Conference on Learning Representations*, 2025. arXiv:2405.14573.

Allen Z. Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. Robots that ask for help: Uncertainty alignment for large language model planners. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *PMLR*, pp. 661–682, 2023.

Zachary S. Siegel, Sayash Kapoor, Nitya Nadgir, Benedikt Stroebl, and Arvind Narayanan. Corebench: Fostering the credibility of published research through a computational reproducibility agent benchmark, 2024. arXiv:2409.11363.

Minyang Tian, Luyu Gao, Shizhuo Dylan Zhang, Xinan Chen, Cunwei Fan, Xuefei Guo, Roland Haas, Pan Ji, Kittithat Krongchon, Yao Li, Shengyan Liu, Di Luo, Yutao Ma, Hao Tong, Kha Trinh, Chenyu Tian, Zihan Wang, Bohao Wu, Yanyu Xiong, Shengzhu Yin, Minhui Zhu, Kilian Lieret, Yanxin Lu, Genglin Liu, Yufeng Du, Tianhua Tao, Ofir Press, Jamie Callan, Eliu Huerta, and Hao Peng. Scicode: A research coding benchmark curated by scientists. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2024.

Eren Unlu. Don't start what you can't finish: A counterfactual audit of support-state triage in llm agents, 2026. arXiv:2604.16752.

Wenxuan Wang, Shi Juluan, Zixuan Ling, Yuk-Kit Chan, Chaozheng Wang, Cheryl Lee, Youliang Yuan, Jen tse Huang, Wenxiang Jiao, and Michael R. Lyu. Learning to ask: When llm agents meet unclear instruction. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, 2025.

Yuanli Wang, Yaoyao Qian, Yue Zhang, Hanhan Zhou, Jindan Huang, Tianfu Fu, Qiuyang Mang, Huanzhi Mao, Wenhao Chai, Wendong Fan, and Liqiang Jing. Deploybench: Benchmarking llm agents for research artifact deployment, 2026a. arXiv:2606.05238.

Yubang Wang, Chenxi Zhang, Bowen Chen, Zezheng Huai, Zihao Dai, Xinchi Chen, Yuxin Wang, Yining Zheng, Jingjing Gong, and Xipeng Qiu. Researchenvbench: Benchmarking agents on environment synthesis for research code execution, 2026b. arXiv:2603.06739.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. tau-bench: A benchmark for tool-agent-user interaction in real-world domains. In *International Conference on Learning Representations*, 2025. arXiv:2406.12045.

Pingyue Zhang, Zihan Huang, Yue Wang, Jieyu Zhang, Letian Xue, Zihan Wang, Qineng Wang, Keshigeyan Chandrasegaran, Ruohan Zhang, Yejin Choi, Ranjay Krishna, Jiajun Wu, Li Fei-Fei, and Manling Li. Theory of space: Can foundation models construct spatial beliefs through active exploration? In *International Conference on Learning Representations*, 2026. arXiv:2602.07055.

Xuan Zhang, Yongliang Shen, Zhe Zheng, Linjuan Wu, Wenqi Zhang, Yuchen Yan, Qiuying Peng, Jun Wang, and Weiming Lu. Asktoact: Enhancing llms tool use via self-correcting clarification. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 13484–13511, 2025. doi: 10.18653/v1/2025.emnlp-main.682.

A Proof and Executable Check for the Saturation Proposition

Proof. Let R be the finite resource-fact universe and let each state be a subset $S \subseteq R$. Every enabled operation has prerequisites in R and a fixed finite set of positive effects in R; executing it replaces S by $S \cup P$ for its declared effect set P. Along any execution, the state therefore increases monotonically in the finite lattice 2^R. Repeatedly applying every enabled operation whose prerequisites are satisfied must reach a fixpoint S^* after finitely many strict state increases. Assumption (iii) makes one such saturation sequence and a final goal probe affordable.

If the contract goal is catalogue-reachable, some finite operation sequence produces a state satisfying it. Monotonicity implies that every fact produced by that sequence is contained in the saturation fixpoint, so the goal is satisfied in S^*. Conversely, if the goal is satisfied in S^*, the finite saturation sequence is itself a witness of catalogue reachability. Catalogue reachability is therefore equivalent to the Boolean goal probe returning true at the fixpoint. If two worlds disagree on catalogue reachability, their final complete goal probes differ, yielding a permitted observation sequence that distinguishes the worlds. The contrapositive gives the proposition. □

Executable check. The checker enumerates bounded finite resource systems satisfying these assumptions and searches for a counterexample: a pair that is observationally indistinguishable under a saturation budget while disagreeing on catalogue reachability. Each generated world specifies initial resource facts, enabled operations, deterministic postconditions, and complete Boolean probes. The checker computes the monotone fixpoint independently of the observation enumeration, then compares reachability across every indistinguishable pair. A discovered counterexample terminates the check and reports the pair. The executable check validates the implementation over its bounded enumeration domain; it is not used as a substitute for the proof above.

B Result Tables

The model-level counts resolve the source comparison, report composition, observation-channel ablation, delivery intervention, and real-execution transfer.

B.1 Source-of-uncertainty aggregate

Undecidable because	Models	n	correct unknown	95% CI
the budget cannot reach the answer	3	360	360/360 (100.0%)	[98.9%, 100.0%]
the environment will not give the answer	3	300	90/300 (30.0%)	[25.1%, 35.4%]

These are the earlier blocks, pooled across three models and across the prompt revisions they were collected under. They reproduce the ordering of the matched estimate in Section 5.1; they do not re-measure it.

B.2 Per-model source detail

Undecidable because	Model	Menu	n	correct unknown	95% CI
budget horizon	gemini-3.8-flash	base	120	2/120 (1.7%)	[0.5%, 5.9%]
budget horizon	gemini-3.8-flash	disambiguated	120	120/120 (100.0%)	[96.9%, 100.0%]
budget horizon	gpt-5.6-sol	base	120	1/120 (0.8%)	[0.1%, 4.6%]
budget horizon	gpt-5.6-sol	disambiguated	120	120/120 (100.0%)	[96.9%, 100.0%]
budget horizon	gpt-5.6-terra	base	120	0/120 (0.0%)	[0.0%, 3.1%]
budget horizon	gpt-5.6-terra	disambiguated	120	120/120 (100.0%)	[96.9%, 100.0%]
opaque field	gemini-3.8-flash	base	32	1/32 (3.1%)	[0.6%, 15.7%]
opaque field	gemini-3.8-flash	disambiguated	120	50/120 (41.7%)	[33.2%, 50.6%]
opaque field	gpt-5.6-sol	disambiguated	60	13/60 (21.7%)	[13.1%, 33.6%]
opaque field	gpt-5.6-terra	base	60	4/60 (6.7%)	[2.6%, 15.9%]
opaque field	gpt-5.6-terra	disambiguated	120	27/120 (22.5%)	[15.9%, 30.8%]

B.3 OPAQUE-FIELD REPORT COMPOSITION

Model	Menu	n	Unknown	Unreachable	Original	Restricted	Other
claude-sonnet-5	base	60	10	49	1	0	0
claude-sonnet-5	disambiguated	60	19	36	5	0	0
gemini-3.8-flash	base	32	1	18	0	13	0
gemini-3.8-flash	disambiguated	120	50	16	1	53	0
gpt-5.6-sol	disambiguated	60	13	25	1	19	2
gpt-5.6-terra	base	60	4	23	6	27	0
gpt-5.6-terra	disambiguated	120	27	40	21	32	0
moonshotai.kimi-k2.5	base	60	1	21	37	1	0
moonshotai.kimi-k2.5	disambiguated	60	6	17	34	1	2

The `Other` column denotes terminal actions that the structural mapper could not assign to `unknown`, `unreachable`, or a submitted artifact. Base-menu sample sizes differ because the available base opaque-field source blocks were collected only for a subset of models and families; these rows are retained for transparency and are not pooled into the disambiguated headline estimate.

B.4 MATCHED BUDGET–OPACITY CELLS

Cell	Correct answer	n	correct	95% CI	median turns
budget 4, no opaque field	unknown	59	58/59 (98.3%)	[91%, 100%]	2
budget 21, opaque field	unknown	60	30/60 (50.0%)	[38%, 62%]	12
budget 37, opaque field	unknown	60	28/60 (46.7%)	[35%, 59%]	21
budget 37, no opaque field	unreachable	60	60/60 (100.0%)	[94%, 100%]	7

Every cell is one model under one prompt wording, one menu and one family set; the budget and the observation channel are the only factors that move. The median-turn column is the recorded interaction length, not a scored quantity.

B.5 OBSERVATION-CHANNEL ABLATION

Arm	opaque_field_unknown	correct
A1_static	unreachable_within_catalog 120/120	no
A2_selfrepair	ready 120/120	no
A3_workflow	ready 120/120	no
A4_envsetup	ready 120/120	no
A5_proposed	unknown 120/120	yes
B1_no_verification	unknown 120/120	yes
B2_no_producer	unknown 120/120	yes
B3_no_scope_check	ready 120/120	no
B4_no_freshness	unknown 120/120	yes
B5_retry_matched	unreachable_within_catalog 120/120	no
B6_no_opaque_channel	unreachable_within_catalog 120/120	no

Identifier	Removed mechanism
B1	acquisition/postcondition verification
B2	producer-aware scheduling
B3	full substitution/scope validation
B4	execution-time freshness verification
B5	three-valued reporting channel, with saved budget reassigned to retries
B6	opaque observation channel; `unresolved` is collapsed to `absent`

B.6 What the instrument requires of a model

Model	Menu	n	reached a terminal report	turns outside the vocabulary
us.meta.llama4-maverick-17b-instruct-v1:0	base	40	0/40	1032/1775
us.meta.llama4-maverick-17b-instruct-v1:0	disambiguated	40	0/40	1137/1802
us.meta.llama3-3-70b-instruct-v1:0	base	40	0/40	183/1235
us.meta.llama3-3-70b-instruct-v1:0	disambiguated	40	0/40	195/1250

Every rate in this paper is read off a terminal report, so a model that never reaches one is not measured by the same instrument as a model that does. These are two open-weight models on the opaque-field condition under both menus. Neither reached a terminal report on any of the 160 runs: each run ended at the turn cap or the run budget, and its recorded outcome came from the harness's structural fallback rather than from the model. The rightmost column counts turns whose action verb was outside the four the schema admits, which is what the runs spent their turns on.

B.7 Three-valued menu specificity

Model	Menu	n	correct unreachable	unknown
gemini-3.8-flash	base	45	45/45 (100.0%)	0
gpt-5.6-sol	base	45	45/45 (100.0%)	0
gpt-5.6-terra	base	45	45/45 (100.0%)	0
gemini-3.8-flash	disambiguated	45	44/45 (97.8%)	1
gpt-5.6-sol	disambiguated	45	44/45 (97.8%)	1
gpt-5.6-terra	disambiguated	45	39/45 (86.7%)	6

The condition is catalogue-exhausted: `unreachable` is the licensed report and `unknown` is wrong. The rightmost column counts the runs that answered `unknown` here, which is the price of the menu on this condition.

B.8 Delivery and qualification factorization

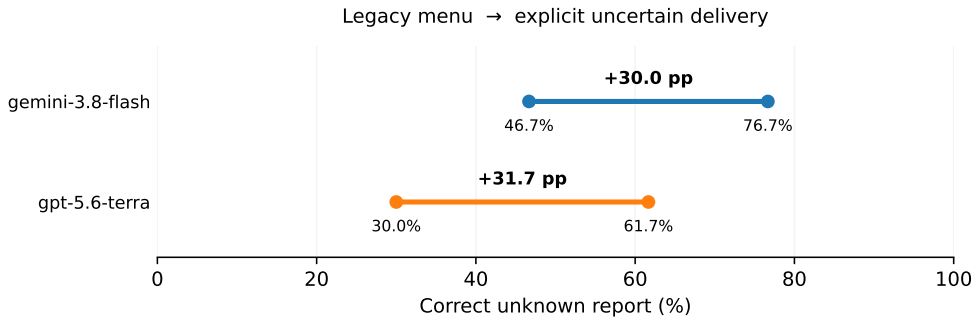


Figure 4: Within-model gains from an explicit uncertain-delivery option on the paired task families. Left and right endpoints give the legacy-menu and expanded-menu rates. Specificity counts accompany the intervention.

Model	n	Three-valued	Four-valued	Gain
gemini-3.8-flash	60	28/60 (46.7%) [35%, 59%]	46/60 (76.7%) [65%, 86%]	+30.0%
gpt-5.6-terra	60	18/60 (30.0%) [20%, 43%]	37/60 (61.7%) [49%, 73%]	+31.7%

Model	Condition	Correct answer	Four-valued
gemini-3.8-flash	bounded_unreachable	unreachable	30/30 (100.0%)
gemini-3.8-flash	ready	ready	22/30 (73.3%)
gpt-5.6-terra	bounded_unreachable	unreachable	28/30 (93.3%)
gpt-5.6-terra	ready	ready	17/30 (56.7%)

B.9 REAL-EXECUTION TRANSFER

Missing resource	Models	Base menu	Disambiguated menu
solver_script	4	0/80	79/80
withheld_distribution	4	0/80	80/80
wrong_units	4	0/80	80/80
version_gate	4	0/50	50/50
capability_absent	4	0/50	50/50
provenance_absent	4	0/50	50/50
integrity_mismatch	4	0/50	50/50
coverage_gap	4	0/50	50/50
encoding_locale	4	0/50	50/50
write_blocked	4	0/50	50/50
all 10 pairs		0/590	589/590

Counts pool the four models, and the model count travels with each pair so that an uneven column would be visible rather than hidden in a rate. The same rows split by model, where the pair count travels for the same reason:

Model	Pairs	Base menu	Disambiguated menu
gemini-3.8-flash	10	0/200	200/200
gpt-5.6-luna	10	0/130	130/130
gpt-5.6-sol	10	0/130	130/130
gpt-5.6-terra	10	0/130	129/130

Every pair carries an execution-time admission certificate. The reduced indistinguishability check is exact and covers all ten pairs; the exhaustive interleaving replay is a cross-check that runs where the enumeration is small enough to finish, which is four of them.

C EVALUATOR SEMANTICS

The evaluator operates on structured receipts; free-form claims do not determine the score. A resource requirement is a tuple over capability, version, scientific scope, quality, and provenance. A successful process exit does not mark a requirement satisfied unless postconditions validate the produced resource. Producer operations are distinguished from scientific consumer operations by the provenance graph: a producer creates a resource required by later computation, while a consumer's outputs enter the submitted artifact's provenance chain.

The factorized interface specifies separate delivery and qualification fields. Delivery records an original-contract artifact, a restricted artifact, or an empty submission. Qualification records `ready`, `unreachable`, or `unknown`. The measured menu intervention implements an uncertain-delivery target that maps to artifact delivery with an unknown qualification. The other terminal mappings retain their recorded semantics.

D BENCHMARK QUALIFICATION

Each benchmark family is admitted only after automatic qualification. Reachable conditions require a replayed witness. Unreachable conditions require exhaustive catalogue search. Twin conditions require observation-equivalence verification in addition to the reachability disagreement. Positive, negative, and metamorphic evaluator controls test the artifact scorer, and controller code is excluded from the independent reachability oracle.

Agent and evaluator processes receive different information. The agent receives the public contract, permitted operation catalogue, observations obtained during the run, and its own workspace. The evaluator retains the hidden twin member, reachability certificate, held-out tests, and scoring state. For synthetic twins, the comparison key is the observation kind, public action identifier, and normalized payload; random identifiers and execution timing are excluded. For real-execution twins, the certificate compares the declared agent-visible outcome projection over the bounded sequence set and records that stderr is excluded.