RULERS, NOT METERS: DISCOVERING FALSIFIABLE WORLD MODELS THROUGH ACTIVE VISUAL EXPERIMENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

A useful world model must predict the effects of new interventions, so it has to be identified, not merely fit. Existing evaluations of vision-language agents fix what the agent observes and cannot tell identification from fitting guesses to feedback. We study *active world-model identification* in **Eyes of Galileo**: from raw renders of 35 counterfactual physics scenes, an agent must build its own measuring references, propose an executable symbolic world model, and revise it when execution at dispersed targets exposes counterexamples. With GPT-5.6, such an agent identifies 63.4% of hidden world models (strict or relaxed), versus at most 44.6% for three visual-reasoning scaffolds under identical feedback, and dispersed counterexamples raise exact identification from 33% to 73% in a controlled revision study. The ability depends on the backbone (33.1% with Qwen3-VL-32B, 24.0% with Qwen3-VL-8B), and so does learning it: fine-tuning on successful GPT-5.6 episodes raises the 32B model's success on unseen scenes from 42.9% to 57.1%, but does not help the 8B model. Identifying world models from pixels thus hinges on both making evidence and learning to use it, a capability that fixed-observation evaluations cannot reveal.

1 INTRODUCTION

An agent that acts in the world needs a world model: an account of how outcomes change when it intervenes (Ha & Schmidhuber, 2018; LeCun, 2022). Most learned world models are fit to whatever observations they are given, but a model meant to hold under new interventions must be *identified*: its evidence has to be made measurable, and its predictions checked against outcomes the agent has not yet seen. Science offers the template. Galileo could not time a free fall, so he slowed it on an inclined plane, laid it against spatial and temporal marks from which a predictive relation could be read, and tested that relation at other inclinations; in less controlled settings, a scientist exploits a stable object, a shadow, or a periodic event already in the scene. We ask whether a vision-language agent can identify a world model this way from pixels alone: build references that turn images into

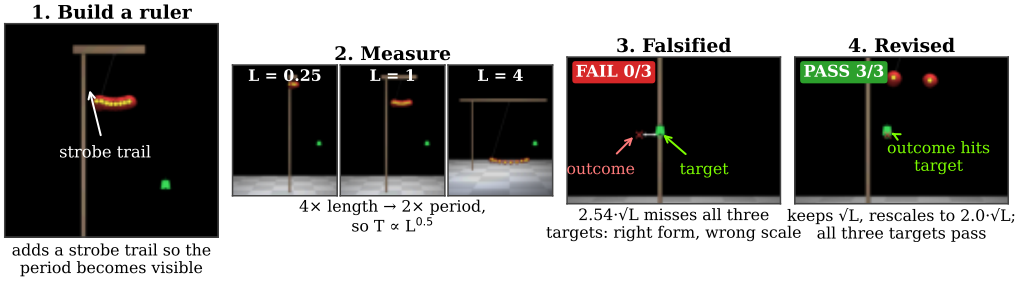


Figure 1: A GPT-5.6 agent discovering a counterfactual pendulum law ($T \propto L^{0.5}$) from pixels. It builds its own reference, reads the functional form from it, and is falsified by goal-conditioned execution at three dispersed targets, which corrects its coefficient without changing its form (Appendix D).

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.

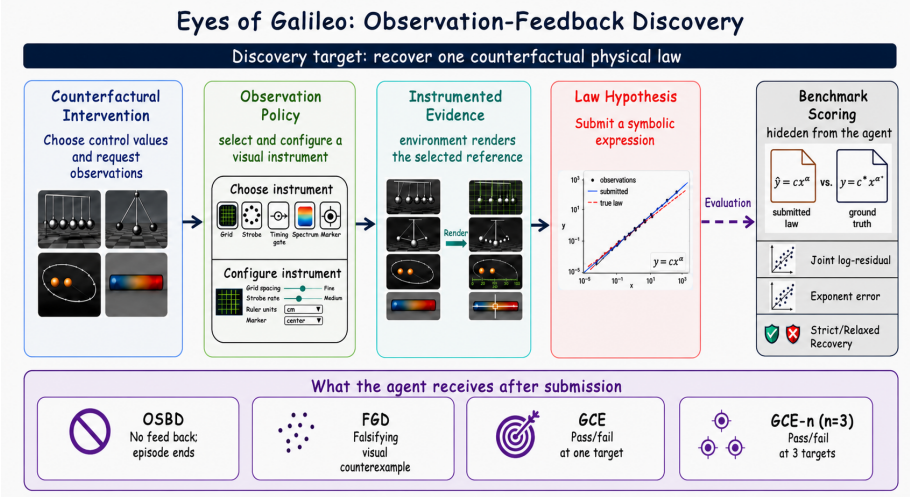


Figure 2: Active world-model identification in Eyes of Galileo. The agent intervenes, configures a visual reference, and converts rendered outcomes into a symbolic model. Feedback tests that model by executing it at dispersed targets (GCE-n, $n = 3$), and evaluation compares its predictions with oracle outcomes on a scene-specific grid. Numeric ground truth is never revealed to the agent.

comparable evidence, propose an executable model, and revise it when its predicted consequences fail.

Existing evaluations rarely test this step. Visual physics benchmarks such as PHYRE, CLEVRER, Physion, and PhysBench supply pre-rendered clips or labeled states (Bakhtin et al., 2019; Yi et al., 2020; M. Bear et al., 2021; Chow et al., 2025); law-discovery benchmarks such as NewtonBench, PhysGym, and LLM-SRBench supply numeric observations or simulator queries (Zheng et al., 2025; Chen et al., 2025a; Shojaee et al., 2025b); and DiscoverPhysics poses counterfactual worlds through a fixed interface (Wiemann et al., 2026). In each case the measurement interface is decided in advance. An agent can then succeed by requesting more views and fitting successive guesses to correctness feedback, and two questions remain untested: **(Q1)** can an agent construct evidence that makes a law identifiable from pixels, and **(Q2)** can it turn execution counterexamples into correct revisions?

We study active world-model identification: from raw renders of a counterfactual physical system, an agent must construct identifiable evidence, propose a symbolic law of an observable as a function of a control, and test that law through interventions. We study this problem in Eyes of Galileo, an environment of 35 annotation-free scenes (175 scene–setting cells, five hidden settings each) spanning mechanics, thermal physics and transport, fluids, waves, optics, and electromagnetism, in which hidden exponents, rates, and coefficients depart from textbook values so that recalling a familiar equation does not suffice. A compositional visual-instrument DSL lets the agent place grids, strobe trails, timing gates, and spectral bands without altering the dynamics, and goal-conditioned execution (GCE-n) inverts the agent's own law to hit n dispersed targets and returns the physical outcome. **Observation-Feedback Discovery (OFD)** is the closed loop the benchmark evaluates (Figure 2).

For a capable model, the answer to both questions is yes: with GPT-5.6, the OFD agent identifies 63.4% of the hidden world models, against at most 44.6% for three recent visual-reasoning agents. Two mechanisms explain the gap. For Q1, constructed instruments make evidence compact and stable—23% fewer input tokens per identified law and fewer changes of formula family—whereas agents without them often report "measurements" that are outputs of their current candidate; the remaining error concentrates in coefficient calibration rather than functional form, the signature of comparison-based measurement. For Q2, in a matched one-revision study, three dispersed counterexamples raise strict recovery from 33% to 73% over a single one. These capabilities do not transfer for free: with Qwen3-VL-8B the same loop trails simpler scaffolds, because the model rarely completes it. Fine-tuning end to end on successful GPT-5.6 episodes transfers the loop to Qwen3-VL-32B, raising strict-or-relaxed recovery on seven unseen scenes from 42.9% to 57.1%, but not to Qwen3-VL-8B, which instead drifts toward textbook forms.

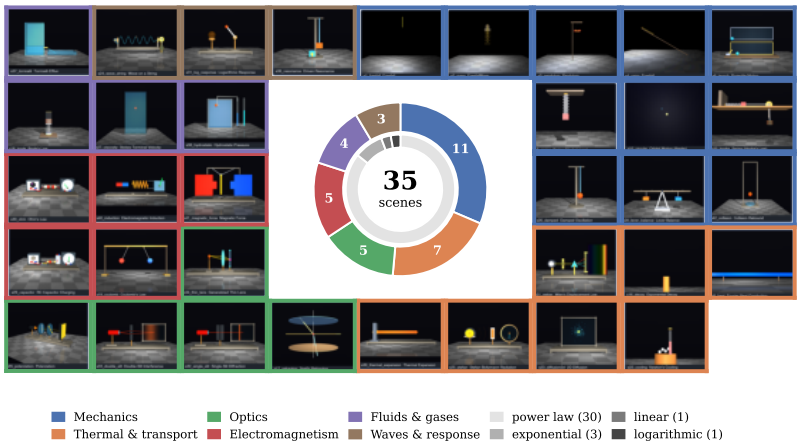


Figure 3: The 35 counterfactual physics scenes of Eyes of Galileo (raw renders, no instruments), arranged around their composition: physical domain (outer ring; thumbnail borders use the same colors) and law family of the hidden relation (inner ring). Each scene exposes one control and one rendered observable governed by a hidden exponent, rate, coefficient, or slope; Appendix A lists every control and observable.

In summary, this work makes the following contributions:

- We formulate active world-model identification: constructing identifiable evidence from pixels, proposing an executable symbolic law, and testing it through interventions.
- We propose a closed loop in which an agent builds its own visual references and falsifies its law through execution at dispersed targets, instantiated in Eyes of Galileo with 35 counterfactual scenes.
- We show that an agent that builds its own references and uses distributed counterexamples outperforms existing visual-reasoning scaffolds, with gains concentrated in recovering the form of a law.
- We show that the loop requires a capable backbone, and that imitation of successful trajectories teaches it to a mid-sized model (Qwen3-VL-32B) but not to a small one (Qwen3-VL-8B).

2 Related Work

Physical reasoning from fixed observations. Visual physics benchmarks test whether models predict outcomes or answer causal questions about pre-rendered scenes (Bakhtin et al., 2019; Yi et al., 2020; M. Bear et al., 2021; Chow et al., 2025; Xiang et al., 2025), including conservation laws and causal structure (Luo et al., 2026; Tang et al., 2026); CHAIN adds embodied interaction (Wu et al., 2026). World-model agents learn latent state-transition models for control and planning (Ha & Schmidhuber, 2018; Hafner et al., 2023; LeCun, 2022). In both lines the sensor interface is fixed by the benchmark, the physics is usually familiar, and success is prediction in a benchmark-defined representation. Eyes of Galileo instead hides counterfactual laws behind raw renders and asks for an executable symbolic model judged under unseen interventions, so the agent itself must decide what to measure.

Law discovery and scientific agents. Symbolic regression recovers compact laws from numeric tables (Schmidt & Lipson, 2009; Udrescu & Tegmark, 2020), LLM-based methods propose equations as programs (Shojaee et al., 2025a), and recent benchmarks evaluate LLM equation discovery from tabulated data or simulator queries (Zheng et al., 2025; Chen et al., 2025a; Shojaee et al., 2025b); DiscoverPhysics is closest in posing counterfactual worlds (Wiemann et al., 2026). ScienceAgentBench and DiscoveryBench evaluate analysis over collected datasets (Chen et al., 2025b; Majumder et al., 2025), and SIM2REASON uses a simulator DSL to train solvers (Prabhudesai et al., 2026);

end-to-end research agents automate hypothesis generation and experimentation in code (Lu et al., 2024a). These settings hand the agent numbers: measurement has already been done. Our agents receive only pixels, so the step that precedes fitting—making the phenomenon measurable—becomes part of the task, and feedback reports whether the law's consequences hold rather than supplying data to fit.

Visual tools and active perception. Set-of-Mark, ViP-LLaVA, and Visual Sketchpad add identifiers, points, boxes, or drawings to support grounding (Yang et al., 2023; Cai et al., 2024; Hu et al., 2024); thinking-with-images systems crop, draw on, or query images during inference (Su et al., 2025; Wang et al., 2026; You et al., 2026; Hong et al., 2025), and video agents search for the frames and regions relevant to a query (Gao et al., 2026); and general tool-using agents (Yao et al., 2023; Schick et al., 2023; Qin et al., 2024) and multimodal benchmarks (Yue et al., 2024; Lu et al., 2024b) operate on prepared inputs. These tools help a model look more closely at what it is given. Our instruments are calibrated spatial, temporal, and intensity references retained across interventions, so their marks keep stable semantics for comparison between conditions—the operation that turns an image into a measurement. Bayesian experimental design, active learning (Lindley, 1956; Chaloner & Verdinelli, 1995; Settles, 2009), and active inference (Friston, 2010) choose informative queries through a given measurement channel; here the agent must also build that channel.

3 Active World-Model Identification and OFD

We cast physical-law discovery as the active identification of a task-level world model. The agent plays three roles: it decides how to observe, turning raw renders into comparable evidence; it proposes a symbolic relation between a control and an outcome; and it tests that relation by intervening. Throughout, it never sees simulator state or numeric ground truth.

3.1 Task and Predictive Validity

Let $s_{1:T}^{(k)}$ denote the latent trajectory produced by intervention k, $I_{1:T}^{(k)}$ its rendered observation, u_k its controllable condition, and y_k the observable outcome. The agent seeks a symbolic model $m_\phi : u \mapsto y$ that predicts outcomes under interventions not used to construct it. This is a deliberately compact world model: it captures the target physical relation rather than reconstructing latent simulator state or generating future pixels.

Because neither s nor y is directly exposed, the agent must first construct evidence from pixels. We represent a visual evidence view as $z^{(k)} = R_{\tau,\theta}(I_{1:T}^{(k)}, I_{1:T}^{(0)}; m_\phi)$, where R is an observation transformation with instrument type τ and configuration θ, and $I^{(0)}$ an optional unperturbed reference. A view is useful when it makes the quantities that distinguish candidate models visually identifiable; its reference may be *natural* (stable scene structure) or *constructed* (a grid, strobe, timing gate, spectral band, or marker), changing the form of the evidence without changing the world. Conditioned on history h_t and current model, the agent selects $(\tau_t, \theta_t) \sim \pi_{\text{obs}}(h_t, m_{\phi_t})$ and updates m_{ϕ_t} after observing the rendered consequence; a successful loop must improve predictions beyond the current operating point.

Prediction, intervention, and evaluation. A physical intervention changes u and hence the latent trajectory; an observation action changes only the reference used to inspect it. Because m_ϕ is executable, its predictions can be compared directly with physical outcomes on a scene-specific grid of interventions (Section 4).

3.2 Observation-Feedback Discovery

OFD organizes identification as a five-stage loop in which the agent (i) selects or builds a visual reference, (ii) infers an executable model from the resulting evidence, (iii) predicts the outcome of an intervention, (iv) executes that prediction at one or more targets, and (v) revises the model from counterexamples. The first two stages concern *identifiability*—whether the pixels expose the comparisons that separate candidate laws. The last three concern *falsifiability*—whether a locally plausible law survives tests away from the operating point that suggested it.

Constructing identifiable evidence. A visual-instrument DSL provides composable strobe trails, background grids, light strips, spectral bands, and other calibrated references, which the agent can add, remove, or reconfigure before its next intervention. Instruments are observation actions: they change the evidence available for identifying m_ϕ but never the dynamics, much as a ruler placed beside a falling ball changes what can be read from a photograph but not how the ball falls.

Falsifying models through execution. A law is tested by acting on it. In **goal-conditioned execution (GCE)**, the environment inverts the submitted law to find the control u that should produce a desired outcome g, executes that control, and returns whether the goal was met together with the rendered result. **GCE-n** repeats this test at n targets; unless stated otherwise we use $n = 3$ targets at low, middle, and high outcomes. A correct law passes every such test, whereas a law fit only near one operating point fails away from it. Weaker feedback contracts used for reference are described in Appendix G.1.

Why dispersed targets, and why numbers. For a true law $y = Cu^\alpha$ and candidate $\hat{y} = \hat{C}u^{\hat{\alpha}}$, the log residual $r(u) = \delta_c + \delta_\alpha \log u$ is affine in $\log u$, and a target at control u_k passes iff $|r(u_k)| \le \varepsilon$.

Proposition 1. *(i) One passing target leaves δ_α unconstrained. (ii) Passing targets at $u_1 < u_2$ imply $|\delta_\alpha| \le 2\varepsilon/\log(u_2/u_1)$ and $|r| \le \varepsilon$ on $[u_1, u_2]$. (iii) Residual signs without magnitudes never bound $|\delta_\alpha|$.*

Dispersed targets therefore certify the exponent, whereas feedback that reveals only the direction of each miss supports rescaling a law but not reshaping it, a prediction borne out empirically (Appendix E.3). Proofs and the decomposition $\mathrm{JLR}^2 = \mathrm{Shape}^2 + \mathrm{Norm}^2$ are in Appendix H.

4 Eyes of Galileo Evaluation

Eyes of Galileo contains 35 counterfactual physics scenes across mechanics, thermal physics and transport, fluids and gases, waves and response, optics, and electromagnetism (Figure 3). In each scene a control drives an observable through a hidden exponent, rate, coefficient, or slope that may differ from its textbook value, so a remembered equation is not enough. Agents see only renders, never simulator state or numbers, and observation requests, apparatus proposals, and law submissions draw on a single interaction budget. Each scene is instantiated with five hidden settings, giving 175 scene–setting cells on which all end-to-end and instrument results are reported; the controlled feedback study uses a subset of 10 scenes. Protocol details and results on an earlier release of the environment appear in Appendices G.1, G.5, and F.

Metrics. The joint log residual (JLR) measures how well a submitted law predicts oracle outcomes on a scene-specific reference grid. Strict recovery requires $\mathrm{JLR} \le 0.20$; relaxed recovery accepts an effective exponent within 0.30 of the counterfactual value or closer to it than to the textbook default, and S/R is the union of the two. The agent never sees oracle values, only a binary law-fit signal derived from the same grid; scoring the final GPT-5.6 laws at unseen grid midpoints instead changes Strict and S/R by at most 1.1 points (Appendix F.1), so this signal has little influence on the reported scores. Missing or invalid submissions count as failures, and confidence intervals resample scenes to respect the dependence between settings of the same scene.

5 What Makes a Visual World Model Predictive?

We ask four questions: how the complete agent compares with existing visual-reasoning scaffolds and across backbones (Section 5.1), which property of execution feedback produces correct revisions (Section 5.2), what constructed observations contribute (Section 5.3), and whether a weaker backbone can learn the loop (Section 5.4).

5.1 End-to-End Predictive Recovery

Baselines. We compare with three recent visual-reasoning agents that gather or construct visual evidence through tools. **DeepEyesV2** (Hong et al., 2025) interleaves reasoning with code execution

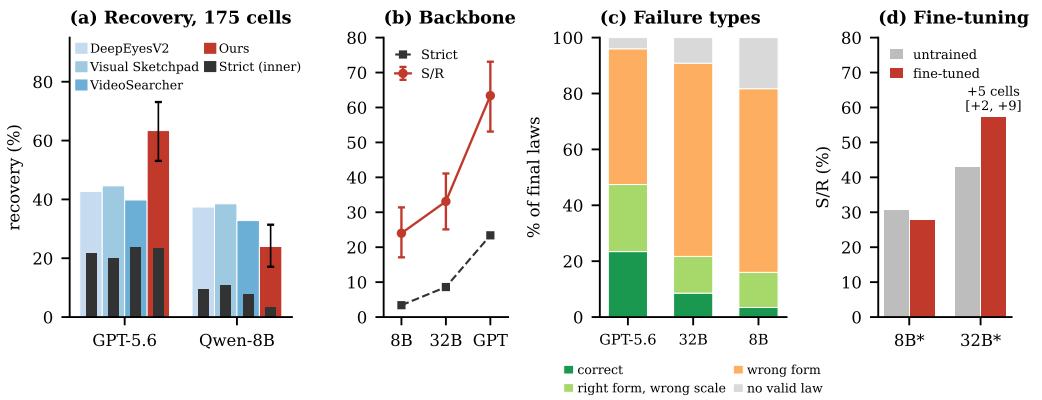


Figure 4: Main results. (a) Recovery on all 175 cells: bars are S/R, dark inner bars Strict; whiskers are 95% scene-cluster bootstrap intervals for our agent. (b) Our agent across backbones (Qwen3-VL-8B, Qwen3-VL-32B, GPT-5.6). (c) Failure types of final laws: correct (Strict), right form but miscalibrated scale (Shape ≤ 0.2), wrong form, or no valid law. (d) End-to-end fine-tuning on successful GPT-5.6 episodes, evaluated on 35 cells of seven held-out scenes (*); 8B bars are seed means.

over images, such as cropping, zooming, and numerical computation on pixels. **Visual Sketchpad** (Hu et al., 2024) lets the model draw auxiliary lines, boxes, and marks on an image and reason over the edited sketch. **VideoSearcher** (Gao et al., 2026) searches a video by temporal localization and spatial focusing, retrieving the frames and regions most relevant to a query. Each baseline replaces only our instrument tools with its own action space; the task interface, hidden laws, frozen backbone, interaction budget, GCE-n feedback, and scorer are identical, and no baseline is retrained with its original recipe.

Building evidence recovers more world models. With GPT-5.6, our agent identifies 63.4% of the hidden world models (S/R; 95% interval $[53.1, 73.1]$), whereas no baseline exceeds 44.6% (Table 1, Figure 4a). It also attains the lowest median JLR and matches the strongest baseline in exact recovery. Its advantage lies in the form of the law: shape error is on par with the best baseline, while the remaining error is mostly one of scale, with about as many final laws having the right form but a miscalibrated coefficient as being fully correct (Figure 4c). This is the error profile one expects from comparison-based measurement, which fixes ratios more reliably than absolute magnitudes. Memorized physics cannot account for the gain: the textbook exponent, even with an oracle-fitted coefficient, identifies only 14.3% of the laws and none of the settings far from the textbook value (Appendix F.2).

The loop requires a capable backbone. The advantage reverses with Qwen3-VL-8B, where our agent identifies 24.0% of the laws against 32.8–38.5% for the baselines. The small model struggles to carry the loop to completion: most of its episodes end with a law of the wrong form, and nearly a fifth without a valid law at all. Recovery grows steadily with capability, from 24.0% with Qwen3-VL-8B to 33.1% with Qwen3-VL-32B and 63.4% with GPT-5.6 (Figure 4b). GPT-4.1, evaluated on the power-law tasks of an earlier release of the environment, fits the same pattern: on the same tasks it approaches GPT-5.6 in S/R (56.4% versus 61.5%) but falls well behind in exact recovery (7.7% versus 23.1%; Appendix Table 15).

5.2 Distributed Falsification Improves Revision

Controlled revision study. To isolate how counterexamples should be chosen, we give every agent the same initial law and the same prompt and vary only the feedback that precedes a single revision. GCE-1 reports one fixed target and GCE-3 three dispersed ones; Adaptive A reports the first failing target found by sequential probing, and Adaptive B, an oracle, the three largest errors among five

Backbone	Method	Strict (%)↑	S/R (%)↑	Median JLR↓	Mean JLR↓	Shape err.↓	Norm. err.↓
GPT-5.6	DeepEyesV2	21.5	42.7	1.764	2.136 ± 1.964	0.947 ± 0.882	$\mathbf{1.362 \pm 1.582}$
	Visual Sketchpad	19.8	44.6	1.593	2.031 ± 1.483	$\mathbf{0.857 \pm 0.752}$	1.489 ± 0.974
	VideoSearcher	**23.8**	39.8	1.359	$\mathbf{1.974 \pm 1.552}$	0.948 ± 0.578	1.431 ± 0.932
	Ours	23.4	**63.4**	**1.205**	2.393 ± 5.011	0.862 ± 2.602	2.025 ± 4.384
Qwen3-VL-8B	DeepEyesV2	9.4	37.4	2.074	2.538 ± 1.692	0.994 ± 0.682	$\mathbf{1.592 \pm 1.329}$
	Visual Sketchpad	**10.9**	**38.5**	1.963	2.842 ± 1.385	0.952 ± 0.576	1.680 ± 1.042
	VideoSearcher	7.7	32.8	**1.956**	$\mathbf{2.484 \pm 1.684}$	$\mathbf{0.864 \pm 0.482}$	1.745 ± 0.958
	Ours	3.4	24.0	2.101	3.736 ± 6.525	1.144 ± 1.621	3.283 ± 6.468
Qwen3-VL-32B	Ours	8.6	33.1	1.431	2.649 ± 6.108	0.886 ± 1.378	2.309 ± 6.026

Table 1: End-to-end comparison on the same 175 scene–setting cells (35 scenes, five hidden settings) with GPT-5.6, Qwen3-VL-8B-Instruct, and Qwen3-VL-32B-Instruct backbones (baselines were not evaluated with Qwen3-VL-32B). All methods receive identical GCE-n execution feedback and interaction budgets. Strict and S/R are exact and strict-or-relaxed recovery; mean JLR and error columns report mean $\pm$ SD over episodes. For each episode, shape error is the standard deviation and normalization error the absolute mean of the signed log residuals over the evaluator grid, so that $\mathrm{JLR}^2 = \mathrm{shape}^2 + \mathrm{norm}^2$; Bold marks the best value in each column within a backbone.

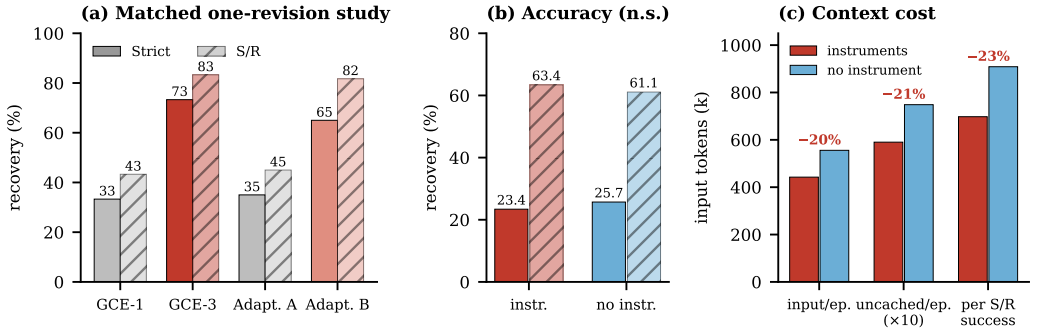


Figure 5: Controlled studies. (a) Controlled revision study (Section 5.2): all feedback conditions start from the same initial law; solid bars show Strict and hatched bars S/R recovery. (b, c) Agents with and without instruments (Section 5.3) reach similar accuracy, but instruments reduce the input tokens needed per episode and per identified law. Full tables are in Appendix C.

probes. Feedback is given as numbers rather than renders, so that the comparison reflects how the agent revises rather than how it perceives.

Coverage, not adaptivity, drives correct revision. Three dispersed counterexamples raise exact recovery from 33.3% to 73.3% over a single one (Figure 5a), and the improvement is nearly one-sided: GCE-3 succeeds where GCE-1 fails in 41.7% of matched cases, the reverse in only 1.7%. Adaptive B likewise outperforms Adaptive A, which shows fewer targets and performs no better than GCE-1. Choosing targets adaptively does not add to coverage: even the oracle Adaptive B, which consults five hidden probes, recovers fewer laws than fixed low, middle, and high targets (65.0% versus 73.3%), plausibly because the largest errors cluster in one region or on a wrong inverse branch. This is the pattern predicted by Proposition 1, and we use fixed GCE-3 throughout.

Because more targets also means more feedback, this study does not fully separate coverage from volume (Appendix F). The benefit also depends on the reviser: with Qwen3-VL-8B or 32B, three counterexamples do not improve exact recovery (Appendix Table 16), in line with the backbone dependence of Section 5.1.

5.3 Constructed Observations Make Evidence Compact

To measure what constructed observations contribute, we compare the complete agent with an otherwise identical agent that has no instrument tools and must work from raw renders.

Instruments make evidence compact. The two agents reach statistically indistinguishable accuracy (Figure 5b): each leads by 2.3 points on one recovery metric, and neither difference is significant (paired tests $p = 0.63$ for both metrics and $p = 0.17$ for JLR). Individual revisions are equally productive, lowering JLR about 55% of the time in both conditions.

Instruments change instead what the evidence costs (Figure 5c). Although the instrumented agent receives more images, they are smaller and more informative: without instruments, an episode consumes 25.6% more input tokens, and each identified law costs 23.2% fewer tokens with them. Constructed references thus make evidence more compact and, as Section 6 shows, more stable, although the current agent does not yet convert this into higher accuracy.

5.4 Learning to Use the Loop

Can a weaker backbone acquire the loop by imitation? We fine-tune Qwen3-VL-8B (LoRA) and Qwen3-VL-32B (4-bit QLoRA) on successful GPT-5.6 episodes, supervising every agent action, and evaluate each student on seven scenes withheld from training, running the complete loop against its untrained counterpart (Appendix E.2).

Imitation transfers the loop only to a sufficiently capable backbone (Table 2, Figure 4d). At 8B, fine-tuning leaves recovery unchanged (27.6% versus 30.5% S/R, averaged over seeds; the 95% interval of the difference spans zero), and the student gravitates toward textbook forms, for instance proposing a square-root pendulum law in a scene whose hidden exponent is 1. At 32B, fine-tuning helps substantially: exact recovery triples from 8.6% to 25.7% and S/R rises from 42.9% to 57.1%, with both gains supported by scene-level intervals that exclude zero. Imitation thus strengthens a loop that the backbone can already partly run; below that threshold, it reproduces the surface of the trajectories rather than their use of evidence.

6 How World Models Change During Interaction

World models rarely improve monotonically during an episode; only about 15% of episodes lower their error at every revision. Feedback can correct a model but can also displace a useful one: without instruments, structural revisions turn a failing law into a passing one less than twice as often as they do the reverse. Revisions are also seldom informed by new observations, since more than 95% of consecutive submissions have none between them. They are not blind guesses either, lowering JLR in 56.5% of cases, but they rarely follow the scientific pattern of diagnosing a failure, gathering a targeted measurement, and then revising. Four cases illustrate these dynamics.

Pendulum: form from evidence, scale from counterexamples. In the episode of Figure 1, GPT-5.6 installs a strobe trail and then a filmstrip, and compares the period at three string lengths. Observing that a fourfold longer string doubles the period, it proposes $T \propto L^{0.5}$ with coefficient 2.54.

Backbone	State	Strict	S/R	Valid law
Qwen3-VL-8B	Untrained	3.8%	30.5%	88.6%
	Fine-tuned	4.8%	27.6%	79.0%
Qwen3-VL-32B	Untrained	8.6%	42.9%	88.6%
	Fine-tuned	**25.7%**	**57.1%**	97.1%

Table 2: End-to-end fine-tuning, evaluated on 35 cells from seven unseen scenes. 8B entries average over sampling seeds.

All three GCE-n targets fail; without returning to the scene, the agent keeps the exponent and rescales the coefficient to 2.0, which passes every target (Appendix D). Constructed evidence supplies the form, and dispersed counterexamples supply the calibration.

Capacitor: readings that follow the candidate. Working from the same six raw renders and making no new observation, an agent without instruments reports a different data table for each candidate law it considers: $(1, 3.7)$ for `10*exp(-time)`, $(1, 10)$ for `10/time`, $(3, 5.8)$ for `10/sqrt(time)`, and $(3, 7.0)$ for `10-time`. Each table agrees with the candidate it accompanies: the evidence follows the hypothesis rather than constraining it. The instrumented agent in the same cell records the gauge geometry and marks the reading as unresolved; it also fails, but its evidence stays the same whatever it hypothesizes.

Thin lens and Torricelli: references stabilize readings. Without an instrument, a thin-lens agent assigns the same frames values ranging from 3.2×10^5 to 10^{20} and pursues exponential families; with one, it retains consistent pixel readings (object centers near 522, 468, 426, 340, and 212 px) and proposes the rational thin-lens family. In Torricelli, the no-instrument agent reads the same jet frames as sublinear, linear, superlinear, and saturating in turn, whereas the instrumented agent reuses the same endpoint readings (about 48, 95, 188, 282, and 376 px) in every revision. Neither agent reaches the exact law, but the reference holds the evidence fixed and steers the search toward the right family.

Viscosity: when more counterexamples displace a useful law. In the controlled revision study, Qwen3-VL-32B retains a nearly correct law `1.21*radius**2` (JLR 0.19) under one target but replaces it with `1.21*radius**3` (JLR 1.12) when shown three; similar reversals occur in the launch and collision scenes (Appendix F).

Across these cases, references stabilize evidence and steer the hypothesis family, and dispersed counterexamples calibrate a law whose form is already right; what agents rarely do is return to the scene for the measurement that would discriminate between competing forms.

7 Discussion and Conclusion

Identifying world models requires making evidence. A world model meant to hold under new interventions is best built on evidence that the agent has made measurable itself. Constructed references make that evidence compact and stable: agents that use them keep the same readings across revisions, whereas agents without them tend to re-read the frames in favor of each new candidate (Section 6). Our loop turns such evidence into a model, and a capable agent identifies most hidden laws this way. What it rarely does yet is return to the scene for new, targeted evidence before revising.

Dispersed falsification and capable backbones. Counterexamples at dispersed targets certify the form of a law (Proposition 1), and a capable model makes use of them: three dispersed targets more than double exact recovery relative to one (Section 5.2). The loop grows stronger with the backbone, from 24.0% with Qwen3-VL-8B to 63.4% with GPT-5.6, and it can be learned from successful trajectories by a mid-sized model but not a small one. Choosing a discriminating measurement after a failure is the natural next capability to learn, and Eyes of Galileo measures it directly.

Conclusion. A world model can be trusted only after the world has been made measurable and the model exposed to interventions it has not seen. From the raw pixels of 35 counterfactual scenes, an agent that builds its own references and tests its law at dispersed targets identifies 63.4% of the hidden world models, well above existing visual-reasoning agents. Learning where to look next after a failed prediction is the central open problem this work leaves.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript language editing, LaTeX formatting, and software-code editing. The authors reviewed the AI-assisted work and take responsibility for the final content of this paper, including its text, claims, analyses, and artifacts.

ETHICS STATEMENT

This work studies agents in simulated, counterfactual physics scenes. It involves no human subjects, personal data, or real-world actuation. The benchmark deliberately departs from textbook laws, so recovered laws should not be read as claims about the physical world.

REPRODUCIBILITY STATEMENT

Appendix A specifies every scene, control, observable, law family, and hidden setting (Table 4); Appendix A gives the evaluation protocol, budgets, and manifest hash; Appendix A shows the exact agent-visible feedback; and Appendix E documents data filtering, training settings, and sealed controls for the distillation studies. We will release the simulator, instrument DSL, episode runner, hidden-law manifest, scoring scripts, and trained adapters.

REFERENCES

Anton Bakhtin, Laurens van der Maaten, Justin Johnson, Laura Gustafson, and Ross Girshick. Phyre: A new benchmark for physical reasoning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Mu Cai, Haotian Liu, Siva Karthik Mustikovela, Gregory P. Meyer, Yuning Chai, Dennis Park, and Yong Jae Lee. ViP-LLaVA: Making large multimodal models understand arbitrary visual prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12914–12923, 2024.

Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical Science*, 10(3):273–304, 1995.

Yimeng Chen, Piotr Piękos, Mateusz Ostaszewski, Firas Laakom, and Jürgen Schmidhuber. Physgym: Benchmarking llms in interactive physics discovery with controlled priors. *arXiv preprint arXiv:2507.15550*, 2025a.

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*, 2025b.

Wei Chow, Jiageng Mao, Boyi Li, Daniel Seita, Vitor Campagnolo Guizilini, and Yue Wang. Physbench: Benchmarking and enhancing vision-language models for physical world understanding. In *International Conference on Learning Representations*, 2025.

Karl Friston. The free-energy principle: A unified brain theory? *Nature Reviews Neuroscience*, 11 (2):127–138, 2010. doi: 10.1038/nrn2787.

Zhenkun Gao, Yicheng Bao, Jinlong Peng, et al. VideoSearcher: Empowering video deep research with multi-tool agentic reasoning via reinforcement learning. *arXiv preprint arXiv:2607.02927*, 2026.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Jack Hong et al. DeepEyesV2: Toward agentic multimodal model. *arXiv preprint arXiv:2511.05271*, 2025.

Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Ranjay Krishna. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *Advances in Neural Information Processing Systems*, 2024.

Yann LeCun. A path towards autonomous machine intelligence. *OpenReview*, 2022. Version 0.9.2.

Dennis V. Lindley. On a measure of the information provided by an experiment. *The Annals of Mathematical Statistics*, 27(4):986–1005, 1956.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI scientist: Towards fully automated open-ended scientific discovery. *arXiv preprint arXiv:2408.06292*, 2024a.

Pan Lu, Hritik Bansal, Tony Xia, Jiacheng Liu, Chunyuan Li, Hannaneh Hajishirzi, Hao Cheng, Kai-Wei Chang, Michel Galley, Jianfeng Gao, et al. Mathvista: Evaluating mathematical reasoning of foundation models in visual contexts. In *International Conference on Learning Representations*, 2024b.

Dezhi Luo, Yijiang Li, Maijunxian Wang, Tianwei Zhao, Bingyang Wang, Siheng Wang, Pinyuan Feng, Pooyan Rahmanzadehgervi, Ziqiao Ma, and Hokin Deng. Vision language models cannot reason about physical transformation. *arXiv preprint arXiv:2603.07109*, 2026.

Daniel M. Bear, Elias Wang, Damian Mrowca, Felix Binder, et al. Physion: Evaluating physical prediction from vision in humans and machines. In *Advances in Neural Information Processing Systems*, volume 34, pp. 22786–22798, 2021.

Bodhisattwa Prasad Majumder, Harshit Surana, Dhruv Agarwal, Bhavana Dalvi Mishra, Abhijeetsingh Meena, Aryan Prakhar, Tirth Vora, Tushar Khot, Ashish Sabharwal, and Peter Clark. Discoverybench: Towards data-driven discovery with large language models. In *International Conference on Learning Representations*, 2025.

Mihir Prabhudesai, Aryan Satpathy, Yangmin Li, Zheyang Qin, Nikash Bhardwaj, Amir Zadeh, Chuan Li, Katerina Fragkiadaki, and Deepak Pathak. Solving physics olympiad via reinforcement learning on physics simulators. *arXiv preprint arXiv:2604.11805*, 2026.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations*, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*, 2023.

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chandan K. Reddy. LLM-SR: Scientific equation discovery via programming with large language models. In *International Conference on Learning Representations*, 2025a.

Parshin Shojaee, Ngoc Hieu Nguyen, Kazem Meidani, Amir Barati Farimani, Khoa Doan, and Chandan K. Reddy. Llm-srbench: A new benchmark for scientific equation discovery with large language models. In *International Conference on Machine Learning*, 2025b.

Zhaochen Su, Peng Xia, Hangyu Guo, Zhenhua Liu, Yan Ma, Xiaoye Qu, Jiaqi Liu, Yanshu Li, Kaide Zeng, Zhengyuan Yang, Linjie Li, Yu Cheng, Heng Ji, Junxian He, and Yi R. Fung. Thinking with images for multimodal reasoning: Foundations, methods, and future frontiers. *arXiv preprint arXiv:2506.23918*, 2025.

Tianyi Tang, Zhuoyi Lin, Zeyu Feng, Tianyi Ma, Yew-Soon Ong, Ivor Tsang, and Haiyan Yin. Causal scaffolding for physical reasoning: A benchmark for causally-informed physical world understanding in vlms. *arXiv preprint arXiv:2606.05966*, 2026.

Silviu-Marian Udrescu and Max Tegmark. Ai feynman: A physics-inspired method for symbolic regression. *Science Advances*, 6(16):eaay2631, 2020.

Chaoyang Wang, Wenrui Bao, Sicheng Gao, Bingxin Xu, Yu Tian, Yogesh S. Rawat, Yunhao Ge, and Yuzhang Shang. Vla-thinker: Boosting vision-language-action models through thinking-with-image reasoning. *arXiv preprint arXiv:2603.14523*, 2026.

Matt L. Wiemann, Lindsay M. Smith, Peter Melchior, Siddharth Mishra-Sharma, Andrew Gordon Wilson, Pavel Izmailov, and Carolina Cuesta-Lázaro. Discoverphysics: Benchmarking llms for out-of-the-box scientific thinking. *arXiv preprint arXiv:2605.26087*, 2026.

Yuhao Wu, Maojia Song, Yihuai Lan, Lei Wang, Zhiqiang Hu, Yao Xiao, Heng Zhou, Weihua Zheng, Dylan Raharja, Soujanya Poria, and Roy Ka-Wei Lee. From perception to action: An interactive benchmark for vision reasoning. *arXiv preprint arXiv:2602.21015*, 2026.

Kun Xiang, Heng Li, Terry Jingchen Zhang, Yinya Huang, Zirong Liu, Peixin Qu, Jixi He, Jiaqi Chen, Yu-Jie Yuan, Jianhua Han, Hang Xu, Hanhui Li, Mrinmaya Sachan, and Xiaodan Liang. Seephys: Does seeing help thinking?—benchmarking vision-based physics reasoning. *arXiv preprint arXiv:2505.19099*, 2025.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chunyuan Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in GPT-4V. *arXiv preprint arXiv:2310.11441*, 2023.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Antonio Torralba, and Joshua B Tenenbaum. Clevrer: Collision events for video representation and reasoning. In *International Conference on Learning Representations*, 2020.

Weihang You, Qingchan Zhu, David Liu, Yi Pan, Geng Yuan, and Hanqi Jiang. Vithinker: Active vision-language reasoning via dynamic perceptual querying. *arXiv preprint arXiv:2602.02873*, 2026.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, et al. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Tianshi Zheng, Kelvin Kiu-Wai Tam, Newt Hue-Nam K. Nguyen, Baixuan Xu, Zhaowei Wang, Jiayang Cheng, Hong Ting Tsang, Weiqi Wang, Jiaxin Bai, Tianqing Fang, Yangqiu Song, Ginny Y. Wong, and Simon See. Newtonbench: Benchmarking generalizable scientific law discovery in llm agents. *arXiv preprint arXiv:2510.07172*, 2025.

A Additional Benchmark Details

A1. Full 35-scene setup

All 35 scenes use one unified MuJoCo visual stack. Each scene exposes one control and one rendered observable whose dependence on the control is governed by a hidden parameter of the listed law family.

Table 3: Complete 35-scene benchmark specification, grouped by physical domain.

Scene	Control	Observable	Law family	Hidden parameter
Mechanics (11)				
s1_freefall	max_t	Distance fallen (m)	power_law	alpha
s2_mass	mass_kg	Distance fallen (m)	power_law	alpha
s3_pendulum	length_m	Period (s)	power_law	alpha
s4_ramp	max_t	Distance fallen (m)	power_law	alpha
s6_launch	speed_mps	Horizontal range (m)	power_law	alpha
s9_spring	mass_kg	Spring extension (m)	power_law	alpha
s10_circular	semi_major_axis	Orbital period (years)	power_law	alpha
s20_hooke	mass_kg	Spring extension (m)	power_law	alpha
s26_damped	max_t	Amplitude envelope	power_law	alpha
s28_lever_balance	arm_distance	Balancing force	power_law	alpha
s42_collision	drop_height	Rebound height	power_law	alpha
Thermal physics and transport (7)				
s8_heat	time_s	Heat front position (m)	power_law	alpha
s16_decay	time	Remaining brightness	exponential	rate
s21_weber	temperature	Peak wavelength (nm)	power_law	alpha
s22_cooling	time	Temperature above ambient (°C)	exponential	rate
s23_diffusion2d	time	RMS spread radius	power_law	alpha
s25_stefan	temperature	Radiated power (arb. units)	power_law	alpha
s39_thermal_expansion	delta_T	Rod extension	power_law	alpha
Fluids and gases (4)				
s18_boyle	pressure_atm	Gas column length (cm)	power_law	alpha
s27_viscosity	radius	Terminal velocity (mm/s)	power_law	alpha
s37_torricelli	water_head	Jet range	power_law	alpha
s38_hydrostatic	probe_depth	Manometer deflection	power_law	alpha
Waves and response (3)				
s24_wave_string	tension	Wave speed (m/s)	power_law	alpha
s31_log_response	stimulus	Response magnitude	logarithmic	coeff
s36_resonance	drive_frequency	Steady-state amplitude	power_law	alpha
Optics (5)				
s17_refraction	angle_incidence_deg	Angle of refraction (degrees)	power_law	alpha
s32_single_slit	slit_width	Central-lobe width	power_law	alpha
s33_double_slit	slit_separation	Fringe spacing	power_law	alpha
s34_thin_lens	object_distance	Image distance	power_law	alpha
s35_polarization	angle_deg	Transmitted intensity	power_law	alpha
Electromagnetism (5)				
s19_coulomb	distance	Force magnitude (N)	power_law	alpha
s29_capacitor	time	Voltage deficit (V)	exponential	rate
s30_ohm	voltage	Current (A)	linear	slope
s40_induction	speed	Pointer deflection	power_law	alpha
s41_magnetic_force	current	Wire offset	power_law	alpha

A2. Textbook references and hidden settings

Each of the 35 scenes is evaluated at five hidden settings of its law parameter: a central value p_0, two near settings $p_0 \pm 0.5$, and two far settings $p_0 \pm 2.0$, giving the 175 scene–setting cells used throughout the main text. Settings need not coincide with the textbook value, so recalling a familiar law is not sufficient, and the far settings deliberately leave the familiar regime. Table 4 lists the textbook reference, the central value, and all five settings; the settings are never shown to the agent. Strict recovery applies to every scene; the relaxed exponent criterion (S/R) is defined only for power-law scenes, so exponential, linear, and logarithmic scenes count as S/R successes only through Strict recovery.

Table 4: Textbook references and the five hidden settings per scene (p_0, $p_0 \pm 0.5$, $p_0 \pm 2.0$). For Boyle compression the listed parameter is transformed to the observable exponent before relaxed scoring.

Scene	Law family	Textbook	Center	Hidden settings	Criteria
s1_freefall	power law	$\alpha = 2$	$\alpha_0 = 3$	1, 2.5, 3, 3.5, 5	S/R
s2_mass	power law	$\alpha = 0$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R
s3_pendulum	power law	$\alpha = 0.5$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R
s4_ramp	power law	$\alpha = 2$	$\alpha_0 = 2.5$	0.5, 2, 2.5, 3, 4.5	S/R
s6_launch	power law	$\alpha = 2$	$\alpha_0 = 3$	1, 2.5, 3, 3.5, 5	S/R
s8_heat	power law	$\alpha = 0.5$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R
s9_spring	power law	$\alpha = 1$	$\alpha_0 = 2$	0, 1.5, 2, 2.5, 4	S/R
s10_circular	power law	$\alpha = 3$	$\alpha_0 = 3$	1, 2.5, 3, 3.5, 5	S/R
s16_decay	exponential	–	$r_0 = -0.5$	-2.5, -1, -0.5, 0, 1.5	Strict
s17_refraction	power law	$\alpha = 1$	$\alpha_0 = 1.5$	-0.5, 1, 1.5, 2, 3.5	S/R
s18_boyle	power law	$\alpha = 1$	$\alpha_0 = 2$	0, 1.5, 2, 2.5, 4	S/R
s19_coulomb	power law	$\alpha = -2$	$\alpha_0 = -2.5$	-4.5, -3, -2.5, -2, -0.5	S/R
s20_hooke	power law	$\alpha = 1$	$\alpha_0 = 2$	0, 1.5, 2, 2.5, 4	S/R
s21_weber	power law	$\alpha = -1$	$\alpha_0 = -2$	-4, -2.5, -2, -1.5, 0	S/R
s22_cooling	exponential	–	$r_0 = -0.3$	-2.3, -0.8, -0.3, 0.2, 1.7	Strict
s23_diffusion2d	power law	$\alpha = 0.5$	$\alpha_0 = 0.5$	-1.5, 0, 0.5, 1, 2.5	S/R
s24_wave_string	power law	$\alpha = 0.5$	$\alpha_0 = 0.5$	-1.5, 0, 0.5, 1, 2.5	S/R
s25_stefan	power law	$\alpha = 4$	$\alpha_0 = 4$	2, 3.5, 4, 4.5, 6	S/R
s26_damped	power law	$\alpha = 1$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R
s27_viscosity	power law	$\alpha = 2$	$\alpha_0 = 2$	0, 1.5, 2, 2.5, 4	S/R
s28_lever_balance	power law	$\alpha = -1$	$\alpha_0 = -1$	-3, -1.5, -1, -0.5, 1	S/R
s29_capacitor	exponential	–	$r_0 = -2$	-4, -2.5, -2, -1.5, 0	Strict
s30_ohm	linear	–	$k_0 = 4$	2, 3.5, 4, 4.5, 6	Strict
s31_log_response	logarithmic	–	$c_0 = 5$	3, 4.5, 5, 5.5, 7	Strict
s32_single_slit	power law	$\alpha = -1$	$\alpha_0 = -1$	-3, -1.5, -1, -0.5, 1	S/R
s33_double_slit	power law	$\alpha = -1$	$\alpha_0 = -1$	-3, -1.5, -1, -0.5, 1	S/R
s34_thin_lens	power law	$\alpha = 1$	$\alpha_0 = 1.5$	-0.5, 1, 1.5, 2, 3.5	S/R
s35_polarization	power law	$\alpha = 2$	$\alpha_0 = 2$	0, 1.5, 2, 2.5, 4	S/R
s36_resonance	power law	$\alpha = 2$	$\alpha_0 = 2$	0, 1.5, 2, 2.5, 4	S/R
s37_torricelli	power law	$\alpha = 0.5$	$\alpha_0 = 0.5$	-1.5, 0, 0.5, 1, 2.5	S/R
s38_hydrostatic	power law	$\alpha = 1$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R
s39_thermal_expansion	power law	$\alpha = 1$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R
s40_induction	power law	$\alpha = 1$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R
s41_magnetic_force	power law	$\alpha = 1$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R
s42_collision	power law	$\alpha = 1$	$\alpha_0 = 1$	-1, 0.5, 1, 1.5, 3	S/R

A3. Evaluation protocol of the 35-scene studies

Table 5 groups the 35 scenes by physical family. All main-text end-to-end studies use the same 175 scene–setting cells (Table 4) and a frozen hidden-law manifest (SHA-256 prefix `3ec0b44c`). The GPT-5.6 agent runs with medium reasoning effort under the `luna-evaluation-v1` protocol: at most 10 actions and five apparatus proposals per episode, scene seed 0, and GCE-n with $n = 3$ targets, with and without apparatus proposals (Section 5.3). The Qwen3-VL end-to-end rows reuse the same episode specifications through the unified runner and an OpenAI-compatible vLLM server (temperature 0.2, seed 0, 4,096-token output cap, one initial keyframe). External baselines are evaluated on the same cells and budget. Missing or invalid final submissions count as failures, and our rows are scored from the last submitted law with the decomposition described in Table 1.

Family	Count	Tasks
Mechanics	11	free fall; mass-dependent fall; inclined ramp; projectile range; orbital period; collision and rebound; lever balance; vertical spring; horizontal spring; pendulum; damped oscillation
Thermal/transport	7	heat conduction; Newton cooling; two-dimensional diffusion; thermal expansion; exponential decay; Stefan–Boltzmann radiation; Wien displacement
Fluids/gases	4	Boyle compression; hydrostatic pressure; Torricelli outflow; viscous terminal motion
Waves/response	3	driven string wave; logarithmic stimulus response; driven resonance
Optics	5	refraction; thin-lens imaging; single-slit diffraction; double-slit interference; polarization
Electromagnetism	5	Coulomb interaction; capacitor charging; Ohm response; electromagnetic induction; magnetic force

Table 5: The 35-scene suite by physical family. The grouping fixes presentation order but does not change the scene-specific physics or visual evidence.

A4. Agent-visible feedback

For GCE and GCE-n, an agent-facing tool result has the following schematic content; the image items are the execution animation and residual render:

```
[
  {"type":"text",
   "text":"ORACLE LAW FIT: FAIL ..."},
  {"type":"text",
   "text":"TARGET 1/3 PHYSICAL EXECUTION: FAIL ..."},
  {"type":"image_url", "image_url":"<execution render>"}
]
```

The saved trace separately records audit fields such as `ok`, `physical_ok`, and `target_results`. The `score.rows` array contains numeric ground truth, predictions, and per-control errors; it is written only after interaction and is not included in the tool payload. Thus "oracle hidden" means that numeric rows remain hidden, not that all oracle-derived signals are hidden.

A5. Visual-instrument DSL

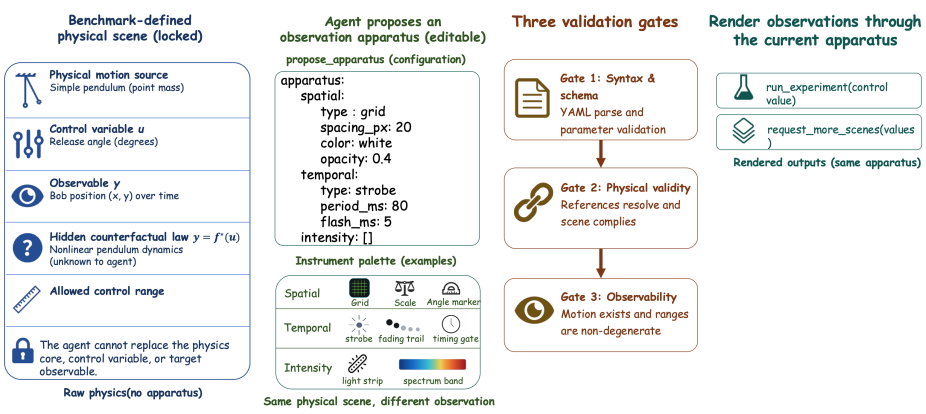


Figure 6: Detailed visual-instrument DSL and rendering pipeline. Each scenario locks the physical source, control variable, target observable, hidden law, and control range. The agent edits only the observation layer. Proposals must pass syntax/schema, physical-validity, and observability gates before the runner renders subsequent experiments through the accepted apparatus.

A6. Goal-conditioned execution details

Figure 7 expands the feedback-policy component of the main overview. The agent submits a symbolic law, while the runner computes the control required for each target and executes it in the simulator. Physical execution feedback and a binary oracle-law-fit result are visible to the agent. Numeric score rows and ground-truth outcomes on the evaluator grid remain offline.

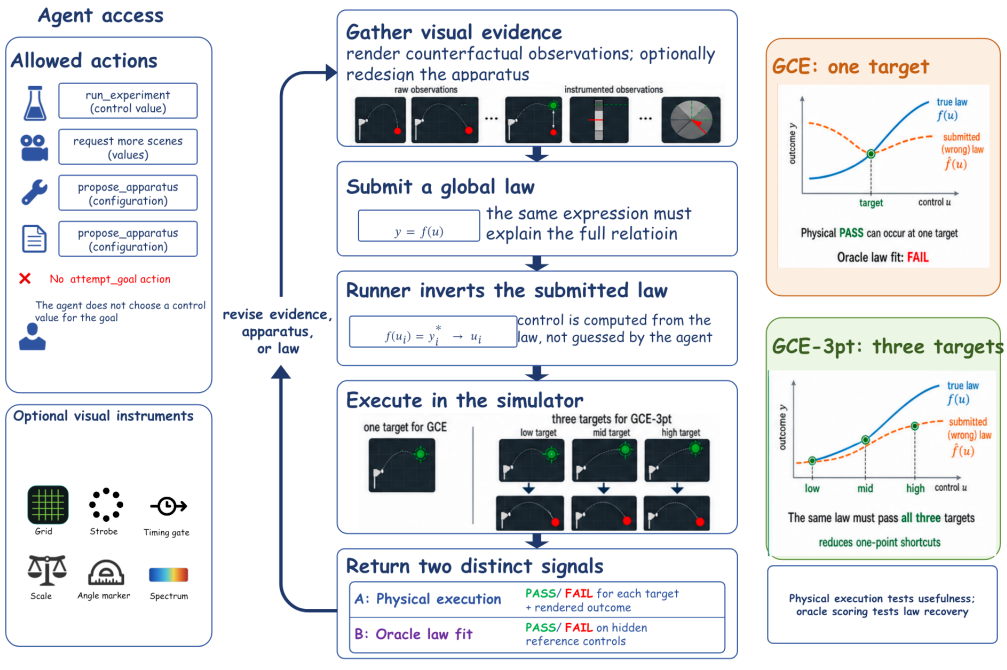


Figure 7: Goal-conditioned execution in the unified runner. The agent gathers visual evidence and submits a global law; it does not manually choose a control value for the goal. The runner inverts the submitted law, executes the resulting control at one target (GCE) or n targets (GCE-n), and returns rendered outcomes with physical-execution and oracle-law-fit pass/fail signals. We use $n = 3$ low/mid/high targets. A law may succeed at one target while remaining globally incorrect, whereas GCE-n requires the same law to succeed across all evaluated targets.

B Scene and Instrument Rendering Examples

B1. Physical and visual coverage of the 35 scenes

The suite covers six physical domains (Table 3): mechanics (11 scenes), thermal physics and transport (7), fluids and gases (4), waves and response (3), optics (5), and electromagnetism (5). Hidden laws include power laws (29 scenes), exponential rates (3), a linear slope (Ohm's law), and a logarithmic coefficient, so the benchmark is not tied to one law family. The observables are read through different visual channels—positions and displacements (freefall, launch, jet range), scalar gauges and bars (heat, cooling, capacitor voltage, current), angles and phases (refraction, pendulum, orbital period), and spatial patterns (diffraction lobes, interference fringes, diffusion spread)—because model failures differ by visual readout type, not only by equation family.

B2. RENDERING GALLERY OF ALL 35 SCENES



Figure 8: Raw rendered snapshots of all 35 scenes (seed 0, no instruments).

B3. Examples of instrument-augmented evidence

Visual instruments organize scene variation around explicit references. Grids, angle markers, and objects of known size provide spatial calibration; strobe trails expose temporal progression; and light strips or spectrum bands make intensity changes comparable across conditions. Figure 9 shows representative outputs rendered from valid DSL configurations.

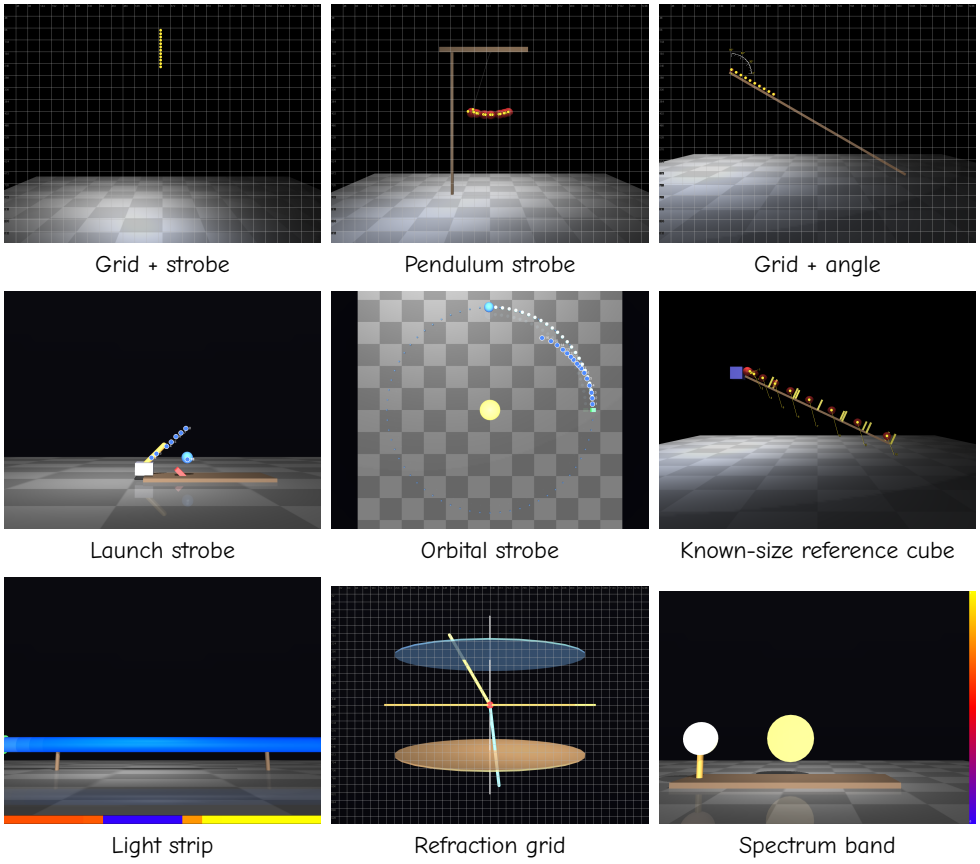


Figure 9: Instrument-augmented observations produced by the DSL. The gallery illustrates spatial, temporal, angular, and intensity references, including a known-size cube that supplies an object-level scale within the scene.

C Detailed Tables for the Controlled Studies

Tables 6 and 7 give the full numbers behind Figure 5.

Feedback arm	Strict	S/R	Mean JLR	Median JLR	Visible targets	Hidden probes
Fixed GCE-1	33.3%	43.3%	4.8268	1.2243	1.00	1.00
Fixed GCE-3	**73.3%**	**83.3%**	4.5453	**0.0000**	3.00	3.00
Adaptive A (first fail)	35.0%	45.0%	4.8753	1.2353	**0.75**	2.10
Adaptive B (max-error top 3)	65.0%	81.7%	**4.3979**	**0.0000**	3.00	5.00

Table 6: Matched one-revision feedback study. Within each matched stratum, all four arms start from the same historical GCE-1 first submission. Visible targets are counterexamples shown to the revising model; hidden probes include oracle evaluations used for selection but not necessarily shown. One invalid GCE-1 expression is counted as a Strict/S/R failure and excluded from finite JLR summaries.

Metric	Instrumented GCE-3	No instrument
Final model quality and completion		
Strict recovery	23.4%	25.7%
S/R recovery	63.4%	61.1%
Mean JLR	2.393	2.517
Completed run with a valid submission	99.4%	100.0%
Interaction and context cost, mean per episode		
Accepted tool calls	8.00	6.46
Law submissions	4.21	4.42
Returned image items	22.62	15.94
Model input tokens	**442,509**	555,676
Uncached input tokens	**59,019**	74,847
Input tokens per final S/R success	**697,650**	908,818
Revision trajectory		
Transitions without an intervening new observation	95.7%	96.7%
Transitions that change the formula skeleton	47.4%	53.8%
Episodes with any formula-skeleton change	64.9%	77.7%
Distinct formula skeletons per episode	2.24	2.43
Episodes with monotonically non-increasing JLR	15.8%	15.4%

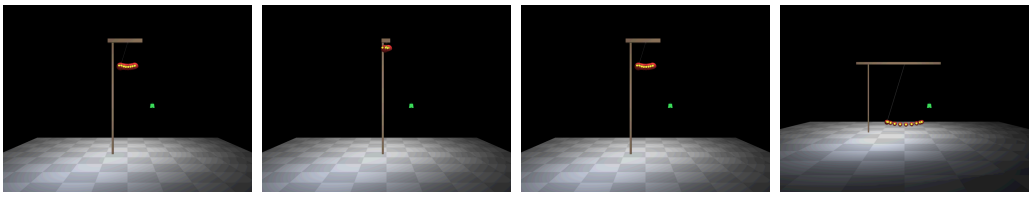
Table 7: Instrumented versus no-instrument GCE-3 on the 35-scene sweep. S/R denotes strict-or-relaxed recovery; accuracy rows use the same scorer as Table 1. A formula skeleton replaces numeric literals while retaining operators, variables, and functions. A new observation is an accepted scene request, apparatus proposal, or experiment call between two submissions. Input-token counts come from the runtime usage ledger and include the accumulated multimodal context. Image-item count alone is not a context-cost proxy because raw and instrumented images differ in size and persistence.

D Complete GPT-5.6 Interaction Example

We transcribe one complete GPT-5.6 GCE-n episode from the 35-scene sweep: the pendulum scene `s3_pendulum` with hidden exponent $\alpha = 0.5$ (period $T \propto L^{0.5}$). Images are the renders the agent received, regenerated by replaying its recorded tool calls through the same runner; rationales are quoted with minor truncation. The episode illustrates the regime identified in the main text: the agent recovers the functional form from constructed evidence, and execution counterexamples then correct its coefficient.

Initial observation and first apparatus. The agent receives one raw render at the default length and the goal statement (three low/mid/high targets; "green markers = hidden y^* targets"). It proposes a pendulum with a 20-sample strobe trail, reasoning that "the suspected changing observable is oscillation period, so a dense strobe trail should reveal the phase progression," and requests lengths 0.25, 1, and 4.

Second apparatus and first submission. Judging that "the period is temporal," the agent replaces the strobe with an eight-column filmstrip to "count phase recurrence directly." It then reads recurrence times of roughly 1.27, 2.54, and 5.08 s at lengths 0.25, 1, and 4, infers a log–log slope of 0.5 from the

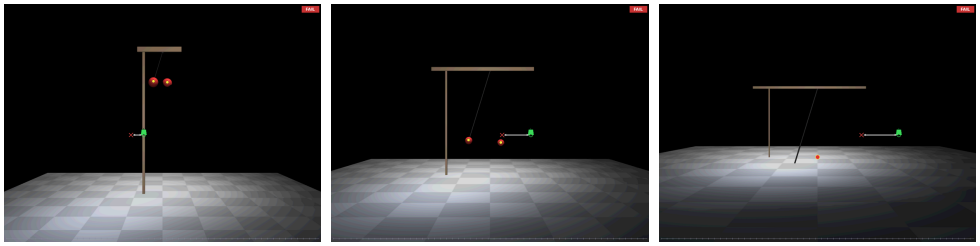


strobe preview (L=1) L=0.25 L=1 L=4

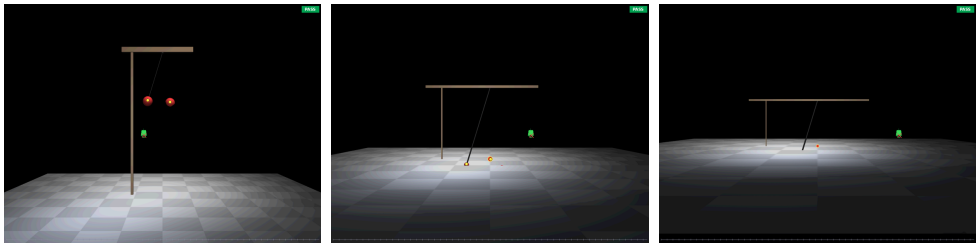
Figure 10: Accepted strobe-trail apparatus and the three requested lengths.

two 4:1 ratios, and submits `2.54245*sqrt(length_m)`. The form is correct but the coefficient is not: the oracle-fit line reports FAIL and all three targets fail (Figure 11, top).

Revision from counterexamples. Without a new observation, the agent keeps the exponent ("the 4:1 length comparison gives approximately a 2:1 recurrence-time comparison") and rescales the coefficient, noting that "the prior coefficient 2.54245 was rejected by the hidden check and by the length-1 catch." It submits `2.0*sqrt(length_m)`, which passes the oracle-fit check and all three physical targets (Figure 11, bottom); the episode ends with a Strict success. Its rewritten data table (now 1.0, 2.0, 4.0 s) is a rescaling of the earlier readings rather than a new measurement—the same candidate-conditioned re-reading analyzed in Section 6, here harmless because the form was already right.



First submission `2.54245*sqrt(length_m)`: targets 1–3 FAIL (controls 1.00, 3.15, 5.30)



Revised `2.0*sqrt(length_m)`: targets 1–3 PASS (controls 1.62, 5.09, 8.57)

Figure 11: GCE-n execution feedback. The runner inverts each submitted law to choose the control for each target and returns the rendered outcome with a pass/fail verdict; no numeric outcome is shown.

E Verified Teacher Data and a Frozen Distillation Pilot

This section documents data preparation, the evaluation protocol, and the resulting development-pilot outcomes for 8B and 32B students. We generate text-only examples of numeric observation feedback followed by a revised symbolic law using GPT-5.6 Luna with medium reasoning effort. Inputs contain the initial law and measured control–outcome pairs, with hidden simulator parameters retained only in the audit records. These are complete inputs for a text revision task, not complete visual tool-use trajectories.

Before generation, we fix four held-out scenes (freefall, circular motion, string waves, and polarization) and three validation scenes (Coulomb, Boyle, and mass-dependent freefall). All corresponding

simulator aliases are excluded from training. The resulting source pool contains 464 distinct prompts across 18 eligible training scenes. Validation contains 78 prompts from its three disjoint scenes and is reserved for checkpoint selection.

Each generated formula is evaluated on 65 uniformly spaced controls in the training scenario's default range. Retention requires finite predictions, correct signs, and log-RMSE at most 0.20. This is a training-data filter, not a sealed-test outcome. Of 88 completed examples, 81 pass this filter. We freeze the first 80 passing examples in the original source-manifest order, before training or evaluation: 80 distinct prompts spanning 17 training scenes. All seven rejected examples and the one extra verified example remain auditable; the extra example is not added to this pilot. The frozen manifest records the split, input hashes, per-example provenance, and training-file checksum. No hidden-parameter metadata is included in the exported student messages.

The pilot compares the initial law, an untrained 8B reviser, and an 8B LoRA reviser trained on these 80 examples. Four held-out scenes with two settings and three repetitions yield 24 matched outputs per reviser, but only eight independent scene–setting cells. Feedback, development, and held-out evaluation controls must remain separated; checkpoint selection uses only the validation split. A pilot used to change data or training settings is development evidence, and a subsequent confirmatory test must be reserved separately. Strict, S/R, paired improvements/regressions, JLR, and completion will be reported without treating repeated outputs as independent cells.

Pilot outcome. We train LoRA adapters (rank 16, α=32, dropout 0.05, all attention and MLP projections; learning rate 10^{-4} with linear decay; effective batch 8; one precommitted epoch; maximum length 1,024 tokens) in two data rounds. *Round 1* uses the 80 frozen GPT-5.6 examples (10 optimizer steps). *Round 2* is fixed before any round-2 evaluation: the same 80 examples plus physically verified revisions generated by Qwen3-VL-32B-Instruct-FP8 on 1,783 unique prompts (18 training scenes $\times$ up to 100 hidden settings; all pilot held-out and validation scenes excluded). Of these, 440 pass the dense-grid filter; we cap each scene at 40 examples (fixed seed), giving 424 examples (53 steps). The 8B student is trained in BF16. The 32B student is trained as 4-bit NF4 QLoRA on a single GPU with BF16 compute, because cross-GPU model sharding was unreliable on our hardware; its untrained reference uses the same 4-bit base. Training the 32B student on Qwen-32B-generated revisions is self-distillation rather than distillation from a stronger teacher.

Each of the 24 cells starts from a frozen first law (the historical GPT GCE-1 first submission at the nearest setting). Untrained and LoRA revisers receive a byte-identical prompt with the three measured feedback pairs and identical decoding (temperature 0.6, top-p 0.95, per-cell seed) and differ only in whether the adapter is applied. All states are scored on four sealed controls per cell (two interpolation and two extrapolation points) that never enter generation, training, or checkpoint selection; truncated or unparseable outputs count as failures.

Table 8 (Appendix E.1) reports all seven states.

Untrained revision is destructive at 8B: relative to the initial law it converts no failures into passes and turns three S/R passes into failures. Every trained student raises Strict recovery relative to the initial law, with hierarchical-bootstrap 95% intervals over the eight scene–setting cells (10,000 resamples) that exclude zero: $+16.7$ points $[+4.2, +33.3]$ for 8B round 1, $+29.2$ points $[+8.3, +54.2]$ for 8B round 2, $+29.2$ points $[+4.2, +62.5]$ for 32B round 1, and $+37.5$ points $[+12.5, +75.0]$ for 32B round 2. Against the matched untrained reviser, S/R rises by $+33.3$ points $[+8.3, +66.7]$ for 8B round 2 and by $+20.8$ points $[+4.2, +37.5]$ for 32B round 1, with no cell in which the untrained reviser passes and the LoRA reviser fails. Against the initial law, S/R gains remain uncertain (8B round 2: $+20.8$ points $[-25.0, +62.5]$; 32B round 1: $+29.2$ points $[-12.5, +66.7]$).

More training data sharpens the recovered laws (lower median JLR, higher Strict) but does not monotonically raise S/R. Round-2 students write longer revisions: verified Qwen targets are about twice as long as GPT targets (median 415 versus 230 characters). At a 1,536-token cap this produces truncation failures concentrated in the low settings of freefall and circular motion; raising the 8B cap to 4,096 tokens recovers most of them (S/R 16 to 17, valid outputs 20 to 22). Under the precommitted decision rule (S/R above the initial law, paired net gain of at least 12.5%, lower median JLR, and no material loss of valid outputs), 8B round 1 is classified as mitigation only, whereas 8B round 2 (4,096 cap) and both 32B students meet the criteria for expansion. Because the same 24 cells were used across rounds, these results are development evidence; a confirmatory claim requires a fresh sealed

test. An earlier Qwen-32B Instruct teacher-generation run (23 scenes, including pilot held-out and validation scenes) is a separate exploratory data source and is not used for training. It is neither the matched 120-continuation test nor a source of either pilot round. Intermediate generation and parsing counts are not teacher-quality rankings. A future teacher-source comparison would require the same training exclusions, unique-prompt coverage, sample count, training budget, and evaluation protocol; a comparison of parseable versus physically verified labels would likewise require a separate matched design.

E.1 Sealed-Control Results of the One-Revision Pilot

The open-weight replication raises a practical question: is the inability to use counterexamples a fixed property of smaller backbones, or can it be learned? We train LoRA revisers on revisions whose laws are physically verified on a dense control grid, and evaluate them on four held-out scenes (24 cells, eight independent scene–setting units) using sealed interpolation and extrapolation controls that never enter generation, training, or checkpoint selection (Appendix E). Round 1 uses 80 GPT-5.6 revisions; round 2 adds 344 verified Qwen3-VL-32B revisions (424 in total).

State	Cap	Strict	S/R	Med. JLR	Valid
Initial law (no revision)	–	12.5%	50.0%	0.724	100.0%
8B untrained	1,536	12.5%	37.5%	0.693	95.8%
8B LoRA, round 1 (80)	1,536	29.2%	50.0%	0.661	87.5%
8B LoRA, round 2 (424)	4,096	41.7%	70.8%	0.249	91.7%
32B untrained (4-bit)	1,536	33.3%	58.3%	0.210†	70.8%
32B LoRA, round 1 (80)	1,536	41.7%	79.2%	0.259	100.0%
32B LoRA, round 2 (424)	1,536	50.0%	70.8%	0.128	79.2%

Table 8: Sealed-control distillation pilot (four held-out scenes, two settings, three repetitions; the eight scene–setting cells are the independent units). Cap is the generation-token limit. The 8B untrained reviser is unchanged at a 4,096-token cap. †Median over the 17 valid outputs only.

Untrained revision is destructive at 8B: relative to the unrevised first law it converts no failures into passes and turns three S/R passes into failures. Every trained reviser raises Strict recovery over that law, with 95% cluster-bootstrap intervals excluding zero (from $+16.7$ points for 8B round 1 to $+37.5$ points for 32B round 2), and none loses a cell its untrained counterpart solves under S/R. Against the untrained reviser, S/R rises by $+33.3$ points ($[+8.3, +66.7]$) for 8B round 2 and $+20.8$ points ($[+4.2, +37.5]$) for 32B round 1. More data sharpens laws (lower JLR, higher Strict) without monotonically raising S/R, partly because round-2 students write longer revisions that hit the generation cap. The 32B round-2 student is partly self-distilled and all 24 cells were reused across rounds, so these are development results, not a confirmatory test.

Numeric versus visual outcome feedback. The revisions above receive measured outcomes as numbers. In the full loop, GCE-n returns only per-target pass/fail, the inverted control, and execution renders, so the agent must read outcomes from pixels. Given the same 588 prompts in this faithful format, the Qwen3-VL-32B teacher usually corrects the *scale* of its law (normalization error falls for 64% of revisions) but rarely its *form* (shape error falls for 24%, and its median rises from 0.26 to 0.31); only 3.9% of revisions pass the dense verifier, against 16.8% with numeric outcomes (Appendix E.3). Revisers trained on the verified laws in this image format do not beat their untrained counterparts on the sealed controls (8B: Strict 16.7% and S/R 41.7% for both; 32B: 20.8% and 45.8% versus 29.2% and 54.2%). Pass/fail feedback on a picture tells the model which way it is off, not how its law is shaped; this bounds the one-revision distillation result to numeric feedback.

E.2 End-to-End Fine-Tuning: Seeds, Folds, and the 32B Student

Data and training. We replay every GPT-5.6 GCE-n episode of the 35-scene sweep through the unified runner, regenerating the renders the agent saw, and keep episodes whose final law is an S/R success under the main-table scorer. Episodes from the seven held-out scenes (freefall, mass-dependent fall, orbit, Boyle, Coulomb, string wave, polarization) are excluded, leaving 85 training

episodes from 24 scenes. The loss is applied to every assistant turn. Qwen3-VL-8B is trained with LoRA (rank 16) in bfloat16; Qwen3-VL-32B is trained with 4-bit NF4 QLoRA, images capped at about 10^5 pixels, and episodes truncated at turn boundaries to 10,000 tokens so that training fits one 48 GB GPU. The 32B student and its untrained baseline are both evaluated at this reduced resolution with an FP8 checkpoint.

8B: seeds and a second fold. On the 35 held-out cells, S/R per sampling seed is 20.0%, 42.9%, and 28.6% for the untrained model and 31.4%, 20.0%, and 31.4% after fine-tuning; the seed-averaged paired difference is -1.0 cells ($[-4.7, +2.7]$). On a second fold that holds out seven different scenes (pendulum, ramp, launch, decay, Hooke, Ohm, single slit) and trains on 92 episodes, S/R falls from 22.9% to 8.6% (one seed). Inspection shows the student reverting to textbook forms, for example a square-root pendulum law when the hidden exponent is 1 and an inverse single-slit law when it is -0.5.

32B. On the first fold, fine-tuning raises Strict from 8.6% to 25.7% ($+6$ cells, $[+2, +10]$) and S/R from 42.9% to 57.1% ($+5$ cells, $[+2, +9]$), with seven cells solved only after training and one only before; valid laws rise from 88.6% to 97.1%. This is a single seed and fold.

E.3 Revision from Visual versus Numeric Outcome Feedback

We give Qwen3-VL-32B-Instruct (FP8) the same 588 revision prompts (seven training scenes, one sample each) in two formats. In the *numeric* format, the prompt lists measured outcomes at the three targets. In the *faithful* format, it contains exactly what GCE-n returns in the full loop: the goal statement, per-target pass/fail, the control inverted from the agent's law, and three execution renders, with no outcome values. Each revised law is compared with the hidden law on 65 uniformly spaced controls. We decompose the signed log residual into a *shape* error (its standard deviation, measuring functional form) and a *normalization* error (the absolute value of its mean, measuring overall scale), so that $\mathrm{JLR}^2 = \mathrm{shape}^2 + \mathrm{norm}^2$.

	Faithful (renders)	Numeric
Usable revisions	587	528
Verified (JLR ≤ 0.2, correct sign)	3.9%	16.8%
Monotone trend correct after revision	100%	99.8%
Normalization error reduced	64%	70%
Median normalization error (initial $\rightarrow$ revised)	1.31 $\rightarrow$ 0.86	1.28 $\rightarrow$ 0.57
Shape error reduced	24%	52%
Median shape error (initial $\rightarrow$ revised)	0.26 $\rightarrow$ 0.31	0.24 $\rightarrow$ 0.20
Revised shape ≤ 0.2	181	263
and normalization ≤ 0.2	38	112

Table 9: One-step revision by the same teacher from visual and numeric outcome feedback on identical prompts. Percentages of reductions are over usable revisions. The prompts cover the first seven training scenes processed and are not a random sample of the benchmark.

With renders only, the teacher reliably infers the direction of its error and rescales its law, but rarely improves the law's form; with numbers, it improves both. Faithful-format students are trained with the same recipe as round 2 (424 verified laws, law-only targets, one epoch; 8B in BF16 with mean training loss 0.66, 32B as 4-bit QLoRA with mean loss 0.61) and evaluated on the same 24 sealed cells; the 32B base and adapter are served from the FP8 checkpoint. Neither student improves on its untrained counterpart (LoRA minus base, 8B: Strict $+0$, S/R $+0$; 32B: Strict -2 $[-7, +2]$, S/R -2 $[-7, +2]$).

F Additional Results and Analyses

This section collects results summarized briefly in the main text.

F.1 Robustness to the Evaluated Controls and Per-Family Results

The runner's visible law-fit bit is computed on the same reference grid used for scoring. To test whether the final laws merely fit those controls, we re-score each final GPT-5.6 GCE-n law at the midpoints between consecutive grid controls, which never enter feedback. The 95% interval resamples scenes with replacement (10,000 resamples).

Scoring controls	Strict (%)	S/R (%)	Median JLR
Reference grid (main table)	23.4 [16.0, 32.0]	63.4 [53.1, 73.1]	1.205
Grid midpoints (never used in feedback)	24.0	62.3	1.202

Table 10: GPT-5.6 GCE-n final laws scored on the reference grid and on unseen interpolation midpoints (175 cells). Only 1 Strict and 4 S/R verdicts change.

Family	Cells	Strict	S/R
Mechanics	55	11	39
Thermal physics and transport	35	2	15
Fluids and gases	20	5	17
Waves and response	15	2	7
Optics	25	16	21
Electromagnetism	25	5	12
All	175	41	111

Table 11: GPT-5.6 GCE-n recovery by physical family. Exponential, linear, and logarithmic scenes (thermal decay and cooling, capacitor, Ohm, logarithmic response) can succeed only through Strict recovery.

F.2 Reference Baseline, Difficulty, Failure Types, and Benchmark Statistics

Textbook recall. For each power-law cell we submit the textbook exponent with the coefficient fitted to the hidden outcomes on the evaluator grid, an oracle advantage no agent has. Non-power-law cells have no textbook exponent and count as failures. This reference isolates what memorizing familiar physics can achieve.

Distance from the textbook regime. Each scene has a central setting p_0, two near settings $p_0 \pm 0.5$, and two far settings $p_0 \pm 2.0$ (Table 4).

	GPT-5.6 GCE-n		Textbook recall (oracle coef.)		
Settings	Strict	S/R	Strict	S/R	Cells
Center p_0	8 (22.9%)	20 (57.1%)	14	18	35
Near $p_0 \pm 0.5$	20 (28.6%)	44 (62.9%)	6	7	70
Far $p_0 \pm 2.0$	13 (18.6%)	47 (67.1%)	0	0	70
All	41 (23.4%)	111 (63.4%)	20 (11.4%)	25 (14.3%)	175

Table 12: Recovery by distance from the central setting. Textbook recall collapses away from the familiar regime, whereas the agent's S/R does not; its Strict recovery drops on far settings, where calibrating the coefficient is hardest. The recall reference has median JLR 0.624 only because its coefficient is fitted to the hidden outcomes.

Failure types. We classify each final law by its evaluator-grid residuals (Appendix H): *correct* (Strict), *right form, wrong scale* (not Strict, Shape ≤ 0.2), *wrong form* (Shape > 0.2), or *no valid law* (missing, unparseable, or non-finite).

Benchmark statistics.

Agent	Correct	Right form, wrong scale	Wrong form	No valid law
GPT-5.6 GCE-n	41	42	85	7
Qwen3-VL-8B GCE-n	6	22	115	32

Table 13: Failure types over the 175 cells. For GPT-5.6, as many laws have the right form but a miscalibrated scale (42) as are fully correct (41); the 8B model mostly fails on form or never submits a valid law.

Scenes / physical families	35 / 6
Law families (cells)	power law 150, exponential 15, linear 5, logarithmic 5
Scene–setting cells	175 (five hidden settings per scene)
Evaluator grid	5–7 controls per scene
Interaction budget	10 actions, at most 5 apparatus proposals
GPT-5.6 GCE-n per episode	8.00 tool calls, 4.21 submissions, 22.6 images, 442.5k input tokens
Qwen3-VL-8B GCE-n per episode	10.07 tool calls, 9.9 turns

Table 14: Benchmark and interaction statistics for the 35-scene studies (GPT-5.6 values from Table 7).

F.3 Historical GPT-4.1 Settings

The historical GPT-4.1 runs used temperature 0.2, seed 0, a 10-turn cap, at most five apparatus proposals, and a 4096-token output cap. Their logged OSBD label did not enforce the intended one-shot contract; the audit and historical settings are retained in Appendices G.1 and G.5. GPT-4.1 versus GPT-5.6 also changes renderer and runtime, and is interpreted descriptively. The matched Qwen replication and the distillation pilot are reported in Appendix F.5 and Section 5.4.

Success rates use all planned task cells; missing or invalid submissions count as failures. Finite-JLR summaries exclude invalid expressions where stated in the table. Task-cell matching in the instrument study does not make the two trajectories identical. The feedback study has 60 matched strata per arm but only 20 independent scene–setting cells because three repetitions share a cell. We report paired discordances descriptively and do not treat these 60 rows as independent experimental units. Historical per-task standard deviations describe heterogeneity, not repeated-seed uncertainty.

F.4 Limits of the Matched Feedback Study

These results have three important limits. The 60 rows per arm contain three stochastic repetitions of only 20 unique scene–setting cells, so the matched counts are descriptive rather than 60 independent inferential units. The shared first law comes from a historical GCE-1 trajectory, and several negative-offset settings are effectively unrecoverable in one revision. Finally, numeric text feedback removes image variation but also supplies different quantities of feedback across arms; target coverage is not isolated from feedback volume. A full visual rerun with multiple revision rounds is required before transferring the effect size to the end-to-end protocol.

F.5 Across Model and Runtime: Full Analysis

Backbone/runtime	Condition	Strict	S/R	Median JLR	Mean JLR $\pm$ SD
GPT-4.1 historical	FGD	2.6%	43.6%	1.710	2.299 ± 2.807
GPT-5.6 Luna	FGD	23.1%	71.8%	1.303	2.630 ± 6.658
GPT-4.1 historical	GCE-1	7.7%	48.7%	1.335	2.119 ± 2.082
GPT-5.6 Luna	GCE-1	**41.0%**	64.1%	**0.436**	$\mathbf{0.791 \pm 0.865}$
GPT-4.1 historical	GCE-3	7.7%	56.4%	0.994	1.561 ± 1.781
GPT-5.6 Luna	GCE-3	23.1%	61.5%	0.767	0.987 ± 1.015

Table 15: Descriptive model/runtime sensitivity on 13 shared power-law tasks and three hidden settings per task. The comparison is not a model-only ablation: the GPT-5.6 run also uses redesigned pixels, ordered PNG input, and a new execution runner. The large GPT-5.6 FGD variance is driven by a Weber-task outlier; its median JLR nevertheless improves from 1.710 to 1.303.

Model	Feedback	Strict	S/R	Parseable
8B Instruct	GCE-1	10.0%	25.0%	96.7%
8B Instruct	GCE-3	10.0%	26.7%	100.0%
32B Instruct	GCE-1	16.7%	26.7%	90.0%
32B Instruct	GCE-3	6.7%	23.3%	91.7%

Table 16: Matched Qwen one-revision results. Both models use FP8 Instruct checkpoints. These numeric-feedback continuations are distinct from full visual-agent episodes and from teacher-data generation. Counts are descriptive; repeated generations are not independent scene samples.

We finally compare the archived GPT-4.1 sweep with GPT-5.6 Luna on the 13 power-law tasks shared by both runs. Each row contains the same 13 task identities and three hidden settings per task, for 39 episodes. We retain only FGD, GCE-1, and GCE-3: the archived row labeled OSBD received corrective feedback and is not a valid model comparison. Final GPT-5.6 expressions are rescored offline with the same Strict threshold, relaxed-exponent rule, and scene-specific evaluator grids used for the GPT-4.1 table.

The newer stack improves recovery in every corresponding row. Pooling the three protocols descriptively, Strict rises from 6.0% to 29.1% (29.1%), S/R from 49.6% to 65.8%, and mean JLR falls from 1.993 to 1.474. The largest Strict increase occurs under GCE-1 (+33.3 percentage points), whereas GCE-3 gains only +15.4 points in Strict and +5.1 points in S/R. Protocol rankings also change: GPT-4.1 favors GCE-3 over GCE-1 in S/R (56.4% versus 48.7%), while GPT-5.6 favors GCE-1 in Strict and JLR (41.0% and 0.791 versus 23.1% and 0.987). FGD attains the highest GPT-5.6 S/R (71.8%) but retains a heavy-tailed error distribution.

This pattern shows that conclusions about the best feedback protocol are not invariant to the model/runtime generation. It is consistent with a stronger agent extracting more value from sparse feedback, reducing the relative gain from three target-conditioned executions. It does not establish that model strength alone causes the shift, because renderer, prompt/input format, and runner changed at the same time. A clean model-scaling claim requires replaying both backbones on identical images and histories; Table 15 should therefore be read as a historical sensitivity analysis.

MATCHED OPEN-WEIGHT REPLICATION

We replay the matched numeric-text, one-revision protocol with Qwen3-VL-8B-Instruct-FP8 and Qwen3-VL-32B-Instruct-FP8. Each model receives 10 scenes, two settings, three repetitions, and GCE-1/GCE-3, giving 120 continuations per model. Matched arms share the same initial law; both models use the same task and seed manifest and a 1,024-token generation cap. Table 16 reports all attempted continuations, counting invalid or truncated answers as failures. The independent analysis units are the 20 scene–setting cells, not the 60 repeated outputs per arm.

The larger model shows no consistent advantage in this comparison: pooled Strict is 11.7% versus 10.0% for 8B, while S/R is 25.0% versus 25.8%. The 8B model produces more parseable outputs (98.3% versus 90.8%). Unlike the GPT-5.6 matched result, additional feedback does not improve Strict for either Qwen model. This bounds the transfer of the feedback benefit across backbones under this protocol; it is not evidence of statistical equivalence or a general inverse scaling law. Section 5.4 tests whether this behavior can be changed by training.

F.6 WHEN ADDITIONAL COUNTEREXAMPLES DISPLACE A USEFUL LAW

In the 32B matched comparison, six repeated trials pass Strict only under GCE-1 and none pass only under GCE-3; the corresponding S/R discordances are three versus one. All six Strict losses have parseable answers in both arms. Truncation is therefore not their explanation: GCE-1 has six truncated outputs and GCE-3 has five. Among the 50 pairs parseable in both arms, GCE-3 lowers JLR in 16, raises it in 18, and leaves it unchanged in 16; the median paired difference is zero. The aggregate Strict decline reflects threshold-crossing failures concentrated in viscosity, launch, and collision, not a uniform deterioration across tasks.

For viscosity, GCE-1 retains a quadratic radius law, `1.21*radius**2` (JLR 0.1906), whereas GCE-3 changes it to `1.21*radius**3` (JLR 1.1221). In launch, approximately linear observed outcomes with coefficient 0.102 lead to quadratic revisions `0.014*speed_mps**2` and `0.067*speed_mps**2` in two repetitions. In collision, the reasoning recovers an observed outcome-to-control ratio of 0.65 but the final expression uses `0.8*drop_height`; the GCE-1 answer uses 0.65. These cases are consistent with failures to distinguish a desired target, the control chosen using the old law, and the actual observed outcome, or to carry a correct calculation into the final answer. They motivate an evidence-use hypothesis, not a proven common causal mechanism. Testing that hypothesis requires a controlled feedback-format intervention.

F.7 Further Interaction Case Studies

Thin lens: grounding improves the hypothesis family. Without an instrument, the agent assigns the raw frames values ranging from 3.2×10^5 to 10^{20} and then rewrites the same frames as relative values $1, 2, 5, 10, 30, 100$. It successively tries `10**object_distance`, `exp(2*object_distance)`, `object_distance**object_distance`, and other exponential families. The instrumented trajectory instead preserves auditable pixel readings—lens center near 640 px and object centers near 522, 468, 426, 340, and 212 px—and uses the image transition near control 14 to propose the physically appropriate rational thin-lens family. Neither trajectory reaches the correct final law; the contrast shows improved grounding and family selection, not guaranteed success.

Torricelli: stable measurements versus changing stories. The no-instrument agent interprets the same jet frames, without re-observing, first as sublinear, then approximately linear, then superlinear, and finally saturating. It submits square-root, affine, quadratic, and rational families in turn. The instrumented trajectory repeatedly reuses the same endpoint readings of approximately 48, 95, 188, 282, and 376 pixels. It also fails to identify the correct calibrated law, but its evidence remains consistent across revisions; the remaining error is calibration and feedback use rather than a data table that changes with the candidate.

Together, the aggregate and textual evidence suggests that instruments are valuable as an auditable measurement interface even when endpoint accuracy is unchanged. The unresolved bottleneck is policy: agents rarely place the instrument inside an adaptive observe–submit–diagnose–re-observe–revise loop. The three cases are mechanism illustrations rather than an estimate of the corpus-wide frequency of pseudo-measurement; a pre-specified blinded trace annotation would be required for that population claim.

G Historical GPT-4.1 13-Scene Sweep

This appendix collects every artifact of the archived GPT-4.1 sweep on 13 power-law scenes (three hidden exponents each, 39 episodes per condition) and its earlier 15-scene suite. These results predate the 35-scene suite, use legacy scene identifiers, and are not pooled with any main-text study.

G.1 Protocol and Reproducibility Audit

We audited both the released runner and the serialized tool messages from all 39 episodes in each logged condition. Table 17 distinguishes the current intended contract from the behavior of the archived alpha-sweep artifacts. The key correction is that the result directory named `OSBD` is not a no-feedback run: all 39 episodes contain corrective oracle feedback, 100.0% contain multiple valid submissions, and the mean number of submissions is 4.38. FGD likewise exposes corrective oracle feedback in all 39 episodes (mean 4.51 submissions). GCE and GCE-n expose both the binary oracle-law-fit result and physical execution feedback in all 39 episodes, with mean submission counts 5.38 and 5.72. The current released runner hard-terminates a non-goal `-shots 1` run after its first valid submission, but those corrected OSBD runs are not the source of the numbers reported in Table 21.

Condition	Law-fit bit	Ref. renders	Execution feedback	n	Reported?
OSBD (current contract)	No	No	No	0	No
Legacy-1 (logged OSBD)	Yes	On failure	No	0	Yes (39)
FGD	Yes	On failure	No	0	Yes (39)
GCE	Yes	No	Pass/fail + render	1	Yes (39)
GCE-n ($n = 3$)	Yes	No	Pass/fail + render	3	Yes (39)

Table 17: Agent-visible feedback contract. "Oracle fit bit" is derived from the evaluator reference grid. Numeric ground truth, per-control predictions, and JLR rows are never copied into the agent message. FGD and Legacy-1 failure messages additionally reveal the endpoint control values and their rendered animations.

Item	Main-sweep setting and provenance
Agent model	`gpt-4.1`, the alias stored in every JSON trace. The exact provider snapshot identifier was not serialized.
Sampling	Model seed 0 in every trace; temperature 0.2 and maximum output 4096 tokens in the released scripts/runner. No repeated model seeds.
Interaction limits	10 model turns and at most 5 apparatus proposals. The legacy scripts stated one submission for the OSBD label and four for other conditions, but the historical runner did not enforce those declarations; realized counts are reported in Table 19.
Sweep	13 power-law scenes, three counterfactual exponents per scene, one run per scene–exponent cell and condition (39 episodes).
Evaluator grid	Five controls for `s1,s3,s4,s6,s9,s10,s20`; six for `s2,s8,s17,s18`; seven for `s19,s21`.
Run artifacts	The two alpha-sweep batches are dated June 29 and July 1, 2026. Legacy JSON stores messages, tool traces, model alias, scene seed, and scores, but not the source commit, provider snapshot, temperature, timeout, or retry settings.
Random-action control	Schedule seed 101; one generated apparatus seed per scene–alpha assignment; 19 relevant and 20 irrelevant assignments.

Table 18: Reproducibility record for the reported alpha sweep. Values absent from legacy JSON are identified as script/released-runner settings rather than claimed as independently serialized metadata. New runner traces store an explicit protocol, feedback contract, and run configuration.

G.2 Interaction and Tool-Use Audits

Table 19 reports realized interaction rather than nominal budgets. Table 20 summarizes how agents use and revise the visual-instrument DSL. These diagnostics support the main-text analysis but are not primary law-recovery outcomes.

Condition	Turns ± SD	Images ± SD	Submits ± SD	Proposals ± SD
Legacy-1	10.00 ± 0.00	20.85 ± 3.12	4.38 ± 1.43	1.05 ± 0.22
FGD	9.87 ± 0.80	21.54 ± 3.04	4.51 ± 1.32	1.15 ± 0.43
GCE	9.79 ± 0.86	15.69 ± 3.31	5.38 ± 0.81	0.97 ± 0.36
GCE-n ($n = 3$)	9.74 ± 1.04	29.79 ± 7.55	5.72 ± 1.23	0.97 ± 0.16

Table 19: Mean realized interaction per episode with sample SD across scene–alpha settings. Images are image-bearing message items, not unique clips or frames.

Condition	Rel./uniq.	Collapse (%)	Retunes (%)	Improved (%)
Legacy-1	92.3/1.21	79.5	2.6	0.0
FGD	92.3/1.38	66.7	0.0	0.0
GCE	87.2/1.13	76.9	0.0	37.5
GCE-n ($n = 3$)	89.7/1.18	79.5	0.0	25.0

Table 20: Tool-use diagnostics computed from saved apparatus YAML and submission traces. Rel./uniq. gives relevant-episode percentage and mean unique types; collapse is the dominant-tool percentage. Retunes is the episode percentage with an effective parameter change; Improved is the percentage of qualifying apparatus-separated submission pairs whose offline JLR decreases.

G.3 Archived Predictive and Observation Diagnostics

The completed 13-scene alpha sweep is retained as a diagnostic of the earlier oracle-visible runner. All four logged conditions contain every completed and scored episode in that sweep. The first row retains its historical OSBD label but is marked Legacy-1 because its saved traces expose the same type of corrective oracle feedback as FGD; it is therefore not a no-feedback baseline.

Logged condition	Strict (%)	S/R (%)	Med. JLR	Mean JLR ± SD	Shape err ± SD	Norm. err ± SD
Legacy-1 (logged OSBD)	0.0	41.0	1.575	2.552 ± 2.776	0.969 ± 0.998	1.623 ± 2.058
FGD	2.6	43.6	1.710	2.299 ± 2.807	0.823 ± 0.722	1.380 ± 1.397
GCE, one target	7.7	48.7	1.335	2.119 ± 2.082	0.821 ± 0.634	1.307 ± 0.853
GCE-n ($n = 3$)	7.7	**56.4**	**0.994**	$\mathbf{1.561 \pm 1.781}$	$\mathbf{0.533 \pm 0.468}$	$\mathbf{0.816 \pm 0.701}$

Table 21: Archived predictive world-model diagnostics on the complete alpha sweep. S/R denotes strict-or-relaxed recovery, and JLR measures predictions on a scene-specific evaluator grid of five to seven controls. Continuous means report sample SD across scene–alpha episodes.

Figure 12 compares visual exposure with observation structure in the same runner. No-instrument episodes receive slightly more image-bearing messages than instrumented Legacy-1, so image quantity alone does not explain the descriptive difference. The balanced-random condition preserves proposal timing and the turn cap but differs in its prompted, unenforced submission budget; it is a descriptive selection control rather than a matched causal ablation.

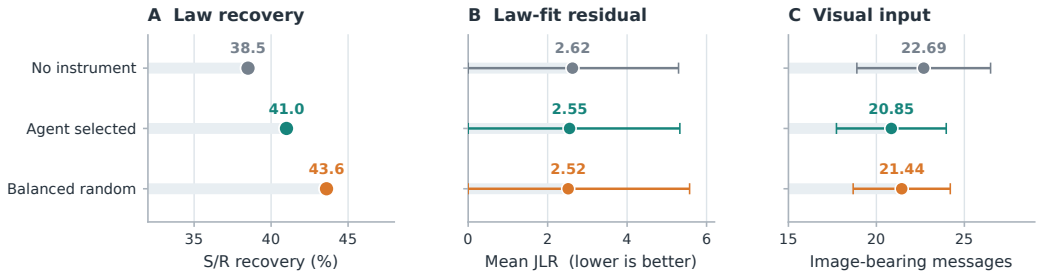


Figure 12: Archived observation-action comparisons. **A** reports strict-or-relaxed recovery, **B** mean JLR (lower is better), and **C** image-bearing messages. Error bars in **B–C** show sample SD across scene–alpha settings.

G.4 Historical Interaction Diagnostics

These diagnostics concern the archived 13-scene GPT-4.1 sweep. They do not describe the current 35-scene instrument comparison. In particular, its context-exhaustion and formula-change patterns should not be transferred to the newer runner.

G.4.1 Evidence Gathering and Instrument Adaptation

Trace diagnostics explain why plausible observation actions do not necessarily identify a better model. Across conditions, 87.2–92.3% of episodes contain at least one scene-relevant tool, but the agent uses only 1.13–1.38 unique tool types per episode and collapses to one dominant type in 66.7–79.5% of episodes. More importantly, fewer than 1% of episodes change the parameters of an already selected tool type. Current agents generally choose an apparatus once rather than adapting its scale, spacing, sampling interval, or range after a prediction fails.

Appendix Table 20 reports the full trace audit. The improvement rate after an apparatus update is too sparsely supported to serve as a stable metric: only 29 qualifying submission pairs exist across all episodes. It is nevertheless informative that the multi-target advantage does not coincide with more apparatus adaptation. The current agent usually treats the tool as a fixed overlay rather than as a measurement whose parameters should be refined.

G.4.2 Revision under Feedback

Legacy-1 and FGD obtain 41.0% and 43.6% strict-or-relaxed recovery, respectively, with mean reference-grid JLR 2.552 and 2.299. This difference cannot estimate the effect of a correctness bit: the trace audit finds corrective oracle feedback in all 39 episodes of both rows, and every legacy episode contains multiple valid submissions. The intended no-feedback OSBD condition must be rerun under the current hard one-shot termination before that ablation can support a causal claim.

Among the two execution conditions, GCE-n with $n = 3$ has higher strict-or-relaxed recovery than single-target GCE (56.4% vs. 48.7%) and lower mean JLR (1.561 vs. 2.119), median JLR (0.994 vs. 1.335), and shape-exponent error (0.533 vs. 0.821). Strict recovery is unchanged at 7.7%. Both expose the same binary oracle-law-fit result; their logged difference therefore concerns the complete one-target versus three-target execution protocols, not oracle access.

GCE and GCE-n have similar turn and submission counts. By design, however, GCE-n renders three target outcomes and therefore receives 29.79 image-bearing message items per episode, compared with 15.69 for GCE. These additional target-conditioned observations are part of the multi-target feedback mechanism rather than a separate input source. The comparison therefore establishes that the complete GCE-n protocol outperforms single-target GCE in the evaluated setting; it evaluates the protocol as a whole rather than isolating target count from the evidence generated at those targets.

Figure 14 reports the complete attempted sweep. The last scored submissions obtain similar law-fit accuracy with and without instruments, so instruments do not improve last-submission accuracy in this comparison. They do substantially improve context efficiency and protocol completion. The

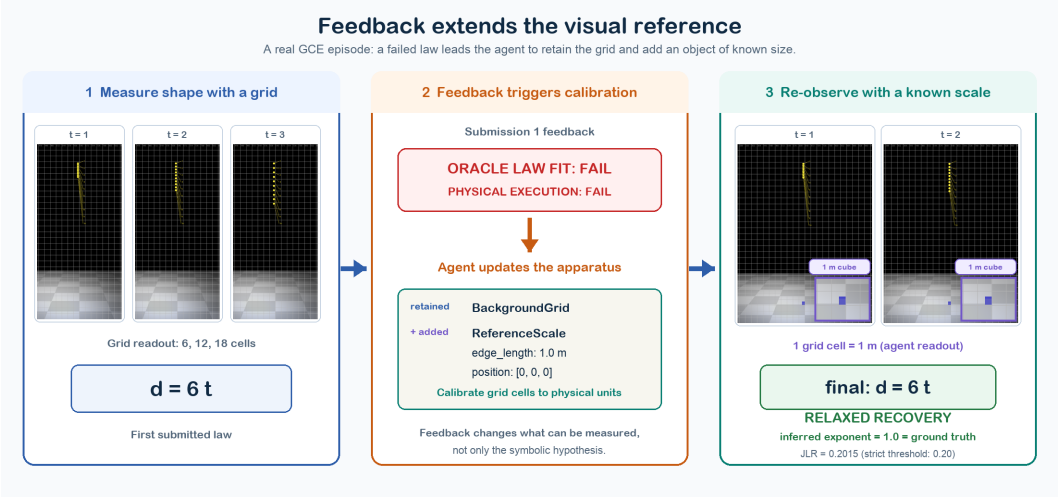


Figure 13: Feedback-triggered observation and model revision in a GCE freefall episode with hidden exponent $\alpha = 1.0$. The agent first adds a BackgroundGrid and proposes $d = 6t$. After the visible oracle-law-fit bit and physical execution both fail, it adds a 1-m ReferenceScale cube to calibrate the grid. The revised model retains the correct linear scaling ($\hat{\alpha} = 1.0$), yielding relaxed recovery; its coefficient remains imperfect, with reference-grid JLR 0.2015 just above the strict threshold of 0.20.

no-instrument protocol uses 51.62 image-bearing message items per episode, 73% more than the instrumented protocol's 29.79, because repeated requests for raw animations expand into multiple keyframes. Although every no-instrument episode produces a scored submission, only 15.4% reach the normal protocol stop; the remainder continue requesting raw visual evidence until the accumulated history exhausts the available context. All instrumented episodes complete normally. Explicit instruments therefore function as a compact, operationally viable evidence channel that allows the multi-target feedback loop to finish.

The submission traces further indicate a difference in how the two conditions update their models. Without instruments, 23.1% of episodes retain the same symbolic formula skeleton across every submission, compared with 12.8% with instruments. The no-instrument condition also ends with an affine or linear hypothesis in 28.2% of episodes, compared with 15.4% for the instrumented condition. This pattern is consistent with feedback-driven local numerical search: without an explicit measurement reference, the agent more often tunes constants or exponents within an existing guess instead of reconstructing the hypothesis from newly organized visual evidence. Thus, similar final accuracy can mask different identification processes: instrumented agents more often reconsider the model family, whereas no-instrument agents more often optimize within an initial template.

G.4.3 Transitions Across Submissions

Figure 15 asks whether the final aggregate advantage emerges through world-model revision within completed interaction traces. This is a post-hoc analysis of intermediate submissions, not a rerun under shorter interaction budgets. All four archived conditions contain corrective feedback, so the curves cannot contrast feedback with no feedback. Legacy-1 decreases from its first submission, while the other three rows end with higher recovery; the largest change occurs under GCE-n ($+15.4$ percentage points), whose strict recovery also rises from 2.6% to 7.7%. For GCE-n, nine episodes move from failure to recovery while three move in the opposite direction. The exact paired McNemar test is not significant ($p = 0.146$), so the trajectory is descriptive rather than standalone evidence of a population-level effect. Moreover, the curves are non-monotonic and final mean JLR is higher than at the first submission in every condition. Structured feedback therefore helps some episodes revise the model's scaling form enough to cross the exponent-based recovery criterion, but it does not yet produce monotonic convergence of the complete expression.

The divergence between recovery rate and JLR clarifies what is and is not learned across submissions. Early hypotheses are often weakly constrained by a small set of visual readouts and may fit one

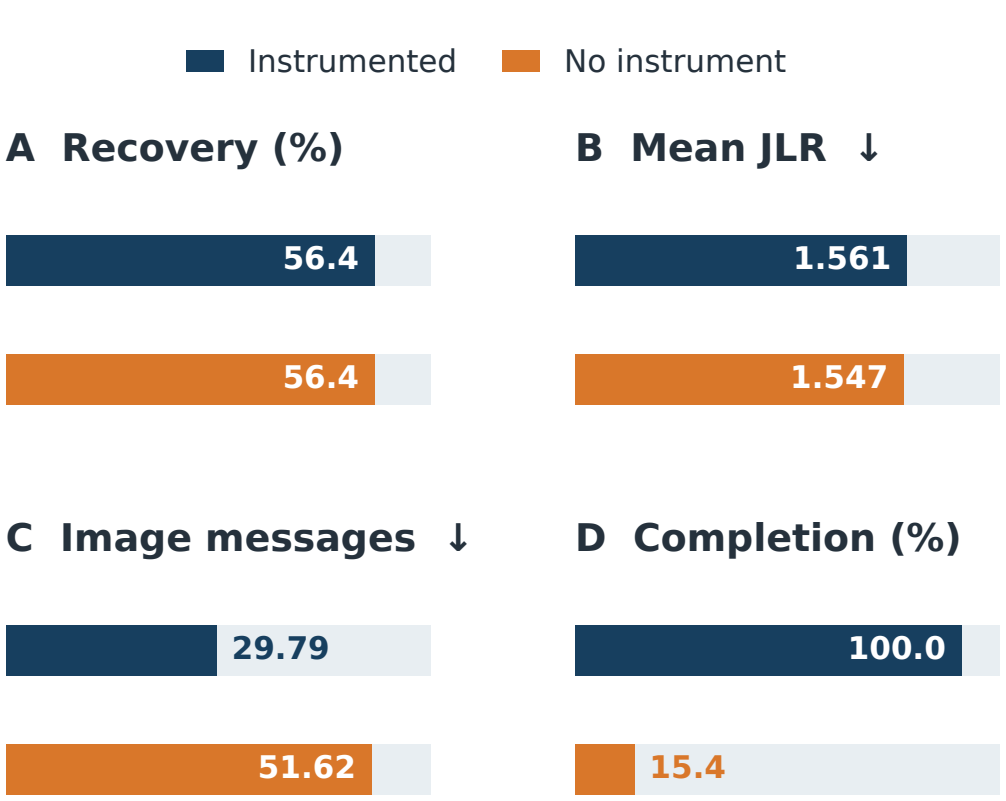


Figure 14: Instrumented versus no-instrument multi-target model revision. **A–B** show that last-submission model recovery and mean JLR are similar. **C–D** expose the operational difference: no-instrument interaction uses 73% more image-bearing messages and reaches normal completion in only 15.4% of episodes, compared with 100.0% under instrumented observation. Accuracy uses the last scored submission from every setting.

operating region or instantiate a plausible formula family without recovering a globally predictive model. Subsequent feedback can move the exponent toward the counterfactual regime and thereby raise strict-or-relaxed recovery, while coefficient estimates and reference-grid fit simultaneously deteriorate. This resembles the instrumented versus no-instrument comparison only at the level of final recovery: similar endpoint accuracy can conceal different discovery processes. Without instruments, traces more often preserve the initial formula template and accumulate raw visual context; with instruments, observations are organized around explicit references that can support model-family revision. Visual tools therefore do not guarantee a better world model or monotonic improvement. They provide a compact measurement interface from which later revision is more evidence-grounded and operationally efficient.

G.5 Historical Evaluation Settings

The following scoring details and settings document the archived GPT-4.1 protocol, not a common configuration for every study in the main text.

G.5.1 Metrics and Analysis Units

We score each final symbolic model with two complementary criteria. The primary metric is joint log residual (JLR), its predictive error against ground-truth outcomes on the scenario's evaluator reference grid. Seven scenarios use five controls, four use six, and two use seven. A *strict pass*

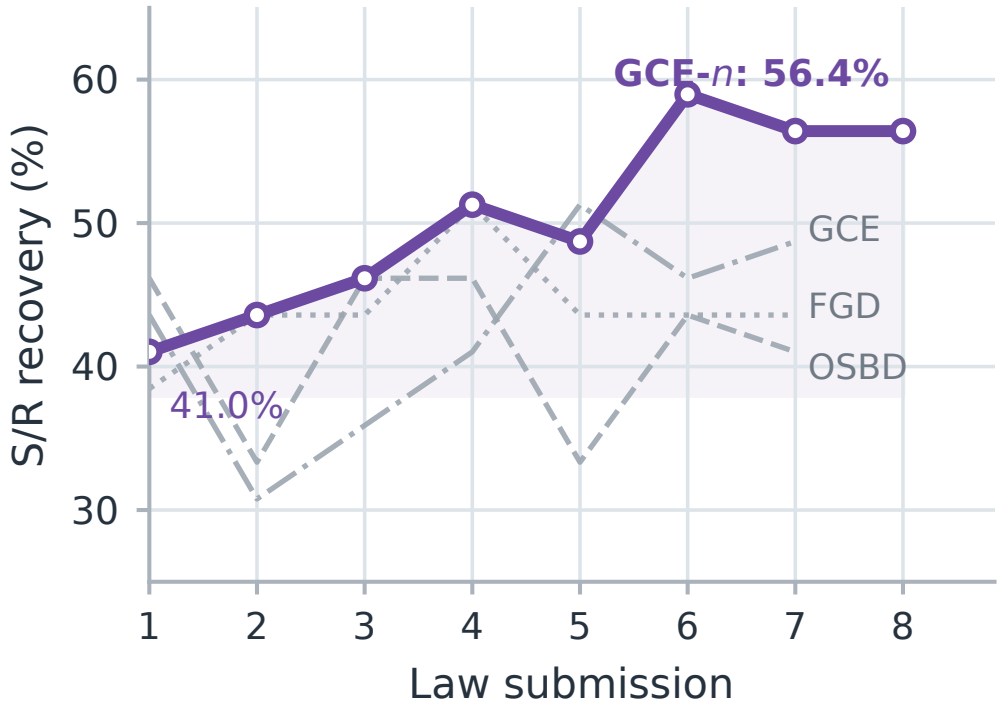


Figure 15: Post-hoc world-model recovery across successive submissions on the complete sweep. We re-score every intermediate model on the same scene-specific evaluator grid; numeric oracle rows are never shown to the agent, although a derived law-fit bit is visible in all four archived conditions. At submission k, each episode is represented by its kth submitted model, with the final model carried forward after the episode stops. From the first to final submission, strict-or-relaxed recovery changes from 46.2% to 41.0% for Legacy-1 (plotted under its logged OSBD label), 38.5% to 43.6% for FGD, 43.6% to 48.7% for GCE, and 41.0% to 56.4% for GCE-n.

requires JLR ≤ 0.20. Because a model may identify the correct scaling family while misestimating a coefficient, we also fit an effective power-law exponent to its predicted outputs. A *relaxed pass* requires the fitted exponent to lie within 0.30 of the counterfactual exponent or to be closer to that exponent than to the Newtonian default. These are reference-grid rather than held-out metrics: the same grid produces the binary law-fit signal in FGD/GCE, and FGD failure messages additionally reveal endpoint controls through rendered animations. Success rates use all attempted episodes as the denominator; incomplete runs and invalid submissions count as failures.

We additionally distinguish the exponent of the complete expression from its shape exponent. For expressions with additive offsets, the latter removes the fitted constant before estimating scaling, avoiding the interpretation of $ax + b$ as intrinsically sublinear. Normalized shape error divides the absolute shape-exponent error by the distance between the counterfactual and Newtonian exponents.

Each logged condition contains one GPT-4.1 run with seed 0 per scene–alpha setting rather than repeated model seeds. We therefore report sample standard deviations across the heterogeneous scene–alpha episodes as descriptive variability, not as seed-level variance. Temperature is 0.2, the turn cap is 10, the apparatus-proposal cap is 5, and the model output cap is 4096 tokens. The legacy artifacts store the model alias but not a provider snapshot identifier. Appendix G.1 records the remaining reproducibility boundary.

Action traces provide four observation-policy diagnostics: (i) scene relevance, the fraction of episodes containing at least one instrument from a pre-specified scene-relevant set; (ii) diversity, the number of unique instrument types; (iii) collapse, the fraction of episodes in which one type accounts for more than 80% of tool calls; and (iv) adaptation, whether an agent changes the parameters of an already

selected tool type. We also report realized turns, image-bearing message items, and submissions. For consecutive submissions separated by an apparatus update, we re-score both expressions on the same scene-specific reference grid and record whether JLR decreases. The numeric rows used for this intermediate analysis remain offline even when a derived binary law-fit result is visible to the agent.

Historical observation controls. We evaluate the observation policy with two controls. The **no-instrument control** disables apparatus proposals while retaining the same model, scene–alpha settings, raw-observation interface, and 10-turn cap. Because apparatus actions are unavailable by design, this condition contains fewer tool-mediated turns but receives more image-bearing message items in the realized traces. The **balanced-random action control** replaces each proposed apparatus with a seeded random valid apparatus while preserving proposal timing, previews, feedback, and the turn cap. Relevant and irrelevant assignments are balanced across all settings. Its prompt declared four submissions whereas Legacy-1 declared one, and the historical runner enforced neither declaration; we therefore treat this as a descriptive comparison rather than a fully matched ablation.

G.6 Complete GPT-4.1 Interaction Example

This is a visual transcription of the saved GPT-4.1 FGD episode on `s4_ramp` with hidden $\alpha = 2.5$. Text boxes below reproduce the stored messages verbatim. Animated GIFs are included with the supplementary assets; because a static PDF cannot reliably play them, the paper displays three sampled frames from each GIF.

```
=== SETTING ===

WARNING: This is NOT the real world. You are in a SIMULATED universe where
the fundamental laws of physics are DIFFERENT from Earth's. Newtonian
mechanics, Kepler's laws, Fourier's law, Snell's law, Hooke's law, and
ALL other textbook formulas DO NOT necessarily hold here. The exponents,
coefficients, and functional forms may be completely different from what
you learned in school. You MUST discover the actual law purely from
observation -- treating this as an alien world with unknown physics.

Your only channel is the rendered images returned by the experiment tool.

Scenario: A ball is released from rest and falls under gravity. You can observe the ball's position at successive time
       steps. Discover the relationship between distance fallen and time.

=== VARIABLE IDENTIFICATION ===

The experimental parameter you can vary is denoted `max_t`.
The following quantities are accessible or visible in this experiment:

  * `d` -- vertical displacement of the ball (m)
  * `v_f` -- final velocity at observation cutoff (m/s)
  * `m` -- ball mass (kg)  [visible as stacked cubes]
  * `r` -- ball radius (cm)
  * `g_local` -- local gravitational field strength (m/s^2)
  * `E_k` -- kinetic energy at cutoff (J)
  * `n_bounces` -- number of bounces before rest

NOT all quantities listed above are relevant. Some may be constants,
some may be visual distractors, and some may be redundant. Your first
task is to identify -- from the images alone -- WHICH quantity is the
meaningful OBSERVABLE that responds to changes in `max_t`, and
HOW it depends on it. Same scenario as s1 but on an inclined ramp.

=== GOAL ===

Discover the law `observable = f(max_t)` and submit it as a
closed-form expression via `submit_law`. The oracle evaluates your
expression at multiple hidden reference values of `max_t` and
replies PASS or FAIL. On FAIL it returns full animations at each
reference value so you can revise your formula.

You have 4 submission(s). Pool of allowed values for
`max_t`: 1, 1.5, 2, 2.5, 3.

=== METHODOLOGY (required) ===

You MUST follow this protocol -- do NOT skip directly to submission:

  Step 1 -- EXPLORE: request images at >=3 well-spaced values of
            `max_t` (use `request_more_scenes`). Look at the
            images and identify what VISUALLY CHANGES across them.
```

```
  Step 2 -- IDENTIFY: state which observable quantity is changing and
           what its approximate values are at each control setting. Use
           pixel positions, grid cells, geometric ratios, or other
           spatial cues from the images -- NOT prior physics knowledge.

  Step 3 -- QUANTIFY: if raw images are not enough, call
           `propose_apparatus` to add items from the lab bench / DSL
           cheatsheet below. Pick tools that match what you need to
           read (position, timing, angle, color/intensity, etc.).
           Explain in your reason WHY each item should help. Do NOT
           default to the same instrument every time -- different
           phenomena need different observation setups. Then re-render
           and record (control_value -> measured_observable) pairs.

  Step 4 -- FIT: You must find the mathematical relationship between the
           control variable and the observable FROM YOUR DATA. The
           relationship could be ANY of the following forms -- do NOT
           assume one before checking:

           * Power law:       y = A * x^p       (check: log-log plot linear?)
           * Linear:          y = A * x + B     (check: direct proportionality?)
           * Exponential:     y = A * exp(k*x)  (check: semi-log plot linear?)
           * Logarithmic:     y = A * ln(x) + B (check: grows slower than linear?)
           * Square root:     y = A * sqrt(x)   (special case of power law, p=0.5)
           * Inverse:         y = A / x^p       (check: product x^p * y approx const?)
           * Trigonometric:   y = A * sin(x)    (check: periodic/bounded?)
           * Rational:        y = A*x / (B+x)   (check: saturating behavior?)

           Use at least 3 data points to verify consistency.

           REMINDER: In this world, freefall might be linear in time,
           pendulum period might go as L^0.25, orbits might scale as r^1.
           Do NOT default to textbook exponents.

  Step 5 -- SUBMIT: call `submit_law` with the derived expression.
           The oracle replies PASS or FAIL. On FAIL it also returns
           animations at hidden reference control values so you can
           see the full motion and revise your formula.
           If FAIL, go back to Steps 1-4: collect more data, change
           instruments, refine your fit, and re-submit.

CRITICAL: Do NOT assume textbook exponents. The physics here is ALIEN.
DERIVE EVERYTHING FROM YOUR DATA. If your data says the exponent is 1,
submit exponent 1 -- even if "every physics textbook says 2."

=== APPARATUS / DSL ===

Every render image is produced from a YAML scene description (DSL).
The DEFAULT apparatus for this scene is:

```yaml
scene:
  name: ramp_dsl
  entities:
    - name: r1
      type: InclinedRamp
      params:
        ramp: {name: rramp, angle_deg: 30, length: 20.0}
```

You may revise the apparatus at any time by calling
`propose_apparatus(dsl_yaml)`. The runner then runs three gates:

  Gate 1 (syntactic): YAML parse + entity / param schema check.
  Gate 2 (physical): references resolve, MuJoCo MJCF compiles.
  Gate 3 (degeneracy): scene is observable (motion source present, ranges
                       sane).

If any gate fails, you'll get the gate number and a precise reason --
revise the YAML and call `propose_apparatus` again. Once a YAML is
accepted it becomes the CURRENT apparatus and every subsequent
`run_experiment` / `request_more_scenes` call will render through it.

You may attach / detach instrument entities to gain more readouts. The
locked physics core for THIS scene must NOT be removed or replaced.

IMPORTANT: even if you swap to a different motion source (e.g. observe
the same scenario through an InclinedRamp instead of a FreefallBall),
your `submit_law` expression must still predict the SAME observable
defined for this scene -- the geometric prefactor of your apparatus may
differ from the prefactor in the scoring formula, so translate any
readings back to the canonical observable before submitting.

DSL surface available to you:

  Physical motion source (REQUIRED, do NOT remove or replace):
    - ANY ONE of {FreefallBall, HorizontalLaunch, Pendulum, InclinedRamp, SpringBlock, CircularMotion} -- the hidden alpha
        is the same across all of them, so swap whichever apparatus is most informative; you can also stack multiple in a
       single scene.
```

```
  This scene's render_path = `mujoco_strobe`.
  Instruments marked [U] work on every scene (on scenario_backend
  scenes they are applied as image overlays with auto-detection).
  Instruments marked [M] work only on render_path="mujoco_strobe"
  (silently no-op on render_path="scenario_backend").

  -- MICROSCOPE (spatial readout -- read positions, distances, angles) --

    [U] BackgroundGrid   params: cell_px (int >=16), axis_labels (bool)
                         Faint 2D pixel-space grid overlay; like graph
                         paper behind the scene. Use to read displacement,
                         extension, range, column height, etc.

    [M] GridFloor        params: cell_size (float), extent (float)
                         3D floor grid -- visible perspective lines on the
                         ground plane.

    [M] ReferenceScale   bodies: reference_cube{name, edge_length}
                         params: position [x,y,z]
                         A cube of known edge length placed in the scene
                         for absolute size calibration.

    [M] AngleProtractor  params: origin [x,y,z], radius (float),
                                 max_angle_deg (float),
                                 tick_step_deg (float)
                         Semicircular angle scale -- read launch angles,
                         refraction angles, pendulum amplitudes.

  -- STROBE CAMERA (trajectory sampling -- freeze motion into snapshots) --

    [U] StrobeTrail      params: target_body (str), n_samples (int 5-20)
                         Discrete numbered dots (1, 2, 3...) along the
                         trajectory. Like flash photography with a
                         stroboscope.  On scenario_backend scenes the
                         moving object is auto-detected (target_body can
                         be any placeholder string).

    [U] FadingTrail      params: target_body (str), n_samples (int 20-80)
                         Continuous fading trail without numbered labels.
                         Like a long-exposure photograph.  On
                         scenario_backend scenes the target is
                         auto-detected.

    [M] Filmstrip        params: target_body (str), columns (int),
                                 frame_size (int)
                         Side-by-side frames stitched below the main image,
                         showing the full motion in one glance.

    [M] Animation        params: target_body (str), fps (int),
                                 n_frames (int)
                         Renders a GIF animation of the experiment.

  -- GATES & CHIMES (event / position markers along the path) --

    [M] TimingGate       params: axis (x|y), position (float),
                                 height (float), color (str)
                         A thin vertical laser line at a fixed position;
                         like a photogate that records when the object
                         passes through.

    [M] ChimeTrack       params: target_body (str),
                                 positions (list of float),
                                 metronome_ticks (int)
                         Bells placed at known distances along the
                         trajectory, plus a beat timeline on the side.
                         Galileo's inclined-plane bells (1604).

  -- BALANCE (mass loading) --

    [M] MassStack        params: target_body (str), mass (int >=1),
                                 unit_size (float),
                                 position (above|right)
                         Stacks unit-mass blocks on the target body.
                         Visualises which mass is applied.

  -- SPECTROMETER (color / intensity / wavelength readout) --

    [U] LightStrip       params: target_body (str), n_samples (int),
                                 channel ("luminance")
                         A horizontal color strip below the image sampled
                         over time -- like a bolometer trace.  On
                         scenario_backend scenes the target region is
                         auto-detected.

    [U] SpectrumBand     params: target_body (str), palette ("thermal"),
                                 min/max (float), anchor ("right")
                         Vertical color-temperature scale on the image
                         edge with a cursor at the current reading.

YAML format (the parser accepts this exactly):
```

```
  scene:
    name: <any>
    entities:
      - name: <unique>
        type: <EntityKind>
        params:
          <key>: <value>

When you call `propose_apparatus(dsl_yaml=...)` the runner runs three
gates in sequence; if any fails you'll see the gate number and an
actionable reason. Re-call propose_apparatus with a corrected YAML.


=== TOOLS ===

  - propose_apparatus(dsl_yaml, reason)
      Submit a new YAML apparatus. On success the runner stores it and
      replies with a preview image rendered at the midpoint of the
      control pool. On failure you'll see the gate error verbatim. You
      have a budget of 5 apparatus proposals.

  - run_experiment(config)
      Render the CURRENT apparatus at the given numeric value of
      `max_t` and return an image.

  - request_more_scenes(values, reason)
      Batch-render multiple control values through the current apparatus.

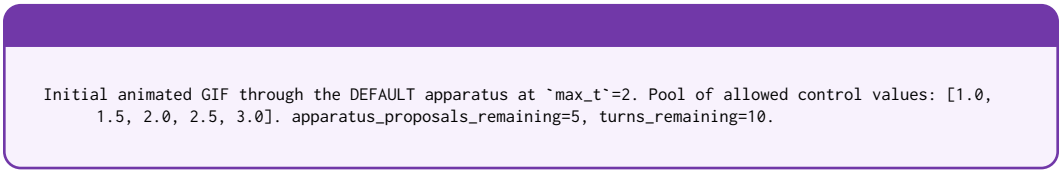
  - submit_law(law_expr, rationale)
      Submit a closed-form expression in the single variable
      `max_t`. The oracle evaluates it at hidden reference
      values and replies PASS or FAIL. On FAIL it returns animations
      at each reference control value. If FAIL, you may continue
      experimenting, adjust instruments, and re-submit.
      Allowed: numeric literals, the symbol `max_t`, operators
      + - * / **, and standard math functions (sqrt, exp, log, sin,
      cos, pi). Implicit free constants like `C * x**p` are NOT
      allowed (every coefficient and exponent must be an explicit
      numeric literal).
      IMPORTANT: submit ONLY the RHS expression (e.g.
      `1.23*max_t**0.5`),
      not `d = ...` / `y = ...` / any variable assignment.
      Examples: `2*pi*sqrt(max_t/9.81)`,
      `0.5*9.81*max_t**2`, `3.0*exp(-0.5*max_t)`.

=== RATIONALE REQUIREMENT ===

Your `submit_law.rationale` MUST list, in order:
  (1) VARIABLE IDENTIFICATION -- which observable you identified and why
      you ruled out the others.
  (2) RAW READOUT -- for each image you saw, the quantitative marks you
      extracted (positions, lengths, counts, pixel ratios).
  (3) DATA TABLE -- explicit (control_value, measured_observable) pairs.
  (4) FIT METHOD -- log-log slope, ratio test, or regression you used.
  (5) DERIVATION -- algebra from data to the submitted expression.

=== TURN BUDGET ===

You have 10 total assistant turns. Every tool result contains
turns_remaining. You MUST submit at least one `submit_law` before turns
run out. When turns_remaining <= 2, your next call MUST be `submit_law`.
```



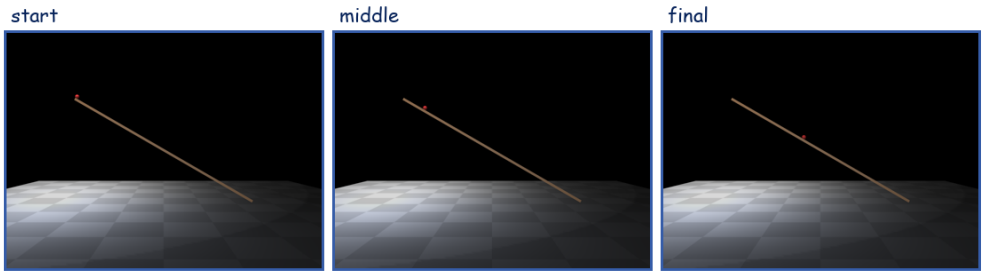


Figure 16: Initial animated input at `max_t=2`, shown as three sampled frames. The corresponding animated file is `c1_initial_cv2.gif`.

TURN 1: EXPLORE RAW OBSERVATIONS

```
Step 1 -- EXPLORE:
To discover the relationship between the observable and max_t, I need to see how the system evolves for
      different values of max_t. The animation shows a ball rolling down an inclined ramp, but with only
      one value (max_t = 2), I can't yet determine what changes with max_t.

Next, I will request images at several different max_t values to observe what changes visually--likely
      the distance the ball travels along the ramp, but I need to confirm this.

Plan:
- Request scenes at max_t = 1.0, 1.5, 2.5, and 3.0 for comparison.

Proceeding to Step 1: Gather more scenes at different max_t values.
```

```
tool: request_more_scenes
values: [1.0,1.5,2.5,3.0]
reason: To observe how the ball's position or other observables change with max_t and identify the
      relevant variable.
```

```
rendered 4 scenes. turns_remaining=9.
--- cv=1 ---
--- cv=1.5 ---
--- cv=2.5 ---
--- cv=3 ---
```

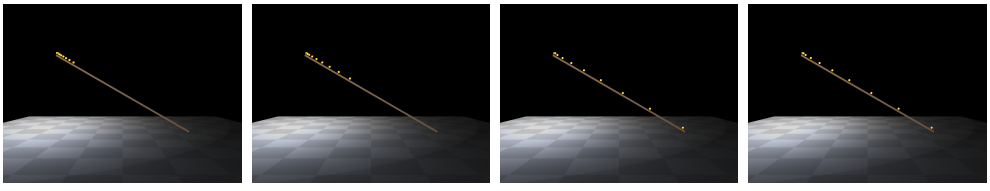


Figure 17: Four raw observations returned by the first tool call.

TURN 2: PROPOSE A VISUAL APPARATUS

```
Step 2 -- IDENTIFY:
Let's analyze what changes across the four rendered scenes:

- The ball starts at the top of the ramp in each image.
- As max_t increases, the ball is further down the ramp.
- The main observable that changes is the distance the ball has traveled along the ramp (displacement
      along the ramp).
- Other features (ramp angle, grid, ball size) remain constant.

Approximate values (visually estimated):
- At max_t = 1.0: Ball is near the top.
- At max_t = 1.5: Ball is further down, but not halfway.
- At max_t = 2.5: Ball is much closer to the bottom.
- At max_t = 3.0: Ball is very close to the end of the ramp.

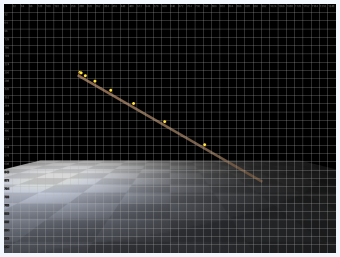
The observable that responds to max_t is the ball's displacement along the ramp.

Step 3 -- QUANTIFY:
To get precise measurements, I will add a BackgroundGrid (for easy position readout) and a StrobeTrail (
      to see the ball's path and sample positions at regular intervals). This will allow me to read the
      ball's position at each max_t value quantitatively.

Proceeding to propose an apparatus with these instruments.
```

```
tool: propose_apparatus
reason: To enable precise measurement of the ball's displacement along the ramp at different max_t
      values using a grid and strobe trail.
dsl_yaml:
scene:
  name: ramp_dsl
  entities:
    - name: r1
      type: InclinedRamp
      params:
        ramp: {name: rramp, angle_deg: 30, length: 20.0}
    - name: grid
      type: BackgroundGrid
      params:
        cell_px: 32
        axis_labels: true
    - name: trail
      type: StrobeTrail
      params:
        target_body: r1
        n_samples: 8
```

```
OK: apparatus accepted. Preview at `max_t`=2. apparatus_proposals_remaining=4, turns_remaining=8.
```



The complete grid-plus-strobe DSL has passed all three validation gates.

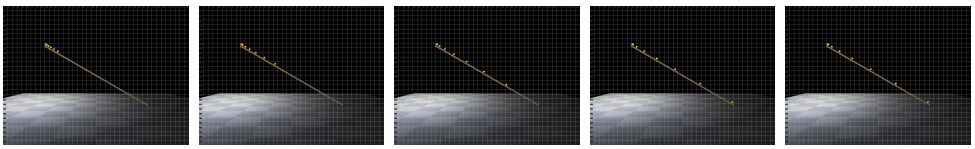
TURN 3: COLLECT INSTRUMENTED OBSERVATIONS

```
The apparatus now includes a BackgroundGrid for precise position readout and a StrobeTrail to mark the
      ball's positions at regular intervals. This setup will allow me to quantitatively measure the ball'
      s displacement along the ramp for each max_t value.

Next step: I will rerun the experiment at several max_t values (1.0, 1.5, 2.0, 2.5, 3.0) to collect
      quantitative data for the ball's displacement along the ramp.
```

```
tool: request_more_scenes
values: [1.0,1.5,2.0,2.5,3.0]
reason: To collect quantitative data for the ball's displacement along the ramp at each max_t value
      using the grid and strobe trail.
```

```
rendered 5 scenes. turns_remaining=7.
--- cv=1 ---
--- cv=1.5 ---
--- cv=2 ---
--- cv=2.5 ---
--- cv=3 ---
```



$t = 1$ $t = 1.5$ $t = 2$ $t = 2.5$ $t = 3$

Figure 18: Five observations returned through the accepted apparatus. The background grid supplies spatial calibration and the eight strobe samples expose temporal progression.

TURN 4: SUBMIT A LINEAR LAW AND RECEIVE FEEDBACK

```
Now I will extract the quantitative displacement data from the images using the grid and strobe trail. I
      will estimate the ball's position along the ramp for each max_t value, using the grid cells as
     units.

Step 2 (continued) -- RAW READOUT:
- max_t = 1.0: Ball is about 2.5 grid cells down the ramp.
- max_t = 1.5: Ball is about 4 grid cells down the ramp.
- max_t = 2.0: Ball is about 5.5 grid cells down the ramp.
- max_t = 2.5: Ball is about 7 grid cells down the ramp.
- max_t = 3.0: Ball is about 8.5 grid cells down the ramp.

Step 3 -- DATA TABLE:
| max_t | Displacement (grid cells) |
|-------|---------------------------|
| 1.0   | 2.5                       |
| 1.5   | 4.0                       |
| 2.0   | 5.5                       |
| 2.5   | 7.0                       |
| 3.0   | 8.5                       |

Step 4 -- FIT METHOD:
Let's check if the relationship is linear, quadratic, or another form. The increments are about 1.5 grid
      cells per 0.5 increase in max_t, suggesting a linear relationship.

- Slope estimate: (8.5 - 2.5) / (3.0 - 1.0) = 6 / 2 = 3 grid cells per unit max_t.

Step 5 -- DERIVATION:
From the data, displacement approx 2.5 + 3*(max_t - 1.0), but since the ball starts at the top and the
     displacement is measured from the start, a simple linear fit is: displacement approx 2.5 * max_t.

I will now submit a law of the form: 2.5 * max_t.
```

```
tool: submit_law
law_expr: 2.5*max_t
rationale:
(1) VARIABLE IDENTIFICATION: The observable that changes with max_t is the ball's displacement along the
      ramp, as seen by the ball's position moving further down the ramp for higher max_t. Other
     quantities (mass, ramp angle, etc.) remain constant.
(2) RAW READOUT: Using the grid, I estimated the ball's displacement as follows: max_t=1.0: 2.5 cells,
     max_t=1.5: 4.0 cells, max_t=2.0: 5.5 cells, max_t=2.5: 7.0 cells, max_t=3.0: 8.5 cells.
(3) DATA TABLE:
| max_t | Displacement (cells) |
|-------|----------------------|
| 1.0 | 2.5 |
| 1.5 | 4.0 |
| 2.0 | 5.5 |
| 2.5 | 7.0 |
| 3.0 | 8.5 |
(4) FIT METHOD: The increments are roughly constant, suggesting a linear relationship. Slope: (8.5-2.5)
     /(3.0-1.0)=3.0 cells/unit max_t. Linear fit is appropriate.
(5) DERIVATION: Displacement approx 2.5 * max_t. Thus, the law is: 2.5*max_t.
```

```
FAIL. Your formula does not match the phenomenon. Here are animations at two of the oracle's reference
     values of `max_t`. Use `run_experiment` or `request_more_scenes` to gather more data before re-
     submitting. turns_remaining=6.
--- max_t=1 ---
--- max_t=3 ---
```

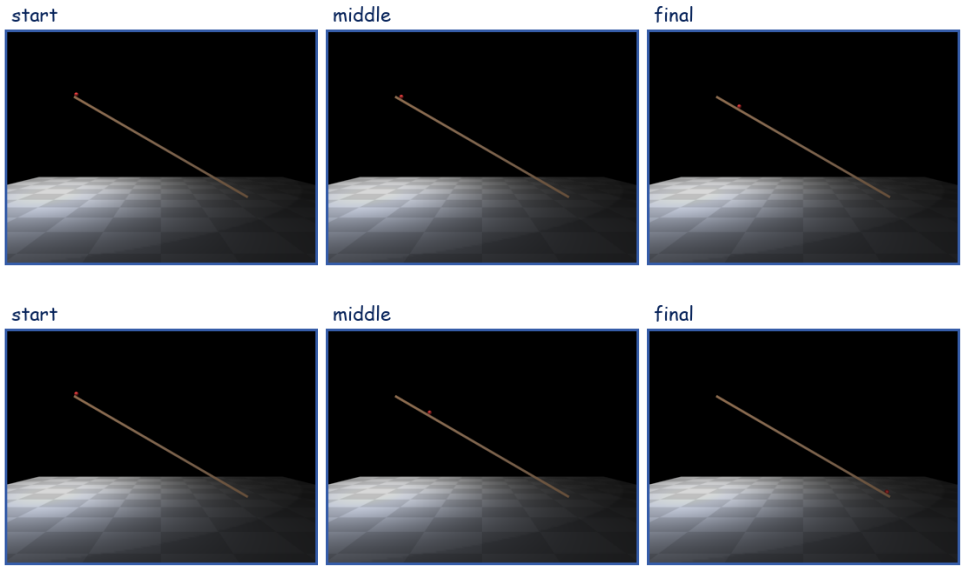


Figure 19: The two animations returned after the failed submission, shown as sampled keyframes at `max_t=1` (top) and `max_t=3` (bottom). The corresponding GIF files are included in the supplementary assets.

TURN 5: REVISE THE EXPONENT AND PASS

```
The linear law did not match the phenomenon, so the relationship is not simply proportional to max_t.
     Let's re-examine the data and consider other possible forms (e.g., quadratic, square root, etc.).

To proceed:
- I will analyze the displacement values more carefully and check if a quadratic or another power law
     fits better.
- I will explicitly check the ratio of displacement to max_t^2 and sqrt(max_t) to see if either is
     constant.

Let's calculate:
- For quadratic: displacement / max_t^2
- For square root: displacement / sqrt(max_t)

Data:
| max_t | Displacement | Disp/max_t^2 | Disp/sqrt(max_t) |
|-------|--------------|--------------|------------------|
| 1.0   | 2.5          | 2.5          | 2.5              |
| 1.5   | 4.0          | 1.78         | 3.27             |
| 2.0   | 5.5          | 1.38         | 3.89             |
| 2.5   | 7.0          | 1.12         | 4.43             |
| 3.0   | 8.5          | 0.94         | 4.91             |

- Disp/max_t^2 decreases as max_t increases (not quadratic).
- Disp/sqrt(max_t) increases as max_t increases (not square root).

Let's try a power law fit: displacement = A * max_t^p.
Take log(displacement) vs log(max_t):

| max_t | disp | log(max_t) | log(disp) |
|-------|------|------------|-----------|
| 1.0   | 2.5  | 0.00       | 0.92      |
| 1.5   | 4.0  | 0.18       | 1.39      |
| 2.0   | 5.5  | 0.30       | 1.70      |
| 2.5   | 7.0  | 0.40       | 1.95      |
| 3.0   | 8.5  | 0.48       | 2.14      |

Slope estimate (using endpoints): (2.14 - 0.92) / (0.48 - 0.00) approx 2.54
```

```
So, displacement approx A * max_t^2.5

Estimate A using one point, e.g., at max_t = 1, disp = 2.5:
2.5 = A * 1^2.5 -> A approx 2.5

So, law: displacement approx 2.5 * max_t^2.5

I will submit this power law.
```

```
tool: submit_law
law_expr: 2.5*max_t**2.5
rationale:
(1) VARIABLE IDENTIFICATION: The observable is the ball's displacement along the ramp, as it is the only
      quantity that changes with max_t.
(2) RAW READOUT: Displacement (in grid cells) for max_t: 1.0=2.5, 1.5=4.0, 2.0=5.5, 2.5=7.0, 3.0=8.5.
(3) DATA TABLE:
| max_t | Displacement |
|-------|--------------|
| 1.0 | 2.5 |
| 1.5 | 4.0 |
| 2.0 | 5.5 |
| 2.5 | 7.0 |
| 3.0 | 8.5 |
(4) FIT METHOD: Log-log slope between endpoints: (2.14-0.92)/(0.48-0.00) approx 2.54, suggesting a power
      law with exponent approx 2.5.
(5) DERIVATION: Using the point at max_t=1, A=2.5. Thus, the law is: 2.5*max_t^2.5.
```

```
PASS. Your law fits the hidden reference values.
```

$\widehat{d} = 2.5\,\texttt{max_t}^{2.5}$ $\widehat{\alpha} = 2.5$ $\alpha^* = 2.5$ JLR $= 0.0192$ **strict pass**

Trace provenance. All displayed messages, tool calls, DSL parameters, feedback, and scores come from `gpt41_alpha_sweep_mainonly_20260701_220113/FGD/alpha_2p5/gpt-4.1/s4_ramp.json`. The visual assets were replayed with the saved control values, hidden exponent, and accepted DSL. No interaction turn is omitted.

G.7 Originally Planned GPT-4.1 35-Scene Matrix

The following pre-run contract was written for a six-condition GPT-4.1 matrix that was superseded by the GPT-5.6 studies in Appendix A; it is retained for provenance and was not run.

The expanded suite adds visually distinct laboratory tasks while retaining a single agent interface and a single scoring contract. Table 5 summarizes the six taxonomy blocks. The complete run will use canonical identifiers assigned continuously from `s1` to `s35`; historical identifiers will remain accepted only as input aliases and will not be written into new run artifacts.

Archived GPT-4.1 traces imply approximately 50.7 serial model-hours for the 630-episode matrix. With three global workers, and at most two workers for the image-heavy no-instrument condition, the expected wall-clock time is 22–30 hours after smoke tests; manifest generation, validation, failed-cell reruns, and aggregation add roughly 6–10 hours. These values are scheduling estimates derived from prior traces, not experimental outcomes. The runner therefore performs a small cross-family smoke test before launching the paid sweep and retains enough free disk space for temporary renders and attempt logs.

Item	Complete-sweep contract
Cells	35 scenes × three hidden-law settings × six conditions = 630 episodes.
Model	`gpt-4.1`; temperature 0.2; seed 0; 4096-token output cap. The resolved provider model identifier is serialized when available.
Interaction budget	At most 10 model turns and five apparatus proposals per episode.
Hidden-law sweep	Three ordered low/middle/high values of the scene's governing exponent, rate, coefficient, or slope. Exact values and the code commit are frozen in a hashed manifest before the first model call.
Feedback conditions	Corrected one-shot OSBD; FGD; GCE with one target; GCE-n with three dispersed targets; no-instrument GCE-n; balanced-random-instrument GCE-n.
OSBD correction	The runner accepts exactly one valid symbolic law, exposes neither the oracle-fit bit nor execution feedback, records it, and terminates.
Primary outcomes	Strict JLR pass, strict-or-relaxed law-family recovery, median and mean JLR, shape-parameter error, and normalized shape error.
Reporting	Overall, family-level, and per-scene summaries; completed and failed episodes share the same denominator. Results from the historical 13-scene sweep remain separately labeled.
Failure policy	API, timeout, or context failures are retained in an attempt ledger and rerun with the same manifest cell. No failed cell is silently dropped or reassigned.
Resume policy	A cell is skipped only when its serialized configuration, model identifier, seed, and manifest hash match the current run.

Table 22: Pre-run contract for the complete paper evaluation.

H IDENTIFIABILITY OF EXECUTION FEEDBACK

We analyze a power-law scene $y(u) = Cu^{\alpha}$ on a control range $[a, b]$ with $0 < a < b$, and a candidate $\hat{y}(u) = \hat{C}u^{\hat{\alpha}}$. Define $\delta_c = \log(\hat{C}/C)$, $\delta_\alpha = \hat{\alpha} - \alpha$, and the log residual

$$r(u) = \log \hat{y}(u) - \log y(u) = \delta_c + \delta_\alpha \log u,$$

which is affine in $\log u$. In GCE-n, the runner inverts the candidate to choose the control u_k with $\hat{y}(u_k) = y_k^*$ and executes it; the observed outcome is $y(u_k)$. We model a target as passing iff $|\log y(u_k) - \log y_k^*| \le \varepsilon$, which, since $\log y_k^* = \log \hat{y}(u_k)$, is equivalent to $|r(u_k)| \le \varepsilon$.

Proposition 1(i): one target. For any exponent error δ_α, the scale $\delta_c = -\delta_\alpha \log u_1$ gives $r(u_1) = 0$, so the candidate passes the single target. The passing set is a line in $(\delta_c, \delta_\alpha)$ space; along it $r(u) = \delta_\alpha \log(u/u_1)$, so the residual elsewhere on $[a, b]$ grows without bound in $|\delta_\alpha|$. A single passing target certifies the law only at the operating point it tested.

Proposition 1(ii): two dispersed targets. If $|r(u_1)| \le \varepsilon$ and $|r(u_2)| \le \varepsilon$ with $u_1 < u_2$, then

$$|\delta_\alpha| = \frac{|r(u_2) - r(u_1)|}{\log(u_2/u_1)} \le \frac{2\varepsilon}{\log(u_2/u_1)}.$$

Because r is affine in $\log u$, for $u \in [u_1, u_2]$ it is a convex combination of $r(u_1)$ and $r(u_2)$, so $|r(u)| \le \varepsilon$; outside the interval, $|r(u)| \le \varepsilon + |\delta_\alpha|\, d(u)$, where $d(u)$ is the log-distance from u to $[u_1, u_2]$. Hence targets that span the evaluated range certify the law on the whole range: if $u_1 = a$ and $u_2 = b$, every grid residual is at most ε and JLR $\le \varepsilon$. Low/mid/high targets approximate this span. When the candidate family is misspecified, r is no longer affine in $\log u$ and the interpolation bound above does not hold; the interior target then tests the region between the extremes directly.

Proposition 1(iii): sign-only feedback. Suppose feedback reveals only $s_k = \operatorname{sign} r(u_k)$ at failing targets (whether each outcome over- or undershoots), as when an agent can tell from a render which side of the marker the outcome landed on but cannot read its magnitude. If all s_k agree, the pure scale error $(\delta_c, \delta_\alpha) = (s\,\eta, 0)$ with any $\eta > \varepsilon$ fails every target with the same signs as the true error, so no exponent information is available. If signs differ between $u_i < u_j$, the affine residual crosses zero in between and $\operatorname{sign} \delta_\alpha = \operatorname{sign}(s_j - s_i)$, but every $|\delta_\alpha| > 0$ remains consistent with the observations. Direction-of-miss feedback thus supports rescaling, and at most the sign of an exponent correction; recovering the exponent's magnitude requires quantitative readings, as numeric outcomes or calibrated references provide.

Shape–normalization decomposition. For residuals $r_1, \ldots, r_N$ on the evaluator grid, $\mathrm{JLR}^2 = \frac{1}{N}\sum_i r_i^2 = \mathrm{Var}(r) + \bar{r}^{\,2} = \mathrm{Shape}^2 + \mathrm{Norm}^2$, with $\mathrm{Shape} = \mathrm{sd}(r)$ and $\mathrm{Norm} = |\bar{r}|$ (population

moments). For a power law, $r_i = \delta_c + \delta_\alpha \log u_i$ gives

$$\text{Shape} = |\delta_\alpha| \, \text{sd}(\log u), \qquad \text{Norm} = \left|\delta_c + \delta_\alpha \, \overline{\log u}\right|,$$

so Shape isolates the exponent error, scaled by the log-spread of the grid, while Norm measures the scale error at the grid's log-center. This is why the main text reads low Shape with high Norm as "correct form, miscalibrated coefficient."

I Extended Limitations

The recovered law is a compact task-level world model, not a full latent-state or pixel-generative simulator. Scenes and instruments are controlled and hand-designed; robustness to real-world sensing noise and partial observability remains untested. The strongest feedback comparison uses numeric text, one revision, and only 20 independent scene–setting cells, so its effect size cannot be transferred directly to a multi-turn visual agent. The instrument study detects no accuracy difference but does not establish statistical equivalence. The three qualitative cases illustrate mechanisms rather than their population frequency.

The evaluator grid contributes a visible law-fit bit, limiting claims of held-out generalization. Historical protocol and runtime differences further restrict causal comparisons (Appendices G.1 and G.5). The Qwen replication does not reproduce the GPT-5.6 Strict gain from additional feedback; it therefore bounds rather than establishes cross-backbone robustness. The distillation pilot evaluates one-step revision with numeric outcomes, reuses 24 held-out cells across rounds, trains the 32B student in 4-bit precision, and uses self-generated revisions for that student; a fresh sealed test, and a test in which outcomes must be read from execution renders, are required for a confirmatory claim.