Atelier: A Human-Owned Protocol for Auditable AI-Assisted Research

Yuchong Geng [1] **Yuchen Lu** [2]

Abstract

Atelier is a file-backed protocol that routes AI-assisted research through three layers, augmenting the researcher rather than replacing them. The researcher remains principal, a manager agent scopes work and resolves agent reports, and worker or critic sub-agents operate inside bounded packets. Every consequential exchange is recorded in durable markdown files, producing a reconstructible chain from intent to claim. A pilot study with eight runs over two sweeps shows Atelier produces structurally better orientation and lifecycle outputs than a freeform baseline, with the gain holding across four model and effort settings. Github Repo.

1. Introduction

AI assistants now perform many of the tasks a research project needs. They propose experiments, draft code, summarize literature, and turn an open-ended question into a confident answer in minutes. The question for researchers is no longer whether a model can produce research-shaped artifacts. It is whether the loop that produced them can be trusted as a research argument. A useful protocol for AI-assisted research should help a researcher take a vague idea and end up with a principled, evidence-backed deliverable that they can stand behind and that a reviewer can inspect.

Two failure modes keep contemporary AI-assisted research from clearing that bar. The first is post-hoc optimization. A loop tries many ideas, keeps the one that scores well on a metric, and writes up the result. The research argument is constructed after a metric is hit, not carried through from assumptions to evidence. Autonomous-scientist platforms (Lu et al., 2024; Gottweis et al., 2025) exemplify the pattern. The second is brittle correctness. Hallucinated citations, drifted scope, and confidently wrong derivations slip through because nothing in the loop separates an agent's self-talk from external observation, or a report from accepted evidence. Benchmarks for ML and science agents (Chan et al., 2025; Chen et al., 2025) document how widely the brittleness appears. Both pitfalls share a root. Freeform agent loops let activity outpace the structure a research argument requires.

Existing systems address adjacent layers of the same problem but leave the protocol layer empty. Autonomous-scientist platforms aim to remove the human from the loop, which moves the problem rather than solving it for researchers who still own the project. Component-level tools, including literature and idea agents (Baek et al., 2025; Skarlinski et al., 2024) and execution frameworks (Yao et al., 2023; Wu et al., 2023; Jiang et al., 2025), produce useful artifacts and execution patterns, but none of them defines what counts as evidence or who holds authority over the project's claims. Agent-native research artifacts (Liu et al., 2026) address reproducibility of finished work, leaving the upstream process where claims are formed unstructured. None of those responses defines the protocol that turns intent into a reviewable argument.

Atelier supplies the protocol layer those responses leave open. The system is project-local and file-backed. Every consequential exchange between the human researcher, agents, and external observations is recorded in durable files rather than in chat history. The human researcher remains principal throughout, setting the research direction, accepting evidence, and approving the project's claims. Section 2 develops the design principles, the assumptions about how agent-assisted research actually fails, and the looped research lifecycle that organize the protocol.

By routing work and review through files, Atelier improves not wall-clock speed but the structure, auditability, and recoverability of research workflows. A project run through the protocol produces a reconstructible chain from intent to claim, with errors surfaced where they happen rather than left silent. Section 3 reports a pilot study where two sweeps probe these claims on a simulated research scenario.

2. Method

Atelier is designed to start where research starts, with a vague idea. The human researcher may bring an unrefined

[1]School of Electrical and Computer Engineering, Cornell University, Ithaca, NY, USA [2]Khoury College of Computer Sciences, Northeastern University, Boston, MA, USA. Correspondence to: Yuchong Geng <yg534@cornell.edu>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

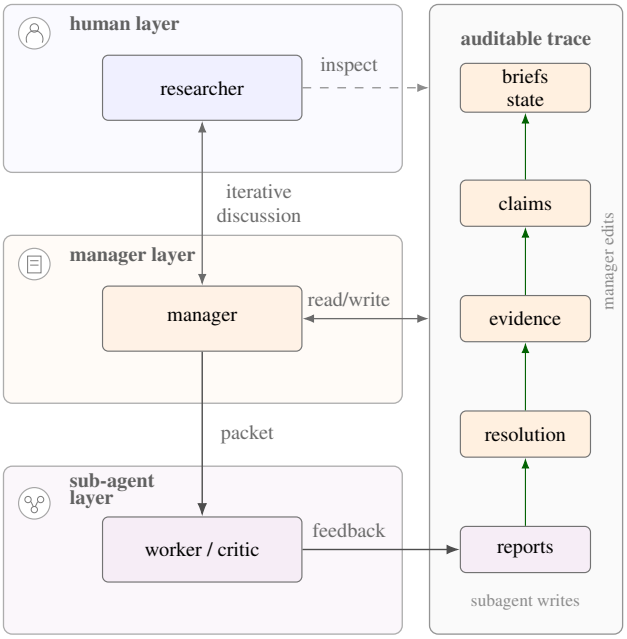

Figure 1. Atelier as a layered protocol for AI-assisted research, backed by human-auditable files. The researcher steers from the top layer, the manager owns canonical state and resolves work in the middle, and workers or critics operate inside scoped packets at the bottom. On the right, the audit trace records the chain. Subagents write reports, and the manager edits them into resolutions, evidence, claims, and researcher-facing briefs.

direction, a half-formed mechanism, or a question they cannot yet phrase precisely. Orientation, the protocol's first phase, turns that intent into a working problem statement, evaluation criteria, assumptions, a source map, and a next move. The system's broader output, across the full lifecycle, is a set of auditable evidence-backed claims. Figure 1 previews the layered protocol, showing the three role layers and the audit trace that the rest of the section develops.

The protocol organizes work around a research lifecycle that loops rather than progresses linearly. Any phase may revise an earlier one when evidence, sources, or critic findings demand it. Within each phase, agent contributions move through a recorded chain. A worker or critic submits a report, the manager's resolution of that report decides what becomes evidence, and claims summarize what the evidence supports. The looped lifecycle and the recorded chain together make research iterative, recoverable, and legible to a human reviewer.

Three principles guide our design.

1. **Governed discovery over metric chasing.** Research stays tied to problem framing, assumptions, falsification criteria, evidence, and claim scope. Metrics are evidence inside a research argument rather than the whole objective.
2. **Human-owned steering.** The researcher remains responsible for research direction, problem framing, and final claim acceptance. The agent may propose, critique, run bounded work, and surface tradeoffs, but it does not silently redefine the research question.
3. **Bounded agency with constructive dissent.** The assistant exercises the bounded autonomy of a mature PhD researcher working under a rigorous advisor. It raises objections, proposes alternatives, and identifies weak evidence, but only through scoped packets, recorded reports, and manager resolution.

2.1. Assumptions

The principles rest on three claims about how agent-assisted research actually fails.

Assumption 2.1 (Information flow)**.** Let Z denote the scientific target of interest. Let $U = (\theta, C_0, S_0)$ denote the agent's initial information, including model parameters, user context, and initially available sources. If an agent produces later turns $T_{1:k}$ only by processing U, then $Z \to U \to T_{1:k}$ is an information-processing chain, and the data processing inequality gives $I(Z; T_{1:k}) \leq I(Z; U)$.

Additional turns may make latent information easier to use, but they do not by themselves add information about Z. New information about Z enters the loop only through observations from outside the agent's self-processing path. Without explicit separation, each observation folds into the agent's state without leaving a distinct trace, so the loop's output T_k is indistinguishable from a function of the agent's initial context. The data processing inequality then bounds T_k by the information already in that context (model parameters, user-supplied context, and initially available sources). Atelier therefore records observations as artifacts distinct from agent reasoning, so the entry of new information into the chain is preserved.

Assumption 2.2 (Bounded sufficiency)**.** For many bounded research subtasks, a less capable or lower-budget model can match a stronger one once both are above the task-specific competence threshold and the harness, context, scope, and acceptance criteria are correct.

A naive workflow assigns every step to the strongest available model and asks it to hold the whole project state. Assumption 2.2 shifts the question from project-wide capability to task-level sufficiency. Once the manager scopes a bounded packet with explicit context, expected output, stopping rule, and escalation criteria, a smaller or lower-budget worker can complete it. Atelier therefore concentrates the manager's effort on packet framing rather than on selecting the strongest model for every step.

Assumption 2.3 (Sub-agent independence)**.** A sub-agent operating in a scoped packet with a fresh session, especially

one using a different model family, platform, prompt context, or tool surface, reduces correlated failure modes with the calling agent.

Workers and critics each operate in a scoped packet with a fresh session, focused on the packet's content rather than project-wide state. For critics, the fresh session also breaks correlation with the original agent's state, surfacing unsupported assumptions, missing evidence, scope drift, brittle reasoning, and weak experiment design that the original agent missed. The critic is not an oracle and can hallucinate or miss errors in turn, but the value is independence, not infallibility. Atelier therefore makes review a first-class operation, with critic packets optionally executed on a different model family or platform.

2.2. Research Lifecycle

An Atelier project begins with orientation, then moves through problem framing, related-work mapping, evaluation design, method development, experiment planning, evidence evaluation, claim update, and deliverable packaging. Orientation is the collaborative entry point to our workflow via an iterative dialogue between manager and user. The user opens the dialogue with a research idea at any level of specificity, from a finished draft to a one-sentence intent. The manager inspects available files and runs a light related-work check, drafting candidate framings and surfacing tradeoffs and gaps for the user to weigh, while the user contributes judgment on ambition, risk, and priority. The exchange sharpens the research direction and helps the user navigate the landscape. Subsequent phases each carry one or more work packets through the chain, accumulating evidence and updating claims.

The manager keeps a current focus that names the active phase, a dominant lens rather than a lock. A paper-writing focus may still trigger experiment planning when a draft exposes an evidence gap. A method development focus may still trigger related-work mapping when a baseline turns out to be unstable. The user can override the current focus at any decision point. The manager proposes the next move based on the current focus, the open packets, and the resolved evidence, rather than enforcing a fixed phase order.

2.3. Roles and Authority

Atelier separates human, manager, worker, and critic authority. The human owns research direction and final acceptance of high-level claims. The manager owns canonical project state (current focus, claims, evidence, work and review queues, resolutions, decisions, deliverable readiness, and user communication), dispatches bounded packets to workers and critics, and advises the human on next moves and tradeoffs. Workers and critics operate inside those packets, an arrangement that rests on Assumption 2.2 and Assumption 2.3. Once context, scope, and acceptance criteria are explicit, a smaller or lower-budget agent can complete a well-defined sub-task without holding the entire project state.

A worker performs one scoped task, writes a report, and stops. A critic, optionally on a different model family or platform, reviews one bounded target (a claim, evidence record, experiment design, mathematical argument, code change, paper draft, or steering decision), writes findings, and stops. Workers and critics may propose evidence records, claim edits, or state changes in their reports, but they do not update canonical state. This authority boundary keeps human-owned steering intact under critic disagreement and enacts bounded agency by scoping agent autonomy to the packet.

Roles are logical rather than tied to a specific execution context (model instance, thread, session, or chat). A manager may take on a worker packet itself for tasks that warrant close manager attention, such as high-consequence steps that may need frequent human intervention. The same file-backed protocol applies regardless of who plays the role or on which platform.

2.4. File-Backed Communication

Agent communication under `.atelier/` flows through a fixed chain of markdown artifacts (packet, progress, report, resolution) into the canonical update they produce. Each file has structured frontmatter, concise enough for agents to act on and readable enough for the human to follow directly. The manager creates `packet.md` for one unit of work, recording intent, scope, context, expected output, and approval policy. The assigned worker reads the instructions from `packet.md` and creates and updates `progress.md` for volatile operational state when work is long-running, external, costly, or hard to recover from chat alone. The worker or critic submits `report.md` as the unit's output. The manager digests sub-agents' `report.md` and writes `resolution.md` as the accepted interpretation.

The chain enforces Assumption 2.1 by giving an agent's processing and the manager's accepted reading distinct artifact types. The same chain serves as an audit trail for the human. Reports are not edited after submission, and disagreement is recorded in `resolution.md` rather than rewritten into the report. A later manager session can recover by reading the project manifest, current state, queue indexes, recent packets, submitted reports, and closed resolutions.

2.5. Evidence and Claims

The chain's report-evidence-claim distinction enacts Principle 1, governed discovery over metric chasing. Evidence

is the manager's accepted reading of a concrete artifact, result, source, observation, or experiment output. Claims summarize what the evidence supports, weakens, refutes, or qualifies.

Workers may run experiments, critics may find flaws, and managers may accept, reject, qualify, or supersede reports. Only the accepted interpretation becomes evidence, and claims remain scoped to the evidence records that support them. An autonomous experiment does not silently become a paper claim.

Negative and stale results are preserved. A failed run can be invalid, inconclusive, or informative depending on the manager resolution. Old claims can be marked historical or superseded instead of deleted. The goal is not to preserve every token of conversation, but the decisions and observations needed to audit the research argument.

3. Evaluation

The evaluation probes three method claims about workflow quality, inspectability, and recoverability.

Setup. We simulate a user interacting with two AI conditions, a freeform agent and Atelier. The research direction is NeuroGolf 2026 (Kaggle, 2026), an active Kaggle competition asking for the smallest neural networks that solve ARC-AGI grid transformations. A vague multi-turn prompt with scripted user replies drives interaction and holds user input constant across conditions. The baseline writes a final research plan from the prompt. Atelier runs the file-backed protocol over the same prompt, either through orientation alone or through the full manager-worker-critic loop. Outputs are scored on 20-point manual rubrics covering hidden-structure recovery, constraint capture, method framing, evaluation design, recoverability, and related dimensions. Section B lists both rubrics, the artifact inventory, and per-run scores.

The two sweeps probe two questions. First, does Atelier turn a vague research idea into a more actionable plan? Both sweeps test this, with the orientation sweep isolating Atelier's first phase and the lifecycle sweep running the full manager-worker-critic loop. Second, does the protocol hold under model variation? Each sweep covers four model and effort settings, probing Assumption 2.2, the claim that bounded packets carry weaker models.

Workflow quality. The rubric scores measure how well an output meets the desiderata of early research, such as a captured problem frame, evaluation criteria, and recoverable state. Atelier's outputs score higher than the baseline's in both sweeps. The mean rises from 16.25 to 19.5 out of 20 in the orientation sweep and from 16.5 to 19.25 in the lifecycle sweep. The improvement holds across all four model and effort settings, including the weakest.

Setting	Baseline	Atelier	Gain
Orientation sweep (/20)			
GPT-5.2, medium	15	19	+4
GPT-5.4-Mini, medium	15	19	+4
GPT-5.5, low	17	20	+3
GPT-5.5, medium	18	20	+2
mean across four settings	16.25	19.5	+3.25
Lifecycle sweep (/20)			
GPT-5.2, medium	16	19	+3
GPT-5.4-Mini, medium	18	20	+2
GPT-5.5, low	16	18	+2
GPT-5.5, medium	16	20	+4
mean across four settings	16.5	19.25	+2.75

Table 1. Pilot study results. See Section B for rubrics and per-run scores.

Auditability and recoverability. The lifecycle runs produced canonical state, briefs, packets, reports, and resolutions inventoried in Section B. The freeform baseline produced only a transcript and a final plan. The difference matters for two reasons. A researcher can read the chain that produced each claim, and a later agent can resume from canonical state without replaying the conversation. Several lifecycle runs also encountered model and platform failures during execution, providing an unplanned test of the recovery story. The file-backed state surfaced those failures and allowed recovery from canonical state, with the final claims remaining valid.

4. Discussion and Limitations

Rather than aiming to replace the researcher with an autonomous scientist, Atelier provides a protocol for routing AI-assisted research through layered abstractions and inspectable files. Our pilot evaluations show that rubric scores rise across both sweeps, lifecycle runs produce inspectable artifacts, and the protocol recovers from external failures. Atelier's value lies in the workflow discipline it provides, augmenting researcher judgment rather than replacing it.

In its current form, the pilot is small (eight runs total), manually scored, and tests Atelier on a simulated research scenario rather than a real, full-scale research project. The scorer is a project participant despite a rubric fixed in advance. The paper does not compare Atelier against other agent frameworks or against blind expert judges. Future work includes blind scoring, benchmarking against alternative agent stacks, validation on full-scale research projects with verifiable outcomes, and a thoughtfully designed UI that makes the project artifacts more readable.

References

Baek, J., Jauhar, S. K., Cucerzan, S., and Hwang, S. J. ResearchAgent: Iterative research idea generation over scientific literature with large language models. In Chiruzzo, L., Ritter, A., and Wang, L. (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6709–6738, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.naacl-long.342. URL https://aclanthology.org/2025.naacl-long.342/.

Chan, J. S., Chowdhury, N., Jaffe, O., Aung, J., Sherburn, D., Mays, E., Starace, G., Liu, K., Maksin, L., Patwardhan, T., Weng, L., and Mądry, A. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations*, 2025. doi: 10.48550/arXiv.2410.07095. URL https://openreview.net/forum?id=6s5uXNWGIh.

Chen, Z., Chen, S., Ning, Y., Zhang, Q., Wang, B., Yu, B., Li, Y., Liao, Z., Wei, C., Lu, Z., Dey, V., Xue, M., Baker, F. N., Burns, B., Adu-Ampratwum, D., Huang, X., Ning, X., Gao, S., Su, Y., and Sun, H. ScienceAgentBench: Toward rigorous assessment of language agents for data-driven scientific discovery. In *International Conference on Learning Representations*, 2025. doi: 10.48550/arXiv.2410.05080. URL https://openreview.net/forum?id=6z4YKr0GK6.

Gottweis, J., Weng, W.-H., Daryin, A., Tu, T., Palepu, A., Sirkovic, P., Myaskovsky, A., Weissenberger, F., Rong, K., Tanno, R., Saab, K., Popovici, D., Blum, J., Zhang, F., Chou, K., Hassidim, A., Gokturk, B., Vahdat, A., Kohli, P., Matias, Y., Carroll, A., Kulkarni, K., Tomasev, N., Guan, Y., Dhillon, V., Vaishnav, E. D., Lee, B., Costa, T. R. D., Penadés, J. R., Peltz, G., Xu, Y., Pawlosky, A., Karthikesalingam, A., and Natarajan, V. Towards an AI co-scientist, 2025. URL https://arxiv.org/abs/2502.18864.

Jiang, Z., Schmidt, D., Srikanth, D., Xu, D., Kaplan, I., Jacenko, D., and Wu, Y. AIDE: AI-driven exploration in the space of code, 2025. URL https://arxiv.org/abs/2502.13138.

Kaggle. The 2026 NeuroGolf championship. Kaggle competition, 2026. URL https://www.kaggle.com/competitions/neurogolf-2026. IJCAI-ECAI 2026 Competitions Track.

Liu, J., Pei, J., Huang, J., Si, C., Qu, A., Tang, X., Lu, R., Chen, L., Bai, X., Zheng, H., Chen, C., Chen, Z., Ye, H., Fu, Y., He, Z., Jin, Z., Zhang, Z., Sun, S., Harmon, M., Wang, J. D., Zeng, J., Sun, J., Wu, M., Zhou, B., You, C., Lu, S., Qiu, Y., Lai, F., Yuan, Y., Li, Y., Hong, J., Zhu, R., Chen, B., Pentland, A., Chen, A., Chowdhury, M., and Zhang, Z. The last human-written paper: Agent-native research artifacts, 2026. URL https://arxiv.org/abs/2604.24658.

Lu, C., Lu, C., Lange, R. T., Foerster, J., Clune, J., and Ha, D. The AI scientist: Towards fully automated open-ended scientific discovery, 2024. URL https://arxiv.org/abs/2408.06292.

Skarlinski, M. D., Cox, S., Laurent, J. M., Braza, J. D., Hinks, M., Hammerling, M. J., Ponnapati, M., Rodriques, S. G., and White, A. D. Language agents achieve superhuman synthesis of scientific knowledge, 2024. URL https://arxiv.org/abs/2409.13740.

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A. H., White, R. W., Burger, D., and Wang, C. AutoGen: Enabling next-gen LLM applications via multi-agent conversation, 2023. URL https://arxiv.org/abs/2308.08155.

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023. doi: 10.48550/arXiv.2210.03629. URL https://openreview.net/forum?id=WE_vluYUL-X.

A. Implementation Details

Adapter boundary. The protocol is agent-agnostic. Project files define the meaning of state, packets, reports, resolutions, evidence, and claims. Adapters define launch mechanics, prompts, tool calls, terminal commands, or manual handoff. Codex, Claude, Gemini, local scripts, and future tools participate as adapters around the same files. One project can use one manager, multiple workers, and fresh critics across different platforms without changing protocol semantics, and the file-backed state survives any change of platform.

Skill library. Atelier provides a library of skills, each defining a structured workflow that produces specific artifacts. Any agent role can invoke the relevant skill. Workers typically run the lifecycle-phase skills, critics run `second_opinion`, and the manager runs `init`, `steer`, `update_claim`, and `resolve_report` over canonical state.

Skill	Trigger
`init`	Start a new project or recover missing project state.
`orient`	Project lacks a stable problem frame, evaluation frame, source map, or next move.
`map_related_work`	Project needs a map of prior work, baselines, datasets, or field expectations.
`design_evaluation`	Project needs an evaluation frame before planning experiments or comparing methods.
`develop_method`	A proposed method, model, or artifact needs to become a working implementation.
`plan_experiment`	A settled evaluation question should become a bounded experiment packet.
`evaluate_evidence`	Accepted results, artifacts, or analyses should become a durable evidence record.
`update_claim`	Evidence, reviews, or framing require creating, revising, or retiring a claim.
`bridge_deliverable`	Claims, evidence, and audience need to be turned into a paper, report, or other deliverable plan.
`steer`	Choose the next research moves from durable state.
`second_opinion`	A claim, evidence record, design, or section needs bounded critique from a fresh critic.
`resolve_report`	A work or review item has a submitted report and needs manager resolution.

Project layout. An Atelier project lives under a `.atelier/` directory at the project root with the following layout.

```
.atelier/
|- project.yaml
|- state/current.md
|- briefs/
|  |- project.md
|  \- deliverable.md
|- resources/index.md
|- work/
|  |- pending/
|  |- active/<W-id>/
|  |  |- packet.md
|  |  |- report.md
|  |  \- resolution.md
|  |- reported/
|  \- closed/
|- reviews/
|  |- pending/
|  |- active/<R-id>/
|  |  |- packet.md
|  |  |- report.md
|  |  \- resolution.md
|  |- reported/
|  \- closed/
|- evidence/<E-id>.md
|- claims/<C-id>.md
\- logs/
   |- research-log.md
   \- decisions.md
```

Top-level files carry the project configuration, the current focus, the project and deliverable briefs, and the source map. The `work/` and `reviews/` queues organize their entries through `pending`, `active`, `reported`, and `closed` subdirectories. Each unit materializes as `packet.md`, an optional `progress.md`, `report.md`, and `resolution.md`. Evidence records and claims accumulate under `evidence/` and `claims/` keyed by identifier. All files are markdown with structured frontmatter, so an agent can act on them and a researcher can read them directly.

B. Evaluation Details

Each rubric dimension is scored as 0, 1, or 2. A score of 0 means absent or materially wrong. A score of 1 means partially present but vague, incomplete, or not actionable. A score of 2 means clear, task-specific, and useful.

Orientation rubric. The maximum score is 20. Severe errors are recorded separately for materially misleading claims, ignored executable-artifact constraints, invented official rules, the absence of a concrete first experiment, or, in the Atelier condition, material project record writes without explicit user approval.

Dimension	What the dimension measures
Hidden structure recovery	Recovers the vague idea as a per-task tiny executable-artifact problem for ARC-like transformations.
Constraint capture	Captures CPU limits, compactness, executable-artifact constraints, and possible static-shape or operation-set issues.
Method framing	Frames the problem as rule inference plus artifact emission, program synthesis, compiler design, or neural circuit synthesis.
Rationale and landscape	Explains why the framing might work, names nearby approaches, and avoids treating the landscape as settled without inspection.
Evaluation design	Separates correctness from compactness and defines support, weakening, or falsification criteria.
Gap-exposing tests	Proposes tests that could reveal failures of ordinary training, written solvers, overfitting, artifact constraints, or benchmark mismatch.
Question quality and interaction use	Asks decision-level questions when useful and uses replies to sharpen the frame rather than restating generic advice.
Bounded next steps	Produces a first action that can start on ordinary CPU resources.
Recoverability	Leaves artifacts or structured state that a future agent can resume from.
Claim and write-gate discipline	Avoids claiming benchmark performance or competition compliance, and in the Atelier condition respects the lack of write approval.

Lifecycle rubric. The maximum score is 20. Severe errors are recorded separately for materially misleading claims, ignored executable-artifact constraints, invented official rules, or the absence of a concrete first experiment.

Dimension	What the dimension measures
Hidden structure recovery	Recovers the vague idea as a tiny executable-artifact problem for ARC-like transformations.
Constraint capture	Captures CPU limits, compactness, executable artifacts, and static-shape or operation-set constraints.
Method framing	Frames the problem as rule inference, artifact emission, program synthesis, or neural circuit synthesis.
Evaluation design	Separates correctness from compactness and defines support or falsification criteria.
Bounded next steps	Produces a first action that can start immediately.
Use of interaction	Uses clarification turns to improve the direction.
Critique and revision	Surfaces risks and revises the plan in response to critique or new constraints.
Recoverability	Leaves artifacts or structured state that a future agent can resume from.
Claim discipline	Avoids claiming benchmark performance, solved ARC, or verified official competition compliance.
Workshop usefulness	Produces evidence relevant to correctness, reproducibility, workflow decomposition, and research steering.

Artifact inventory. The lifecycle sweep records the delegation and critique chain that the freeform baseline does not expose. Orientation-sweep runs produce the initialized project state only, since the simulated user never grants write approval.

Artifact	Freeform baseline	Atelier (lifecycle)
Transcript	recorded	recorded
Final research plan	recorded	recorded
Project state	absent	recorded
Project brief	absent	recorded
Work packet	absent	recorded
Worker report	absent	recorded
Manager work resolution	absent	recorded
Review packet	absent	recorded
Critic report	absent	recorded
Manager review resolution	absent	recorded
Deliverable brief	absent	recorded

Per-run scores. Tables 2 and 3 report the per-dimension scores for every run in the two sweeps.

Sample outputs. The preliminary orientation run uses the user prompt below, with three scripted follow-up replies, and ends with each condition's final-orientation summary. The boxes that follow reproduce the prompt and summaries from the run artifacts.

User prompt

Initial. I'm interested in whether small neural networks can solve ARC-style grid transformation tasks. I don't want a giant model. Ideally each task would have a tiny executable model that transforms inputs correctly. I'm not sure whether this is a learning problem, a program synthesis problem, or something else. Help me turn this into a research direction.
Reply 1. I care more about getting a concrete research direction and first experiment than about solving many tasks today. I only have ordinary laptop or CPU resources right now.
Reply 2. I want the final artifact to be something like a tiny executable model per task, not just a written solver. I am also curious whether this could eventually map to a constrained benchmark or competition.
Reply 3. Please make the next step something I could realistically start today.

Baseline

This should be framed as **task-specific neural program synthesis**, not ordinary neural training and not pure symbolic ARC solving.
The research direction is to search for a tiny executable model per task, with strict limits on parameter count, inference steps, and allowed features. The scientific question is whether ARC-style transformations can be compressed into very small grid-aware neural mechanisms that generalize from a few examples.
A realistic first experiment is a CPU-friendly toy ARC benchmark with generated tasks. Start with five transformation families, train one tiny coordinate-conditioned MLP per task, and evaluate exact-match accuracy on hidden generated examples. Keep the model small enough that the artifact itself is meaningful.
The immediate deliverable is not a full ARC solver. It is evidence about whether tiny per-task executable neural models are viable at all, and which inductive biases matter.

Setting	Condition	Hidden structure	Constraints	Method	Landscape	Evaluation	Gap tests	Questions	Next steps	Recovery	Write-gate	Total
GPT-5.2, medium	Baseline	2	1	2	1	2	1	1	2	1	2	15
	Atelier	2	2	2	2	2	2	1	2	2	2	19
GPT-5.4-Mini, medium	Baseline	2	1	2	1	1	1	2	2	1	2	15
	Atelier	2	2	2	2	2	2	1	2	2	2	19
GPT-5.5, low	Baseline	2	1	2	1	2	1	2	2	2	2	17
	Atelier	2	2	2	2	2	2	2	2	2	2	20
GPT-5.5, medium	Baseline	2	2	2	1	2	2	2	2	1	2	18
	Atelier	2	2	2	2	2	2	2	2	2	2	20

Table 2. Orientation sweep per-run scores.

Setting	Condition	Hidden structure	Constraints	Method	Evaluation	Next steps	Interaction	Critique	Recovery	Claim discipline	Workshop	Total
GPT-5.2, medium	Baseline	2	2	2	2	2	1	1	1	2	1	16
	Atelier	2	2	2	2	2	2	2	2	2	1	19
GPT-5.4-Mini, medium	Baseline	2	2	2	2	2	2	1	1	2	2	18
	Atelier	2	2	2	2	2	2	2	2	2	2	20
GPT-5.5, low	Baseline	2	2	2	1	2	2	1	1	2	1	16
	Atelier	2	2	2	1	2	2	2	2	2	1	18
GPT-5.5, medium	Baseline	2	2	2	1	2	2	1	1	2	1	16
	Atelier	2	2	2	2	2	2	2	2	2	2	20

Table 3. Lifecycle sweep per-run scores.

Atelier

The project frame is compression-constrained per-task neural induction for ARC-style grid transformations.

The research question is whether each ARC-style task can be represented by a tiny executable learned model that maps inputs to outputs, without relying on a giant pretrained model or only returning a written solver.

The proposed first method is a per-task tiny Neural Cellular Automaton or small convolutional transition system. Each task gets its own fitted parameters and rollout count. The executable artifact is the fixed runtime plus the learned task artifact.

The first experiment should be CPU-scale and start today. Select 5 to 8 ARC public training tasks by visual inspection. Train one tiny model per task from scratch. Evaluate exact-grid match, leave-one-example-out behavior where possible, serialized artifact size, parameter count, train time, inference time, rollout steps, and failure type.

The evaluation should include a compact symbolic baseline for simple color maps, copies, crops, and fills. This prevents the project from mistaking a weak learned artifact for progress when a smaller symbolic artifact solves the same task.

The likely eventual benchmark frame is NeuroGolf-style scoring: solve ARC-style tasks with the smallest executable per-task artifact under explicit CPU, size, and task-specific-engineering limits.