PHYSEVOLVE: VERIFIER-GROUNDED SELF-EVOLUTION FOR PHYSICAL REASONING WITH EXECUTABLE WORLDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Physical reasoning underpins how embodied systems understand and act in the physical world, yet vision-language models (VLMs) remain unreliable on questions about dynamics and physical causality. Progress is critically constrained by the scarcity of grounded counterfactual training data, as observational videos capture only the factual trajectory that occurred. The self-evolving paradigm offers a route beyond fixed datasets by enabling models to construct their own training problems. Extending this paradigm to physics, however, requires an external source of truth that can both validate model-proposed environments and verify alternative physical outcomes under intervention. We introduce PhysEvolve, a verifier-grounded self-evolving framework that trains VLMs for causal physical reasoning from model-proposed executable worlds and simulation-authored tasks. At each round, a shared VLM proposes executable world programs and evidence schemas; a deterministic physics engine freezes each valid world, simulates matched baseline and body-elided counterfactual branches, and synthesizes discrete, answer-bound visual tasks. Exact verifier feedback from task solving updates both the shared policy via GRPO and an adaptive curriculum over physical intents. Starting from Qwen3-VL-8B-Instruct, PhysEvolve achieves targeted gains across PhysBench, CausalPhys, and ContPhy without human intervention data, improving PhysBench Throwing by 8.8 points and ContPhy Counterfactual Removal and Goal-Pit by 7.4 and 24.4 points. These gains concentrate on interaction-sensitive and altered-world questions, demonstrating zero-shot transfer of causal principles to unseen complex dynamics (cloth, fluid, soft bodies) and real-world video.

1 INTRODUCTION

Multimodal understanding of the physical world is foundational to embodied intelligence and robotic systems, where reliable decision-making requires anticipating how scenes evolve and respond to external interventions (Black et al., 2024; Zhou et al., 2026). Yet current vision-language models (VLMs) remain substantially less reliable on physical dynamics, intervention effects, and counterfactual outcomes than on static object recognition or spatial relations (Chow et al., 2025; Tang et al., 2026; Zheng et al., 2024). Closing this gap is fundamentally bottlenecked by training data: high-quality, verifiably annotated examples that isolate how physical outcomes change under controlled interventions are exceptionally scarce. Real-world videos are not only costly to annotate, but fundamentally capture only the factual trajectory that occurred, rendering counterfactual alternatives unobservable (Pearl, 2009).

A growing paradigm of self-evolving systems circumvents static data bottlenecks by enabling models to autonomously generate the problems from which they learn (Zhao et al., 2025; Xia et al., 2025; Gao et al., 2026). The viability of this paradigm hinges on an external source of truth that can independently validate self-generated problems and verify their correct answers as the training distribution shifts. In symbolic domains such as mathematics and code, an interpreter provides this exact supervision: a Proposer generates an executable specification, the interpreter computes its deterministic output, and exact correctness supplies the Solver's learning signal. Physics simulators offer a natural counterpart for the physical world by executing proposed environments and exposing their complete internal states (Prabhudesai et al., 2026). This analogy suggests an appealingly direct

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.

route to physical self-play: retain the proposer–solver curriculum and substitute a physics simulator for the symbolic interpreter.

However, naively mapping this paradigm to physical reasoning reveals a fundamental structural breakdown (Figure 1). When a shared Proposer–Solver policy proposes rigid-body scenes and evaluates a single simulated trajectory using continuous proximity scores, two failure modes emerge (Section 3.1): (1) single-trajectory readouts cannot construct or verify intervention tasks, starving the curriculum of causal supervision; and (2) continuous proximity rewards inflate learnability signals for uniformly incorrect rollouts, inducing reward hacking without improving exact physical correctness. These findings establish two indispensable requirements for physical self-play: the verification environment must actively execute interventions across paired branches, and derived visual tasks must be bound to discrete, exact ground truth.

We introduce PhysEvolve, a verifier-grounded self-evolving framework that learns causal physical reasoning from policy-proposed executable worlds and simulation-authored tasks (Figure 2). Rather than authoring text questions directly, a task-conditioned Proposer generates an executable *world program* and an *evidence schema*. A deterministic physics executor validates and freezes the scene, executes matched baseline and body-elided counterfactual branches under identical initial conditions, and derives an answer-bound question from their divergent outcomes. The shared VLM then solves the task from the baseline video, receiving exact discrete feedback. This architecture establishes a strict information boundary: the policy guides the exploration of physical environments, while the fixed simulator alone determines physical validity and factual/counterfactual ground truth. The resulting exact solver feedback simultaneously drives shared-policy updates via GRPO and guides an adaptive curriculum over physical reasoning intents.

Training on executor-synthesized tasks over policy-proposed worlds produces targeted transfer across three physical-reasoning benchmarks without any human-annotated intervention data. Starting from Qwen3-VL-8B-Instruct (Bai et al., 2025a), PhysEvolve improves PhysBench Throwing by 8.78 percentage points and Judgment by 5.08 points. Crucially, on ContPhy, performance on Counterfactual Removal and Goal-Pit increases by 7.38 and 24.44 points, transferring causal intervention principles zero-shot to held-out continuous media (cloth, fluid, rope, and soft bodies) entirely absent from the rigid-body training templates. On real-world videos in CausalPhys, PhysEvolve likewise achieves consistent gains on Anticipation ($+2.46$) and Intervention ($+2.27$).

Our core contributions are summarized as follows:

1. **Verifier-grounded physical self-play:** We present PhysEvolve, the first self-evolving framework that trains VLMs for causal physical reasoning via model-proposed executable worlds, enforcing a strict information boundary between policy exploration and deterministic simulator verification.
2. **Intervention-grounded task synthesis:** We introduce an execution pipeline that executes matched baseline and body-elided branches under identical seeds, transforming continuous multi-body dynamics into discrete, answer-bound multi-choice tasks with exact verifier rewards.
3. **Out-of-distribution causal transfer & audit:** We demonstrate that self-play on rigid-body programs transfers zero-shot to complex continuous dynamics and real-world video, and provide an empirical audit exposing why naive single-trajectory physical self-play fundamentally collapses.

2 Related Work

Physical reasoning in vision-language models. Extensive studies reveal that current multimodal models handle static visual appearance and geometric arrangements far more reliably than temporal prediction, physical intervention, and counterfactual reasoning (Riochet et al., 2018; Bear et al., 2021; Tung et al., 2023; Yi et al., 2020; Ates et al., 2022; Chow et al., 2025; Tang et al., 2026). To diagnose physical understanding beyond passive prediction, counterfactual benchmarks introduce paired simulator-generated alternative trajectories and intervention questions (Baradel et al., 2020; Chen et al., 2022; Patel et al., 2022). Recently, simulator-grounded training has emerged to convert simulation traces into verifiable supervision: notably, Sim2Reason synthesizes physics QA pairs from predefined simulator rollouts and filters shortcut solutions (Prabhudesai et al., 2026). Unlike offline trajectory-to-QA pipelines, PhysEvolve autonomously synthesizes verifiable counterfactual supervision *online* by executing paired intervention branches over active, policy-proposed physical worlds.

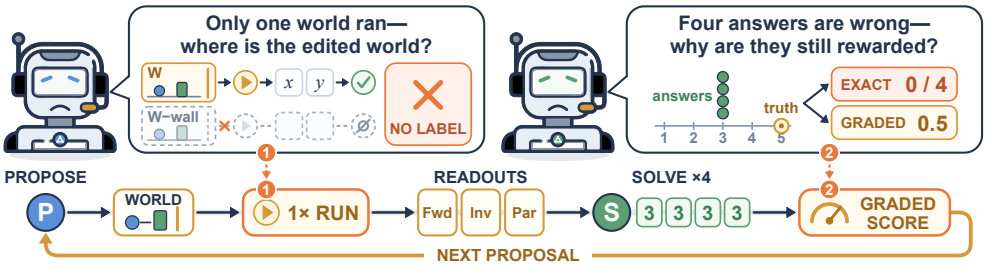


Figure 1: Naive self-play baseline for physical reasoning. A Proposer generates a world program; the executor simulates a single trajectory and derives Forward, Inverse, and Parameter tasks. Graded continuous proximity can reward a Solver that remains fundamentally incorrect, broadcasting distorted learnability signals back to the Proposer.

Self-evolving agents and environment co-design. While standard self-training bootstraps reasoning over fixed problem sets (Zelikman et al., 2022; Gülçehre et al., 2023), self-evolving agents generate the very curricula and environments from which they learn. Absolute Zero pioneered shared-policy proposer–solver self-play with code interpreters (Zhao et al., 2025), inspiring subsequent frameworks to expand task substrates across multi-agent debates, geometric spaces, games, and tool-augmented execution (Huang et al., 2025; Chen et al., 2025; Xia et al., 2025; Liu et al., 2025; Wei et al., 2025; He et al., 2025; Wang et al., 2025; Li et al., 2026). In parallel, unsupervised environment design (UED) co-evolves adaptive environments alongside control policies based on learner regret or progress (Wang et al., 2019; Dennis et al., 2020; Parker-Holder et al., 2022; Zala et al., 2024; Liang et al., 2024). While existing self-play frameworks operate primarily in discrete symbolic spaces, PhysEvolve shifts the frontier of self-evolution to continuous multi-body physics, delegating the physical environment proposal to the policy while retaining deterministic verification and task synthesis within the executor.

Reinforcement learning with verifiable rewards. Reinforcement learning with verifiable rewards (RLVR) substitutes brittle, exploitable learned reward models with deterministic, ground-truth verifiers in domains admitting objective computation (DeepSeek-AI, 2025; Lambert et al., 2024). Verification methods have expanded from exact answer matching and test-suite execution to hybrid dense-sparse rewards, self-consistency checks, and formal proof checking (Tao et al., 2026; Zhao et al., 2026; Wilf et al., 2026). While RLVR has achieved remarkable success in mathematical deduction and program synthesis, extending objective verification to multimodal physical reasoning has remained an open challenge. PhysEvolve realizes RLVR for multimodal physics by grounding policy updates in discrete, simulation-verified causal outcomes rather than heuristic continuous proximity metrics.

3 Methodology

We introduce PhysEvolve, a verifier-grounded self-evolving framework that learns causal physical reasoning by having a shared vision–language policy propose executable worlds while a deterministic physics executor authors and verifies derived multimodal tasks. Below, Section 3.1 first diagnoses the structural pitfalls of naively transferring code self-play to physical reasoning and establishes two necessary design principles. Sections 3.2–3.5 then detail the framework architecture, the task-conditioned proposer, the intervention-grounded task synthesis pipeline, and the hierarchical co-evolution training loop.

3.1 Problem Formulation and Pitfalls of Naive Physical Self-Play

In discrete symbolic domains such as mathematics and code synthesis, self-play enables models to surpass fixed data limits by autonomously generating and solving their own problems (Zhao et al., 2025). The policy evolves by optimizing a learnability reward that dynamically prioritizes tasks near the boundary of its current competence. The viability of this paradigm stems from its objectivity: a deterministic interpreter executes model-written specifications and returns exact correctness signals without human supervision.

Given that grounded, intervention-level annotations are exceptionally scarce in physical reasoning, transferring this self-play paradigm to multimodal physics by substituting a physics simulator for the symbolic interpreter appears to be a natural solution. In a direct physical transfer, a shared Proposer–Solver policy interacts with a rigid-body simulator. For each proposed world $\mathcal{W}$, the executor simulates one factual trajectory τ under seed ξ, extracts an observation x alongside a simulator-derived target y, and scores the Solver's prediction $\hat{y}$ via a graded similarity metric s:

$$\tau = \Phi_\xi(\mathcal{W}), \qquad (x, y) = \mathrm{readout}(\tau), \qquad r_{\mathrm{naive}} = s(\hat{y}, y). \tag{1}$$

Here Φ_ξ denotes deterministic simulation and s assigns partial credit based on continuous numerical or spatial proximity, intended to prevent sparse, uninformative all-zero rewards early in training. While this transfer ostensibly preserves both autonomous task generation and verifiable supervision, empirical auditing reveals two fatal structural failure modes.

Failure Mode 1: Single-trajectory execution cannot ground causal interventions. Physical reasoning distinctively demands understanding how scenes respond to counterfactual interventions—outcomes that never occurred in the observed trajectory (Pearl, 2009). However, evaluating a single trajectory τ restricts the task space strictly to passive observational readouts (e.g., predicting an object's future position or initial velocity). In our audit of the naive transfer baseline, exactly zero intervention or counterfactual tasks were synthesized across 100 training rounds, causing PhysBench dynamics accuracy to stagnate and reasoning performance to degrade (Section 4.4). This limitation is structural: causal ground truth cannot be inferred from a single trajectory without paired, counterfactual branch executions. Neither scaling model capacity nor expanding rollout volume can produce verified labels that lie beyond the verification interface itself.

Failure Mode 2: Graded proximity rewards induce learnability illusions. In code and math, execution verification yields a rigorous binary correctness signal. In continuous physics, however, substituting exact match with graded proximity s fundamentally distorts reinforcement signals. Because s scores relative closeness rather than true task resolution, clusters of uniformly incorrect rollouts (e.g., predicting stationary velocities or median parameters) yield intermediate rewards. These reward distributions artificially inflate the empirical learnability metric $r_{\mathrm{learn}} = 1 - 2|\bar{a} - 0.5|$, misleading the curriculum into classifying failed, trivial tasks as productive learning boundaries. Consequently, graded reward steadily increases during naive training while exact physical correctness remains near zero (Section 4.4, Table 4).

These failure modes establish that continuous physics cannot be treated as a trivial drop-in replacement for discrete code interpreters. Autonomous physical reasoning requires two non-negotiable architectural guarantees:

1. **Branch-level verification:** The executor must actively execute paired, counterfactual interventions (divergent simulation branches) under identical initial conditions to establish objective causal ground truth.
2. **Discrete answer binding:** Derived physical tasks must map continuous trajectories into discrete, answer-bound problems, ensuring that Solver success reflects exact correctness rather than deceptive proximity.

3.2 Framework Overview

Guided by these requirements, we introduce PhysEvolve (Figure 2). At each round, a *Task-Conditioned Proposer* transforms a sampled reasoning intent into an executable candidate world. Through *Intervention-Grounded Task Synthesis*, a fixed executor establishes factual and counterfactual outcomes via deterministic simulation and synthesizes discrete, answer-bound multiple-choice questions. Finally, *Co-Evolution Training* translates exact Solver correctness into joint updates for the shared policy and an adaptive curriculum.

Let π_θ denote the shared vision–language policy. The identical parameters serve as Proposer π_θ^{P} and Solver π_θ^{S}. The Proposer controls the exploration of executable physical environments, while the Solver receives only rendered baseline video observations and the resulting question and options. Crucially, a strict *information boundary* is enforced: the fixed physics executor alone accesses internal simulator states, computes counterfactual branches, and binds the correct answer. This boundary enables the policy to actively shape its own training distribution without ever setting its own training labels, eliminating label contamination and reward hacking.

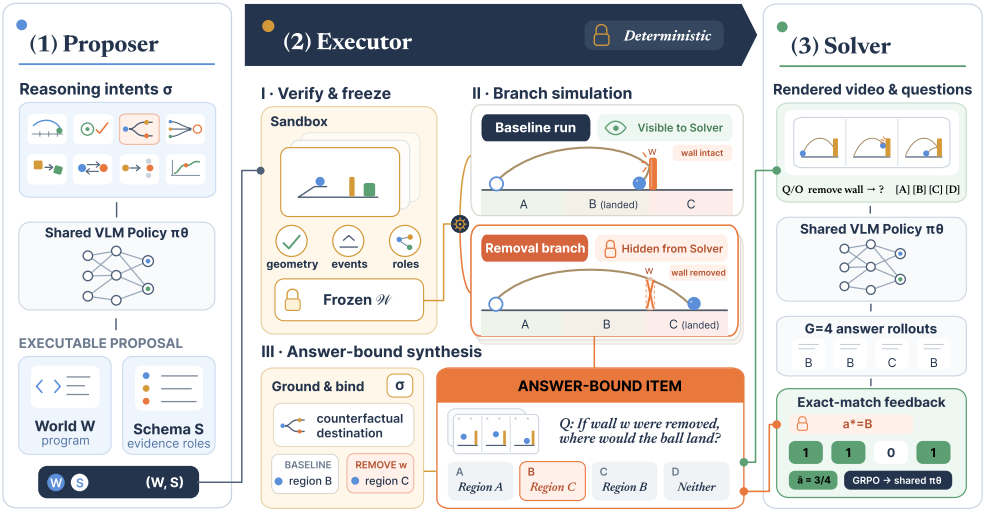


Figure 2: Overview of PhysEvolve. A shared VLM proposes an executable world program and evidence schema. A fixed deterministic executor validates and freezes the scene, simulates matched baseline and body-elided branches when required, and synthesizes an answer-bound item. Exact discrete Solver feedback updates the shared policy and the curriculum.

3.3 Task-Conditioned Proposer

The Proposer receives a reasoning intent σ, which specifies the target physical relation to probe and the evidence required to make that relation uniquely decidable. Grounded in classical cognitive physics (Battaglia et al., 2013; Spelke, 1990), we instantiate simulation across three fundamental Newtonian paradigms: (1) projectile motion and parabolic dynamics (*Throwing*), (2) impulse-momentum transfer and kinematic collisions (*Collision*), and (3) continuous kinematics and contact interactions (*Motion*), spanning eight active reasoning intents.

Conditioned on σ, the Proposer samples a tuple $(\mathcal{W}, \mathcal{S}) \sim \pi_\theta^{\mathrm{P}}(\cdot \mid \sigma)$: an executable *world program* $\mathcal{W}$ and an *evidence schema* $\mathcal{S}$. The world program is authored in PhysDSL (a domain-specific Python API over rigid-body simulation) specifying 3D geometry, material properties (mass, restitution, friction), initial states, kinematic constraints, and rendering viewpoints. The evidence schema $\mathcal{S}$ declares the semantic roles of key entities (e.g., query projectiles, barrier obstacles, potential collision targets) required by intent σ. The executor validates these role declarations against the realized simulation, rejecting any proposal whose physical dynamics cannot support the requested reasoning intent. Complete specifications of PhysDSL and evidence schemas are provided in Appendices A and A.2.

3.4 Intervention-Grounded Task Synthesis

A candidate pair $(\mathcal{W}, \mathcal{S})$ represents only a potential reasoning opportunity; simulation execution must establish its physical semantics before it can constitute a valid training task. The executor admits an item only when execution recovers unambiguous physical evidence that determines a unique answer. This contract is executed across three sequential stages:

World verification and freezing. The executor parses and executes the world program in a sandboxed simulator, resolving evidence-schema roles against simulated rigid bodies. It admits only worlds exhibiting valid geometry, non-trivial physical events, and the causal prerequisites of intent σ. A bounded deterministic repair stage corrects minor placement or boundary syntax violations without introducing new bodies or joints. Once admitted, the world definition is strictly *frozen* into an immutable scene definition. Freezing guarantees that all subsequent intervention branches share identical initial geometry, material properties, and simulation seeds, ensuring that divergent outcomes are strictly attributable to declared interventions.

Branch elision. For interventional reasoning intents, the executor constructs a finite set of body removals $\mathcal{B}_\sigma$. For each $B \in \mathcal{B}_\sigma$, it elides those bodies and their dependent constraints from the

frozen world, yielding the matched execution pair:

$$\big(\tau_\emptyset, \tau_B\big) = \big(\Phi_\xi(\mathcal{W}), \Phi_\xi(\mathcal{W} \setminus B)\big), \qquad B \in \mathcal{B}_\sigma. \tag{2}$$

Both trajectories share the identical simulator state, random seed ξ, and time horizon. The Solver observes only a video rendering of the factual trajectory $\tau_\emptyset$, while the executor retains the divergent branch trajectories $\{\tau_B\}$ as objective ground truth.

Answer-bound question synthesis. Each reasoning intent defines a discrete readout function, feasibility margins, and natural language templates. The executor applies the readout across baseline and branch trajectories. An item is suppressed if readouts are non-unique, infeasible, or fall within sensitive physical margin thresholds (e.g., velocity differences below 0.30 m/s). Otherwise, the executor determines the unique factual or counterfactual outcome, binds it to ground-truth option index $a^\star \in \{A, B, C, D\}$, and renders the final visual question and distractors. For inverse intervention tasks (e.g., "Which wall should be removed to make the ball land in the target pit?"), the executor runs all candidate elisions (remove A, remove B, remove both) and validates whether exactly one branch achieves the goal; "none of the above" is selected only if no branch reaches the target. Detailed contracts, margin tests, and binding rules are documented in Appendix C.

3.5 Co-Evolution Training and Adaptive Curriculum

The synthesized answer-bound items close the self-play loop by providing exact, discrete supervision to both policy roles.

Exact solver supervision. Each retained item defines a group of G Solver rollouts from policy π_θ^{S}. Given the executor-bound label $a^\star$ and predicted options $\{\hat{a}_i\}_{i=1}^G$, the binary Solver reward and empirical item success rate are:

$$r_i^{\mathrm{solve}} = \mathbf{1}[\hat{a}_i = a^\star], \qquad \bar{a} = \frac{1}{G}\sum_{i=1}^{G} r_i^{\mathrm{solve}}. \tag{3}$$

In contrast to the continuous proximity score s in Equation (1), r_i^{solve} provides unambiguous binary verification immune to proximity-induced reward hacking.

Hierarchical co-evolution. Co-evolution coordinates task selection and world exploration across two complementary levels:

- *Macro-level intent curriculum:* Across reasoning intents, the rolling Solver accuracy $\bar{a}_f$ for each task family f governs adaptive intent sampling. After an 8-round cold start, intent sampling weights follow $w_f = \rho_f \max(0.05, 4\bar{a}_f(1-\bar{a}_f))$, where ρ_f denotes structural family priors. This formulation concentrates training on intents situated at the frontier of the model's evolving competence (where $4\bar{a}_f(1-\bar{a}_f)$ peaks near $\bar{a}_f = 0.5$). A family-difficulty multiplier further modulates Proposer credit according to rolling intent challenge.
- *Micro-level world exploration:* Conditioned on the sampled intent σ, the Proposer explores the space of valid, physically rich world programs. Proposer rewards reward successful compilation, dynamic collision feasibility, and structural edit-distance diversity I, penalized by anti-copy and saturation controls (Appendix E).

Shared-policy GRPO optimization. Both Proposer and Solver roles are optimized jointly via Group Relative Policy Optimization (GRPO) on the shared parameters π_θ, regularized against a frozen reference policy π_{ref}. Proposer groups share a sampled intent σ, while Solver groups share an executor-rendered item. Joint optimization ensures that while the Solver masters causal physical deduction, the Proposer continuously adapts to author executable environments that remain informative and verifiable (Appendix F).

4 Experiments

4.1 Experimental Setup

Research questions. We design our empirical evaluation around four central questions: **(Q1) Targeted capability transfer:** Does PhysEvolve yield targeted improvements on the specific

physical capabilities it simulates during training? **(Q2) Out-of-distribution transfer:** Do causal principles learned in synthetic rigid-body environments generalize to unseen complex media (fluid, cloth, soft bodies) and real-world video? **(Q3) Architectural necessity:** How critically do paired branch simulation and joint Proposer–Solver optimization contribute to performance? **(Q4) Failure anatomy:** What exact supervisory signals are forfeited by relying on single-trajectory execution and continuous proximity rewards?

Training configuration. All experimental variants are initialized from the open-weight base model Qwen3-VL-8B-Instruct (Bai et al., 2025a). Training is executed in a single run for 100 steps using one-epoch GRPO with a learning rate of 10^{-6}. At each step, the curriculum samples $N = 4$ reasoning intents and generates $G = 4$ world program proposals per intent, capping retention at 32 active Solver groups with 4 rollouts per group. The complete hyperparameters and training loop are detailed in Appendix F and Algorithm 2.

Benchmarks and evaluation protocols. We evaluate across three rigorous physical reasoning benchmarks: (1) **PhysBench** (Chow et al., 2025) covers broad static and dynamic physical QA over 9,802 items; (2) **ContPhy** (Zheng et al., 2024) benchmarks counterfactual and goal-directed physical reasoning across 1,950 scenarios involving complex continuous dynamics (cloth, fluid, rope, and soft bodies); and (3) **CausalPhys** (Tang et al., 2026) assesses physical anticipation and intervention in 3,057 real-world video scenarios. All evaluated models share identical prompts, visual frame sampling, greedy decoding, and official evaluation protocols (Appendix F.3).

Baselines and comparative context. Our primary, strictly matched comparison comprises: (1) the base **Qwen3-VL-8B-Instruct** model; (2) **Naive self-play**, which replicates single-trajectory execution with continuous graded proximity rewards; and (3) **PhysEvolve**. To contextualize performance across model scales, we also report unaligned results for prominent open-weight models: InternVL2.5-8B (Chen et al., 2024), Llama-3.2-11B-Vision (Meta, 2024), Pixtral-12B (Agrawal et al., 2024), Qwen2.5-VL-7B (Bai et al., 2025b), and the scaled Qwen3-VL-32B (Bai et al., 2025a).

4.2 Main Results: Targeted and External Transfer

Table 1 presents overall and capability-specific accuracies across the three benchmarks, separating matched 8B configurations from open-weight context models.

Q1: Targeted capability transfer on PhysBench. Within the matched 8B evaluation, PhysEvolve achieves substantial gains on tasks directly aligned with simulated Newtonian paradigms, improving Throwing by 8.78 percentage points, Judgment by 5.08, and Motion by 1.74. In stark contrast, Naive self-play yields negligible or negative differences on these categories (-0.24 on Throwing, $+0.26$ on Judgment). Crucially, PhysEvolve improves overall accuracy ($+0.55$) without catastrophic forgetting of general visual capabilities: performance remains close to or above the base model across most non-target partitions, including static relations and object identification (Appendix G, Table 16).

Q2: Out-of-distribution transfer to complex media and real video. The most compelling evidence of physical generalization emerges on ContPhy: despite being trained exclusively on synthetic rigid-body simulations, PhysEvolve demonstrates remarkable zero-shot transfer to continuous media (cloth, fluid, rope, and soft bodies). On Counterfactual Removal, accuracy increases by 7.38 points over Base (51.64% vs. 44.26%), while on Goal-Pit, accuracy surges by 24.44 points (44.44% vs. 20.00%, surpassing even Qwen3-VL-32B's 26.7%). On CausalPhys real-world videos, PhysEvolve similarly yields consistent gains on Anticipation ($+2.46$) and Intervention ($+2.27$). Because the non-rigid dynamics and visual textures of ContPhy are completely absent from the training templates, these gains confirm that PhysEvolve internalizes an abstract *intervention operator*—the ability to deduce how physical states diverge under structural removal—rather than memorizing superficial simulation artifacts.

Fine-grained task transfer and checkpoint dynamics. Figure 3 breaks down transfer across fine-grained causal subcategories. Relative to the Base model, PhysEvolve improves Fluid Flow by 11.00 points and Action Substitution by 9.09 points, significantly outperforming the branch-disabled control (by 23.00 points on Fluid Flow). Furthermore, longitudinal checkpoint analysis (Figure 4,

Table 1: Accuracy (%) across the three physical reasoning benchmarks. The upper block provides contextual reference models; bold text indicates the best performance within the strictly matched 8B comparison block.

Model	PhysBench				CausalPhys			ContPhy		
	Ovr.	Throw.	Motion	Judg.	Ovr.	Antic.	Interv.	Ovr.	CF-Rm.	Goal-Pit
Contextual open-weight references										
InternVL2.5-8B (Chen et al., 2024)	43.8	40.0	27.7	32.7	50.2	51.5	42.4	31.4	48.4	23.3
Llama-3.2-11B-Vision (Meta, 2024)	38.7	36.6	27.2	30.8	44.2	44.1	40.1	20.7	0.0	20.0
Pixtral-12B (Agrawal et al., 2024)	46.9	41.2	35.2	31.0	50.2	53.0	44.5	21.5	40.2	13.3
Qwen2.5-VL-7B (Bai et al., 2025b)	49.4	44.4	50.6	33.1	52.6	56.5	45.7	25.9	11.5	24.4
Qwen3-VL-32B (Bai et al., 2025a)	60.9	47.1	83.5	44.3	58.9	61.9	52.9	41.0	45.9	26.7
Matched Qwen3-VL-8B comparison										
Qwen3-VL-8B (base) (Bai et al., 2025a)	52.9	46.1	54.7	34.9	53.9	55.9	42.8	37.1	44.3	20.0
Naive self-play	52.4	45.9	55.5	35.2	**55.8**	57.6	**45.9**	37.4	43.4	25.6
PhysEvolve (ours)	**53.5** (+0.6)	**54.9** (+8.8)	**56.5** (+1.7)	**40.0** (+5.1)	54.5 (+0.6)	**58.4** (+2.5)	45.0 (+2.3)	**39.5** (+2.5)	**51.6** (+7.4)	**44.4** (+24.4)

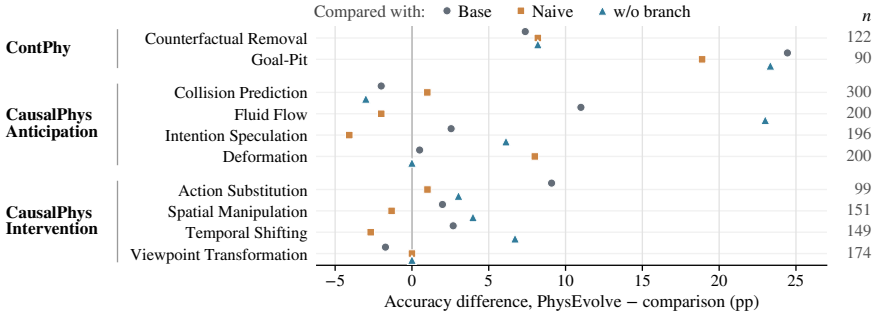


Figure 3: Task-level transfer across matched comparisons. Markers show accuracy differences between PhysEvolve and each comparison model on ContPhy (Counterfactual Removal, Goal-Pit) and CausalPhys subcategories. Positive values favor PhysEvolve. Sample counts appear at right; see Table 19 for complete accuracies.

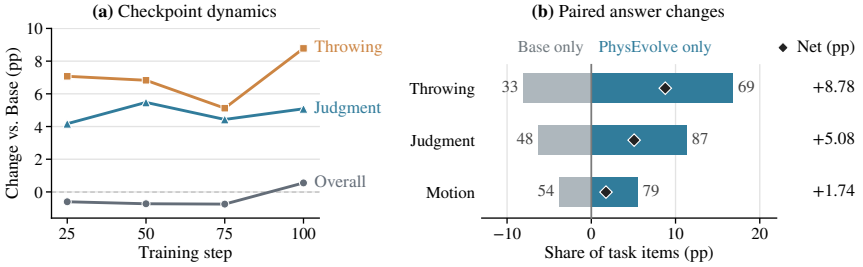


Figure 4: PhysBench training dynamics and paired correctness analysis. Left: accuracy gains over Base across scheduled training checkpoints. Right: item-level paired analysis at the terminal checkpoint, separating questions answered correctly only by PhysEvolve from those answered correctly only by Base. Diamonds indicate net correct item gains; see Table 20 for complete counts.

left) shows consistent gains over Base on Throwing and Judgment throughout self-play. At the terminal checkpoint (Figure 4, right), item-level paired auditing reveals that PhysEvolve correctly answers 69 Throwing items that Base misses against only 33 reversals (a net gain of 36 items), and net gains of 39 and 25 items on Judgment and Motion, confirming genuine competence acquisition.

4.3 Ablation Studies

Q3: Necessity of branch-enabled synthesis and joint co-evolution. We investigate the architectural components of PhysEvolve via two targeted endpoint ablations:

Essentiality of paired branch execution. In Table 2, removing branch simulation and its dependent Counterfactual/Inverse task families sharply degrades downstream causal transfer. The performance drop is especially drastic on ContPhy Counterfactual Removal (-8.20 points) and Goal-Pit (-23.33

Table 2: Ablation of the complete branch-enabled training path. Evaluating endpoint accuracy (%) under identical protocols when branch execution and dependent Counterfactual/Inverse families are deactivated.

	Training Path		Realized Exposure		CausalPhys		ContPhy		
Configuration	Branch Sim.	CF/Inv	Groups	Rollouts	Ovr.	Interv.	Ovr.	CF-Rm.	Goal-Pit
PhysEvolve	On	On	2,167	8,668	**54.50**	**45.03**	**39.54**	**51.64**	**44.44**
w/o branch-enabled path	Off	Off	1,990	7,960	51.42	41.71	37.28	43.44	21.11
Difference	–	–	+177	+708	**+3.07**	**+3.32**	**+2.26**	**+8.20**	**+23.33**

Table 3: Ablation of joint Proposer–Solver optimization. Disabling Proposer parameter updates while updating only Solver parameters collapses policy distribution and answer parse rates (%).

	Training updates		CausalPhys		ContPhy	
Configuration	Proposer	Solver	ACC	Parse rate	ACC	Parse rate
PhysEvolve	✓	✓	**54.50**	**97.84**	**39.54**	**100.00**
w/o Proposer-role updates	–	✓	15.01	26.07	17.85	99.90
Δ	–	–	+39.48	+71.77	+21.69	+0.10

Table 4: Training-time audit of the single-trajectory naive verifier across 100 rounds.

Prediction task	Mean graded reward	Exact accuracy	Zero-exact rounds
Forward	0.373	0.00%	100/100
Inverse	0.664	6.63%	75/100
Parameter	0.787	5.44%	83/100

points), dropping performance back to the un-finetuned Base level. This ablation empirically confirms that paired branch elision is the indispensable engine for acquiring causal physical reasoning.

Criticality of joint Proposer–Solver optimization. In Table 3, we freeze Proposer role updates, training the shared policy exclusively via Solver rollouts. Without co-evolution, the policy undergoes severe representational collapse: accuracy drops precipitously to 15.01% on CausalPhys and 17.85% on ContPhy, accompanied by a collapse in answer parse rate (falling to 26.07% on CausalPhys and 0% in later steps). Updating the Proposer keeps world generation diverse and physically non-trivial, preventing the Solver from over-adapting to a degenerate data stream.

4.4 Auditing the Single-Trajectory Interface

Q4: What causes naive physical self-play to collapse? We conduct an empirical audit of the single-trajectory verifier and task stream in Naive self-play to validate the theoretical pitfalls identified in Section 3.1.

Divergence between graded reward and exact correctness. Table 4 exposes a striking pathology: continuous proximity rewards s consistently provide deceptive reinforcement. For Forward prediction, rollouts achieve a high mean graded reward of 0.373 despite *zero* exact matches across all 100 rounds. For Inverse and Parameter tasks, mean rewards reach 0.664 and 0.787 while exact accuracy remains below 7%. An error audit reveals severe mode collapse: the policy learns to output stationary initial velocities (746 instances) or the median rung of the parameter ladder (145 instances), exploiting the graded metric to reap partial credit without solving the physical events.

Task space starvation. Over 100 training rounds, the naive single-trajectory verifier generated *zero* intervention questions and *zero* counterfactual tasks. Constrained to an un-intervened factual trajectory τ, the system is incapable of synthesizing causal supervision. Consequently, downstream PhysBench dynamics accuracy stagnates (Table 1).

5 Conclusion

PhysEvolve demonstrates that verifier-grounded self-evolution extends to continuous multimodal physical reasoning by anchoring self-play in executable 3D world programs and paired counterfactual branches. Converting continuous physics into discrete verifier contracts eliminates the dual traps of continuous reward hacking and causal blindness inherent in naive physical self-play. Strong targeted gains on Newtonian dynamics and zero-shot transfer to unseen continuous media and real-world video show that counterfactual execution provides a scalable foundation for generalizable physical intelligence in vision-language models, entirely without any human-annotated physical reasoning supervision during self-play training. Each training target is bound to a verified physical outcome.

AI USE STATEMENT

Generative AI tools assisted manuscript writing and editing, code implementation, experimental design, and the analysis and interpretation of results. All AI-assisted work was manually reviewed and verified by the authors, who take responsibility for the final text, implementation, experimental decisions, and reported conclusions. The use of a vision–language model to propose executable worlds and solve generated tasks is part of the research method described in Section 3.

ETHICS STATEMENT

The experiments reported here use simulated worlds and existing physical-reasoning benchmarks. These evaluations do not establish that the trained model is safe for deployment in physical systems. In particular, benchmark-level transfer should not be taken as a guarantee of reliable behavior under untested interventions or environments. Such deployment requires additional, task-specific reliability and safety evaluation. Use and redistribution of benchmark data and pretrained models should respect their respective terms.

REPRODUCIBILITY STATEMENT

The method and experimental design are described in Sections 3 and 4. Appendix A specifies the executable-world interface; Appendices C and D document task contracts, answer binding, validation, and branch execution. Reward and curriculum controls are provided in Appendix E. Appendix F reports the training loop, hyperparameters, construction seeds, run multiplicity, and benchmark evaluation protocols. Appendix G provides extended results and paired correctness analyses to support inspection of the reported comparisons.

REFERENCES

Pravesh Agrawal, Szymon Antoniak, Emma Bou Hanna, Baptiste Bout, Devendra Chaplot, Jessica Chudnovsky, Diogo Costa, Baudouin De Monicault, Saurabh Garg, Theophile Gervet, Soham Ghosh, Amélie Héliou, Paul Jacob, Albert Q. Jiang, Kartik Khandelwal, Timothée Lacroix, Guillaume Lample, Diego Las Casas, Thibaut Lavril, Teven Le Scao, Andy Lo, William Marshall, Louis Martin, Arthur Mensch, Pavankumar Muddireddy, Valera Nemychnikova, Marie Pellat, Patrick Von Platen, Nikhil Raghuraman, Baptiste Rozière, Alexandre Sablayrolles, Lucile Saulnier, Romain Sauvestre, Wendy Shang, Roman Soletskyi, Lawrence Stewart, Pierre Stock, Joachim Studnia, Sandeep Subramanian, Sagar Vaze, Thomas Wang, and Sophia Yang. Pixtral 12B. *arXiv preprint arXiv:2410.07073*, 2024.

Tayfun Ates, M. Atesoglu, Cagatay Yigit, Ilker Kesen, Mert Kobas, Erkut Erdem, Aykut Erdem, Tilbe Goksun, and Deniz Yuret. CRAFT: A benchmark for causal reasoning about forces and interactions. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2602–2627, 2022. doi: 10.18653/v1/2022.findings-acl.205.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025a. URL https://arxiv.org/abs/2511.21631.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025b. URL https://arxiv.org/abs/2502.13923.

Fabien Baradel, Natalia Neverova, Julien Mille, Greg Mori, and Christian Wolf. CoPhy: Counterfactual learning of physical dynamics. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1909.12000.

Peter W. Battaglia, Jessica B. Hamrick, and Joshua B. Tenenbaum. Simulation as an engine of physical scene understanding. *Proceedings of the National Academy of Sciences*, 110(45):18327–18332, 2013.

Daniel M. Bear, Elias Wang, Damian Mrowca, Felix J. Binder, Hsiao-Yu Fish Tung, R. T. Pramod, Cameron Holdaway, Sirui Tao, Kevin A. Smith, Fan-Yun Sun, Fei-Fei Li, Nancy Kanwisher, Joshua B. Tenenbaum, Daniel L. K. Yamins, and Judith E. Fan. Physion: Evaluating physical prediction from vision in humans and machines. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*, 2021.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Yixing Chen, Yiding Wang, Siqi Zhu, Haofei Yu, Tao Feng, Muhan Zhang, Mostofa Patwary, and Jiaxuan You. Multi-agent evolve: LLM self-improve through co-evolution. *arXiv preprint arXiv:2510.23595*, 2025.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024.

Zhenfang Chen, Kexin Yi, Yunzhu Li, Mingyu Ding, Antonio Torralba, Joshua B. Tenenbaum, and Chuang Gan. ComPhy: Compositional physical reasoning of objects and events from videos. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2205.01089.

Wei Chow, Jiageng Mao, Boyi Li, Daniel Seita, Vitor Guizilini, and Yue Wang. PhysBench: Benchmarking and enhancing vision-language models for physical world understanding. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2501.16411.

DeepSeek-AI. Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2012.02096.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Qihan Ren, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, Cheng Qian, Zhenhailong Wang, Minda Hu, Huazheng Wang, Qingyun Wu, Heng Ji, and Mengdi Wang. A survey of self-evolving agents: What, when, how, and where to evolve on the path to artificial super intelligence. *Transactions on Machine Learning Research*, 2026. arXiv:2507.21046.

Çağlar Gülçehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (ReST) for language modeling. *arXiv preprint arXiv:2308.08998*, 2023.

Yicheng He, Chengsong Huang, Zongxia Li, Jiaxin Huang, and Yonghui Yang. VisPlay: Self-evolving vision-language models from images. *arXiv preprint arXiv:2511.15661*, 2025.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-Zero: Self-evolving reasoning LLM from zero data. *arXiv preprint arXiv:2508.05004*, 2025.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Xinxi Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Christopher Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hanna Hajishirzi. Tülu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Dingming Li, Yingxiu Zhao, Xinrui Cheng, Kangheng Lin, Hongbo Peng, Hongxing Li, Zixuan Wang, Yuhong Dai, Haodong Li, Jia Wang, Yukang Shi, Liang Zhao, Jianjian Sun, Zheng Ge, Xiangyu Zhang, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. SpatialEvo: Self-evolving spatial intelligence via deterministic geometric environments. *arXiv preprint arXiv:2604.14144*, 2026.

William Liang, Sam Wang, Hung-Ju Wang, Osbert Bastani, Dinesh Jayaraman, and Yecheng Jason Ma. Eurekaverse: Environment curriculum generation via large language models. In *Conference on Robot Learning (CoRL)*, 2024. arXiv:2411.01775.

Bo Liu, Leon Guertler, Simon Yu, Zichen Liu, Penghui Qi, Daniel Balcells, Mickel Liu, Cheston Tan, Weiyan Shi, Min Lin, Wee Sun Lee, and Natasha Jaques. SPIRAL: Self-play on zero-sum games incentivizes reasoning via multi-agent multi-turn reinforcement learning. *arXiv preprint arXiv:2506.24119*, 2025. URL https://arxiv.org/abs/2506.24119.

Meta. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, 2024.

Jack Parker-Holder, Minqi Jiang, Michael Dennis, Mikayel Samvelyan, Jakob Foerster, Edward Grefenstette, and Tim Rocktäschel. Evolving curricula with regret-based environment design. In *Proceedings of the 39th International Conference on Machine Learning (ICML)*, 2022. arXiv:2203.01302.

Maitreya Patel, Tejas Gokhale, Chitta Baral, and Yezhou Yang. CRIPP-VQA: Counterfactual reasoning about implicit physical properties via video question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9856–9870, 2022.

Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009.

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. In *International Conference on Machine Learning (ICML)*, 2026. URL https://sim2reason.github.io/. arXiv:2604.11805.

Ronan Riochet, Mario Ynocente Castro, Mathieu Bernard, Adam Lerer, Rob Fergus, Véronique Izard, and Emmanuel Dupoux. IntPhys: A framework and benchmark for visual intuitive physics reasoning. *arXiv preprint arXiv:1803.07616*, 2018.

Elizabeth S. Spelke. Principles of object perception. *Cognitive Science*, 14(1):29–56, 1990.

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.

Leitian Tao, Ilia Kulikov, Swarnadeep Saha, Tianlu Wang, Jing Xu, Sharon Li, Jason E. Weston, and Ping Yu. Hybrid reinforcement: When reward is sparse, it's better to be dense. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2510.07242.

Hsiao-Yu Tung, Mingyu Ding, Zhenfang Chen, Daniel M. Bear, Chuang Gan, Joshua B. Tenenbaum, Daniel L. K. Yamins, Judith E. Fan, and Kevin A. Smith. Physion++: Evaluating physical scene understanding that requires online inference of different physical properties. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Qinsi Wang, Bo Liu, Tianyi Zhou, Jing Shi, Yueqian Lin, Yiran Chen, Hai Helen Li, Kun Wan, and Wentian Zhao. Vision-Zero: Scalable VLM self-improvement via strategic gamified self-play. *arXiv preprint arXiv:2509.25541*, 2025.

Rui Wang, Joel Lehman, Jeff Clune, and Kenneth O. Stanley. Paired open-ended trailblazer (POET): Endlessly generating increasingly complex and diverse learning environments and their solutions. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2019. arXiv:1901.01753.

Yuxiang Wei, Zhiqing Sun, Emily McMilin, Jonas Gehring, David Zhang, Gabriel Synnaeve, Daniel Fried, Lingming Zhang, and Sida Wang. Toward training superintelligent software agents through self-play SWE-RL. *arXiv preprint arXiv:2512.18552*, 2025. URL https://arxiv.org/abs/2512.18552.

Alex Wilf, Pranjal Aggarwal, Bryan Parno, Daniel Fried, Louis-Philippe Morency, Paul Pu Liang, and Sean Welleck. Propose, solve, verify: Self-play through formal verification. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2512.18160.

Peng Xia, Kaide Zeng, Jiaqi Liu, Can Qin, Fang Wu, Yiyang Zhou, Caiming Xiong, and Huaxiu Yao. Agent0: Unleashing self-evolving agents from zero data via tool-integrated reasoning. *arXiv preprint arXiv:2511.16043*, 2025.

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 (ICLR)*, 2020.

Abhay Zala, Jaemin Cho, Han Lin, Jaehong Yoon, and Mohit Bansal. EnvGen: Generating and adapting environments via LLMs for training embodied agents. In *Conference on Language Modeling (COLM)*, 2024. arXiv:2403.12014.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. STaR: Bootstrapping reasoning with reasoning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Andrew Zhao, Yiran Wu, Yang Yue, Tong Wu, Quentin Xu, Matthieu Lin, Shenzhi Wang, Qingyun Wu, Zilong Zheng, and Gao Huang. Absolute zero: Reinforced self-play reasoning with zero data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2505.03335.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. Learning to reason without external rewards. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2505.19590.

Zhicheng Zheng, Xin Yan, Zhenfang Chen, Jingzhou Wang, Qin Zhi Eddie Lim, Joshua B. Tenenbaum, and Chuang Gan. ContPhy: Continuum physical concept learning and reasoning from videos. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

Fengzhe Zhou, Jiannan Huang, Jialuo Li, Deva Ramanan, and Humphrey Shi. PAI-Bench: A comprehensive benchmark for physical AI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. arXiv:2512.01989.

A COMPLETE PHYSDSL AND EXECUTABLE WORLD INTERFACE

A.1 COMPLETE CALLABLE SURFACE

A proposal is a Python module containing `build_scene(api)` and a literal `SCENE_MANIFEST`. Every construction/setter call mutates a builder and returns `None`; the proposal cannot branch on simulator state. The executor alone calls `freeze()` to obtain a scene. Table 5 lists the complete public surface used by the current executor, including operations not exercised by every reported world class. Section B instantiates the interface with three auditable golden-world illustrations.

Table 5: Complete PhysDSL callable surface. Arguments are validated when the call is made; `freeze()` returns the immutable executor input.

Capability	Calls	Effect
Geometry	`add_box`, `add_sphere`, `add_cylinder`, `add_capsule`, `add_plane`	Register fixed or free rigid bodies and visual/material attributes.
Initial state	`set_velocity`, `set_rotation`	Set linear/angular velocity or a normalized orientation quaternion.
Constraints	`add_hinge`, `add_slide`, `add_spring`	Add one-DOF joints or a damped spatial spring between existing bodies.
Environment	`set_gravity`, `set_duration`, `add_camera`	Set the execution horizon, gravity vector, and look-at camera.
Finalization	`freeze`	Add default ground/camera if absent and snapshot a `SceneDef`.

Hard validation envelope. Table 6 gives the actual validator bounds. Stage 2 uses `continuous_levels=True`: mass, friction, and restitution remain the submitted real values inside their intervals rather than being snapped to the Stage-1 ladders.

Table 6: PhysDSL construction-time bounds. Intervals are closed unless an inequality is written explicitly.

Field	Enforced contract
Names	Pattern `[a-z_][a-z0-9_]*`; one namespace across bodies, joints, springs, and cameras; `ground/world/sky` reserved (`ground` is allowed for the plane).
Body centers and count	$\lvert x\rvert, \lvert y\rvert < 1.5$ and $0 < z < 2.5\,\mathrm{m}$; a plane may use $z = 0$. At least one body after finalization and at most 20 dynamic bodies; fixed scene furniture does not consume the dynamic-body cap.
Geometric size	Each non-plane radius/half-size/half-height in $[0.01, 1.0]\,\mathrm{m}$; each plane half-size is at least $0.01\,\mathrm{m}$ and has no API upper cap.
Material	Mass $[0.01, 100]\,\mathrm{kg}$, sliding friction $[0, 2]$, restitution $[0, 1]$, and every RGBA channel $[0, 1]$.
Initial state	Each linear and angular velocity component has magnitude at most 20; orientation is a nonzero quaternion and is normalized.
Joint	Nonzero axis; optional limits satisfy $\ell < u$; parent and child differ; only the parent may be `world`; one explicit joint per child.
Spring	Stiffness $[0.1, 10^4]$, damping $[0, 10^3]$, body-local anchor components in $[-2, 2]\,\mathrm{m}$, and optional rest length > 0; at least one endpoint is dynamic and neither endpoint is a plane.
Environment	Duration $[0.1, 60]\,\mathrm{s}$; finite three-vector gravity; camera position and target must differ.

A.2 REPORTED EVIDENCE SCHEMAS

A reasoning intent specifies the semantic relation to probe and the evidence needed to decide it. PhysDSL can express more worlds than the reported run uses; the run enables exactly the three schemas in Table 7, which jointly serve the eight active families in Section C. The executor resolves every declared role against the frozen world before constructing an item.

B AUDITABLE WORLD-PROGRAM GALLERY

Three human-authored *reference worlds* illustrate the executable world-program interface: Throwing, Collision, and Motion. We execute the v01 programs with seed 7 in MuJoCo 3.6.0, retaining

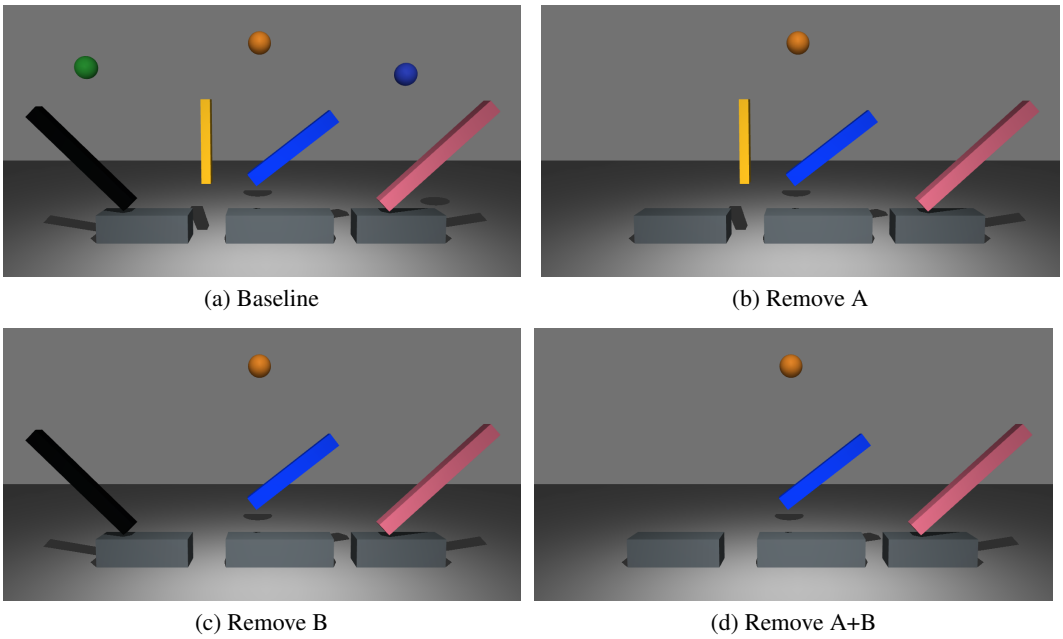
Table 7: The three SCENE_MANIFEST contracts enabled in the reported run. Braces denote a body-role mapping, not model-predicted state.

World class	Required role fields	Served active families
Throwing	balls, three or more fixed platforms, two or more removable walls, region_kind, and three destination regions derived from platform gaps	final destination; current-run judgment; counterfactual destination; inverse intervention
Collision	dynamic target; trigger; exactly four dynamic movers; relation_pair={a,b,expected}	final state; final relation; motion attribution
Motion	dynamic motion_target; optional speed_pair={a,b}	motion-state change, including forward, speed-profile, and pairwise surface forms

their colors, materials, and lighting. Initialization images precede the first simulation step; subsequent frames show the accepted world's execution. Program copies, evidence schemas, accepted scenes, and execution records are stored under paper/assets/world_program_demo/. The rendered components are stored separately under paper/figures/world_program_demo/.

B.1 THROWING: ONE PROGRAM, FOUR EXECUTABLE INPUTS

The reference program defines three balls, three platforms, four walls, and three destination regions. Figure 5 uses the orange ball (ball_b) as the query. Wall A is black_left_wall; wall B is orange_wall. Each branch preserves the retained bodies' initial conditions and removes the designated walls together with the two non-query balls.



(a) Baseline (b) Remove A

(c) Remove B (d) Remove A+B

Figure 5: Throwing reference world and executable interventions. The frozen baseline and three altered inputs share the same camera and initial conditions for retained bodies. Each altered input removes the designated wall set and the non-query balls. The executor runs these inputs separately to obtain the region readouts used for question and answer construction.

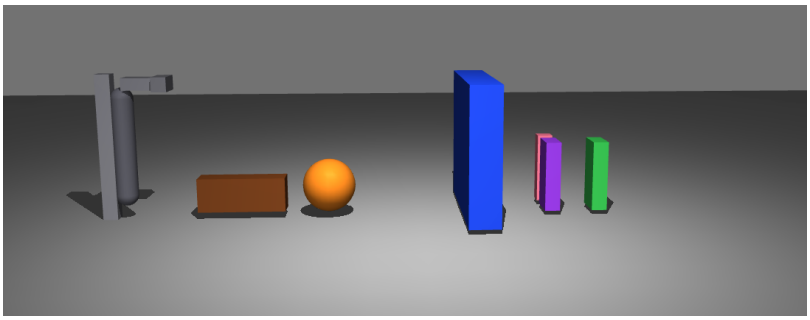
The orange ball settles in the left pit in the baseline and in the middle pit when wall A and the other balls are removed. This executed difference supplies the counterfactual answer. The four evidence objects in Table 8 also support inverse questions by comparing which interventions reach a requested destination.

Table 8: Throwing evidence objects. Each altered input is executed with the same seed and horizon as the frozen baseline.

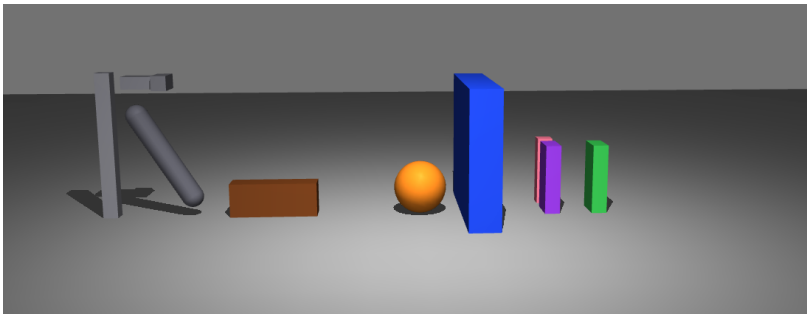
Evidence key	Elided bodies	Readout
$\tau_0(b)$	none	baseline settled region
$\tau_A(b)$	wall A and all non-query balls	altered settled region
$\tau_B(b)$	wall B and all non-query balls	altered settled region
$\tau_{AB}(b)$	walls A+B and all non-query balls	altered settled region

B.2 Collision: roles and contact evidence

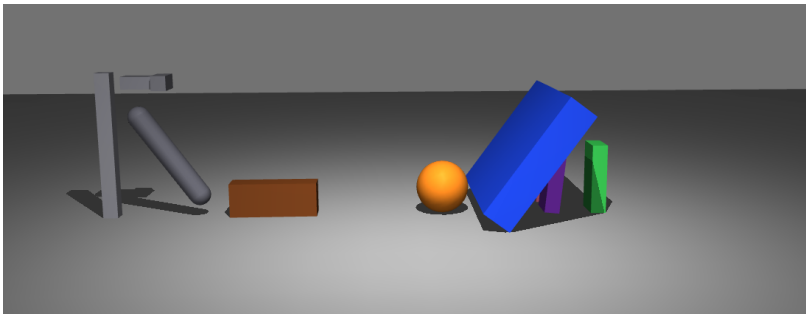
The evidence schema names `swing_striker` as the trigger, `blue_plate` as the target, four attribution candidates, and the relation pair `orange_ball`/`blue_plate`. These roles support attribution, final-state, and relation questions.



(a) Initialization, $t = 0$ s



(b) $t = 1.032$ s

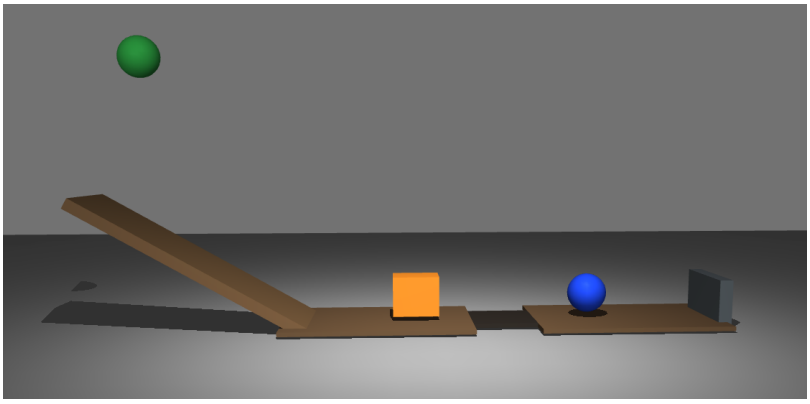


(c) $t = 1.678$ s

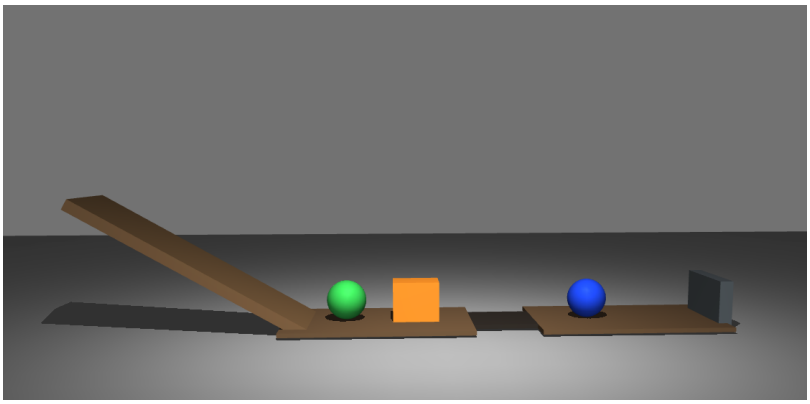
Figure 6: Collision reference world. The initialization and later frames connect the entity roles to execution. The lower panels bracket contact between the orange ball and blue plate; times are measured from simulation start.

B.3 MOTION: AN APPARATUS FOR MOTION AND CONTACT

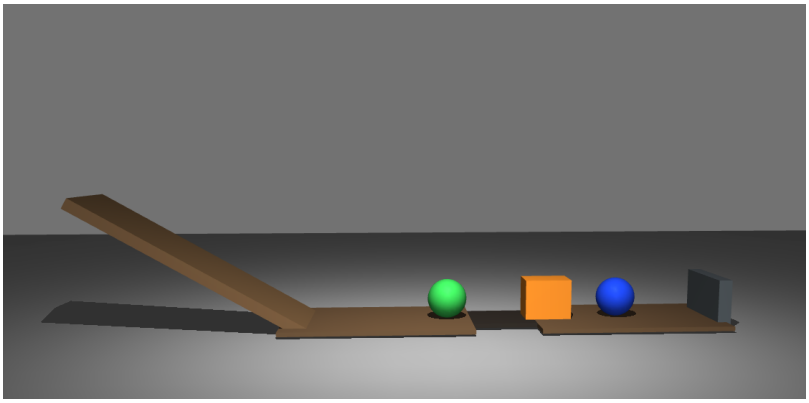
The Motion world combines a ramp, a platform with a separate surface patch, a slider, and two balls. The green ball descends the ramp and contacts the amber slider, which subsequently contacts the blue ball. Its trajectory provides evidence for the active motion-state family.



(a) Initialization, $t = 0$ s



(b) $t = 0.790$ s



(c) $t = 1.356$ s

Figure 7: Motion reference world. Initialization and two execution states show the ramp-to-platform motion and resulting contacts. All frames share the same camera; times are measured from simulation start.

Table 9: Exact contracts for the four active throwing families. The option descriptions give canonical semantics; an eligible surface layout may permute slots and substitute compatible distractors, but must preserve a unique binding for the admitted answer semantic.

Family	Representative item and option semantics	Evidence, binding, and suppression
`throwing_final_destination`	"What is the final destination of the [referent]?" Regions use {left pit, right pit, center pit, none}; an unsettled layout uses {left, right, neither, none}.	Baseline only. A region requires final speed ≤ 0.30 m/s, a non-ejected status, exactly one derived gap interval, and final height no more than 0.30 m above the platform tops. A settled region binds that region; otherwise *neither* is bound rather than guessed.
`throwing_current_run_judgment`	"Will the [referent] eventually drop into the left pit?" Options are {Yes, No, Can not answer, Uncertain}; a second surface asks whether it reaches a different destination region than a named one.	Baseline only. *Yes* iff the settled-region readout equals the named target; otherwise *No*, including an unsettled/ejected readout. Target choice alternates between matching and nonmatching regions across ball indices.
`counterfactual_destination`	"If we removed the [wall] and other balls, which pit would the [referent] drop into?" Options are {left, right, center, none}.	Matched baseline and one elision branch. Admit only when the branch is available, settles in one region, and differs from the baseline region. The branch region is the answer; failed, unsettled, or unchanged branches suppress the item.
`inverse_intervention`	"What can we do to make the [referent] drop into the [target] pit?" Options are {remove A + other balls, remove B + other balls, remove both + other balls, none}.	Baseline plus all three A/B/A+B branches. In a first fixed-order pass, the first non-baseline target reached by exactly one branch binds that intervention. Only if no target has a unique achiever, a second pass binds *none* to the first non-baseline target reached by no branch. A multiply achieved target is skipped; missing branches suppress.

Table 10: Exact contracts for the other four active families. Canonical options are shown; eligible layouts may change slot order and compatible distractors while preserving the admitted answer semantic.

Family	Representative item and option semantics	Evidence, binding, and suppression
`collision_final_state`	"Which phrase best describes the final pose of the [target]?" Options are {standing upright, leaning, lying horizontally, floating}.	Terminal status and speed. Ejected with $z > 0.35$ binds floating; fallen binds lying; moving with speed ≥ 0.10 binds leaning; stable with speed ≤ 0.20 binds standing. Other/borderline combinations suppress the item.
`collision_final_relation`	"Which phrase best describes the interaction between A and B during the run?" Options are {ends in contact, never touches, passes near without touching, ends vertically separated without contact}; a binary surface uses {Yes, No, Sometimes, Uncertain}.	Contact is trajectory-level: any recorded pairwise contact dominates and binds the contact semantic (canonically rendered as "ends in contact"). Otherwise vertical separation requires final $\lvert\Delta z\rvert \geq 0.30$ and lateral gap < 0.30 m; pass-by requires initial or final distance ≤ 0.20 m; the remainder is no-contact. The computed class must equal the manifest declaration or the family is suppressed.
`motion_attribution`	"Which object experienced a change in motion as a result of the [trigger]?" The four options are the manifest's four mover bodies.	Candidates must be distinct, dynamic, and different from the trigger. Eligibility requires trigger contact and velocity-change magnitude $\lVert\mathbf{v}_f - \mathbf{v}_0\rVert_2 > 0.30$ m/s; the argmax is bound only if its lead over a second eligible candidate is at least 0.30 m/s. The absence of an eligible candidate, or a lead below this margin, suppresses the item.
`motion_state_change`	Three surfaces: motion change {stop, slow but moving, speed up, same}; speed profile {accelerate, decelerate, accelerate then decelerate, constant}; pairwise semantics {final tie, final A faster, final B faster, non-tied order reversal}. The canonical middle options use "consistently faster" wording.	Require target initial speed ≥ 0.10 m/s. Stop uses final speed < 0.10; speed-change/profile boundaries use an absolute 0.30 m/s gap; pairwise order uses 0.10 m/s. Crossover is bound only when both endpoint orders are non-tied and reverse; otherwise the final order is bound. The pairwise surface requires both declared bodies.

C Intervention-Grounded Task Contracts

The implementation follows one order for every family: define the physical semantic, test whether the executed evidence makes it decidable, map the admitted case to an answer semantic, and only then realize a question and option layout. Tables 9 and 10 enumerate all eight active families. Frozen registry entries are not counted.

Table 11: Branch-result contract used for counterfactual and inverse items.

Execution	Baseline	Remove A	Remove B	Remove A+B
Stored semantic	r_0	r_A	r_B	r_{AB}
Required for counterfactual(A)	yes	yes	no	no
Required for inverse	yes	yes	yes	yes

C.1 Branch matrix and unique-answer binding

For a query ball b, let $r_0, r_A, r_B, r_{AB} \in \{\text{left}, \text{right}, \text{middle}, \perp\}$ be the settled-region readouts of the four executions in Table 8; $\perp$ denotes no admitted region. Table 11 formalizes which cells each family must observe before any option letters are assigned.

A counterfactual(A) item is admitted exactly when $r_A \neq \perp$ and $r_A \neq r_0$, then binds answer semantic r_A. For an inverse target $t \neq r_0$, define $H_t = \{A : r_A = t,\ B : r_B = t,\ AB : r_{AB} = t\}$. The generator requires all three branch views and makes two fixed-order passes over left/right/middle. The first returns the first target with $|H_t| = 1$ and binds its sole lever. Only when no target has a unique achiever does the second return the first target with $|H_t| = 0$ and bind *none of the above*; a target with $|H_t| > 1$ is skipped in both passes. Thus "none" is an executor-derived negative, not a catch-all for failed execution, and an ambiguous target does not preclude a decidable target elsewhere.

C.2 Surface realization boundary

Only after an answer semantic is admitted does the task-interface library select a compatible stem and four-option layout and bind the semantic to an index. A canonical family layout is used whenever a sampled surface lacks the required semantic. This ordering keeps the physical decision rule executor-determined while allowing multiple linguistic realizations of the same evidence.

D Verification, Repair, and Branch Execution

Typed termination. Every proposal receives exactly one status. `FAILED_SYNTAX`, `FAILED_SEMANTIC`, and `FAILED_RUNTIME` cover parsing, eager DSL validation, and `build_scene` execution. `FAILED_COMPILE`, `FAILED_OVERLAP`, and `FAILED_NAN` cover simulator-model, initial-interpenetration, and numerical failures. `FAILED_TIMEOUT` and `FAILED_RENDER` cover bounded execution and rendering. `FAILED_VQA` means the manifest or generated item violates its contract. `TRIVIAL` means maximum displacement is below 0.05 m and there is no collision. Only `OK` worlds with admitted items enter Solver groups.

Bounded repair. Repairs are recorded as an ordered trace. Minor repairs address supported syntax/argument variants, manifest repairs recover fields mechanically determined by authored geometry, and geometry repairs clamp workspace positions or separate simple initial overlap. Repair does not add a new body, joint, spring, or declared causal relation. The strongest applied tier yields a proposal-credit multiplier of 0.90 (minor), 0.80 (manifest), or 0.75 (geometry); unrepaired worlds use 1.0.

Frozen matched branches. After repair, the executor calls `freeze()` once. A branch clones that immutable `SceneDef`, elides its named body set and every incident joint/spring, and re-executes with the baseline seed and horizon. Identical removal sets share one cached execution. At most six distinct removal sets are run per world; failed/capped branches are absent, causing dependent items to be suppressed. The throwing scheduler places the A/B/A+B inverse sets for the first two balls before compatible counterfactual requests, so the latter can reuse the same branch views.

E Optimization and Active Curriculum

General objective. Let z be the terminal proposal status, n the number of retained Solver rollouts, $G_{\min} = 4$, and I the intrinsic score. Before repair, anti-copy, saturation control, and family scaling,

Algorithm 1 Executor-determined item construction under an explicit contract

Require: proposal module p, stage-2 DSL, simulation seed s
parse `build_scene` and literal `SCENE_MANIFEST`
execute the sandboxed builder with eager DSL checks
apply bounded repair and freeze one immutable world W
compile, execute, and render baseline $\tau_0 \leftarrow \mathcal{E}(W, s)$
validate manifest roles against W and enumerate required elisions
for each distinct removal set B up to the branch cap **do**
    $W_B \leftarrow W$ with B and incident constraints elided
    $\tau_B \leftarrow \mathcal{E}(W_B, s)$; cache by B
end for
for each requested active family f **do**
    compute evidence predicates from τ_0 and required τ_B
    if all decidability and margin conditions for f hold **then**
        derive answer semantic; select a compatible surface layout
        bind semantic to one option index and emit the item
    end if
end for

the implementation defines

$$r_{\text{base}} = \begin{cases} \lambda_{\text{fail}} c_z, & z \neq \texttt{OK}, \\ 0, & z = \texttt{OK} \text{ and } n < G_{\min}, \\ \lambda_{\text{learn}}^{\text{eff}} r_{\text{learn}} + \lambda_{\text{int}} I, & \text{otherwise.} \end{cases} \tag{4}$$

Equation 4 separates terminal failure, insufficient retained evidence, and admitted worlds. Here $r_{\text{learn}} = 1 - 2|\bar{a}_{\text{prop}} - \frac{1}{2}|$ is the instance-level tent statistic, where $\bar{a}_{\text{prop}}$ denotes the retained-rollout Solver accuracy used to score the proposal. The Solver receives exact option-match reward. Status prices are $c_z = -1.0$ for syntax, semantic, and VQA failure, -0.8 for runtime failure, -0.5 for compile and overlap failure, -0.3 for numerical, timeout, and render failure, and 0 for a trivial world.

Reported instantiation and hierarchical co-evolution. Co-evolution operates through a two-level curriculum. At the macro level, family sampling tracks the Solver's competence boundary across reasoning intents: after an eight-step prior-weighted cold start, selection uses $\max(0.05, 4\bar{a}_f(1 - \bar{a}_f))$ multiplied by the structural priors in Table 12, so saturated or unsolvable families are down-weighted. The family-difficulty multiplier further scales Proposer credit by rolling $\bar{a}_f$, so Solver outcomes reshape both which intents are sampled and how admitted worlds are valued. At the micro level, conditioned on the sampled intent σ, the shared policy explores the program space for valid, physically non-trivial environments. Instance-level scoring prioritizes physical validity, collision feasibility, and structural diversity I, with the instance-level tent coefficient held at $\lambda_{\text{learn}}^{\text{eff}} = 0$. This hierarchy combines competence-adaptive task selection with verifier-grounded exploration of executable worlds. The executor supplies the verifiable answers that connect both levels. Executable demonstrations remain available as syntactic references. Through this dual-level loop, Solver progression reshapes the training distribution via the adaptive intent curriculum, and shared-parameter updates of π_θ align world generation with physical reasoning.

The active proposal signals are typed failure/validity, repair credit, structural diversity, anti-copy, saturation control, and family-difficulty scaling; the active Solver signal is exact option match. Table 12 records these controls.

Exact Solver outcomes update the rolling family statistics used for subsequent rounds. Family sampling directs task generation toward the Solver's competence boundary, and the family-difficulty multiplier adjusts Proposer credit across intents. Within each intent, validity, repair, diversity, and saturation controls shape executable world generation.

Table 12: Reward and sampling controls in the reported 100-step run.

Component	Reported configuration
Failure scale λ_{fail}	1.5
Instance-level tent coefficient $\lambda_{\text{learn}}^{\text{eff}}$	held at 0 (validity/diversity dominate instance scoring)
Intrinsic weight λ_{int}	0.1; structural diversity only
Repair factors	0.90 minor; 0.80 schema; 0.75 geometry
Near-copy control	similarity ≥ 0.95 capped at 0.1 before repair and family scaling
Exact/within-group duplicate proposal	pre-family reward override -0.8
Edit-distance diversity weight	1.5
Saturation control	threshold 0.75; weight 1.0
Family-difficulty multiplier	$\text{clip}(1 + (0.5 - \bar{a}_f), 0.5, 1.5)$
Rolling family statistic	64 most recent per-world family accuracies
Family-sampling cold start	8 rounds
Family-sampling priors	$2.0\times$ for priority families; $2.0\times$ for `collision_final_relation`; $1.2\times$ for each branch family; $1.0\times$ otherwise
Solver reward scale	1.5

Algorithm 2 Reported verifier-grounded shared-policy training loop

Require: shared policy π_θ, executor $\mathcal{E}$, intent curriculum $\mathcal{C}$, intent batch N, group size G

for $t = 1, \ldots, 100$ **do**
  sample $\{\sigma_n\}_{n=1}^{N} \sim \mathcal{C}$ with scene-kind minima
  sample G program/manifest proposals for each σ_n
  validate proposals and execute Algorithm 1
  retain up to the per-scene and per-step Solver-group caps
  sample G Solver answers per retained item; score exact option match
  compute proposal rewards and role-wise GRPO advantages
  update the shared π_θ from Proposer and Solver rows
  update rolling family accuracy and $\mathcal{C}$
end for

F Reproducibility, Training, and Evaluation Configuration

F.1 Role contracts and update loop

The shared policy alternates between two tagged interfaces. The Proposer receives a reasoning intent and returns a world program plus manifest inside `<propose>` tags. The Solver receives a rendered video and an executor-authored multiple-choice item and returns one option inside `<answer>` tags; it has no simulator tool. The fixed executor joins the roles by validating programs, executing the required baseline/branches, and binding each admitted answer.

F.2 Resolved training configuration

Table 13 transcribes the v7i launcher and current configuration builder. The launcher fixes a 100-step horizon and saves every 25 steps. We report the terminal step-100 checkpoint and show every scheduled intermediate checkpoint in Figure 4.

F.3 Evaluation protocols

Table 14 gives the evaluation contract used for each benchmark.

PhysBench frames are passed directly to the Qwen processor as an array. For CausalPhys, five annotations whose gold field is not an A–D label are excluded before scoring. ContPhy evaluates one question per generation and normalizes extracted open responses by lowercasing and stripping whitespace. All models within one benchmark share the item list, visual preprocessing, prompt, decoding, answer parser, and scorer. An unparsed response is scored as incorrect.

Table 13: Resolved configuration for the reported PhysEvolve run.

Component	Configuration
Base and horizon	Qwen3-VL-8B-Instruct; 100 steps; checkpoints at 25/50/75/100
Run multiplicity	One training run per reported configuration; no training-seed sweep
Compute	4 GPUs; 32 Ray CPUs; 4 executor workers
Proposer sampling	$N = 4$ intent groups per step; $G = 4$ programs per intent; at least one throwing, collision, and motion intent per step
Solver retention and rollout	At most 5 items per scene and 32 groups per step; 4 answers per group; rollout chunk size 8
Visual variants	One additional color permutation per accepted baseline world; one executor camera; 32 rendered frames at 480×480
Golden demonstration pool	10 throwing, 8 collision, and 6 motion programs; active for steps 1–100
Auxiliary geometry item	Camera-depth bypass enabled; at most one executor-geometry item on an eligible collision/motion scene, outside the eight sampled intents
Generation	temperature 0.7; top-$p = 0.95$; top-$k = 20$; 6,144-token prompt and 4,096-token response budgets
GRPO	1 epoch; clip ratio 0.2; learning rate 10^{-6}; entropy coefficient 0
Batch sizes	training batch 4; prompt-group mini-batch 1; per-GPU micro-batch 1
Reference regularization	frozen reference policy; low-variance actor KL and reward KL coefficient 0.03
Reference buffer	enabled during demonstrations; capacity 32; 2 references per prompt
Family sampling	8-step prior-weighted cold start; accuracy-adaptive sampling thereafter. Structural priors throughout: $2.0\times$ for priority and relation coverage; $1.2\times$ for counterfactual/inverse families.
Online construction seeds	executor sequence starts at 9001; curriculum/reference RNG base 9001 (resolved code defaults)
Executor limits	0.002 s timestep; 30 s simulation and 15 s render timeouts; at most 6 distinct branch executions per world

Table 14: Evaluation protocols for the three reported benchmarks.

Benchmark	Items	Visual input	Generation	Scoring
PhysBench	9,802	8 uniform frames	greedy; 2,048 tokens	official prompts, normalization, and scorer
CausalPhys	3,057	5 uniform frames	CoT; greedy; 512 tokens	final A–D answer; unparsed output is incorrect
ContPhy	1,950	10 uniform frames	greedy; 64 tokens	normalized MC or normalized open-response exact match

We report accuracy and percentage-point differences on the shared item set within each benchmark.

G Extended Results

G.1 PhysBench taxonomy

Table 15 reports the matched claim-facing views. Table 16 then gives every top-level task and ability partition from the same step-100 evaluation, plus the three physical dynamics subtypes used in the main text.

G.2 Complete contextual reference table

Table 17 restores the complete set of views for the open-weight contextual references. These heterogeneous scales are not used for component attribution; Base, Naive, and PhysEvolve remain the matched 8B comparison.

G.3 CausalPhys categories and ContPhy domains

Table 18 exposes all four CausalPhys categories and all four ContPhy physical domains from the reported protocols. The intervention-operation views used in the main text are included after the domain partition.

The complete branch-path control is reported in Table 2.

Table 15: PhysBench accuracy and paired differences for matched models.

View (n)	Base	Naive	Δ vs. Base	PhysEvolve	Δ vs. Base
Overall (9,802)	52.92	52.39	-0.53	53.47	$+0.55$
Dynamics (2,932)	44.37	44.44	$+0.07$	45.29	$+0.92$
Throwing (410)	46.10	45.85	-0.24	54.88	$+8.78$
Collision (650)	38.62	38.77	$+0.15$	40.77	$+2.15$
Motion (1,433)	54.71	55.48	$+0.77$	56.45	$+1.74$
Prediction (1,430)	42.17	42.87	$+0.70$	43.71	$+1.54$
Judgment (767)	34.94	35.20	$+0.26$	40.03	$+5.08$
Reasoning (2,056)	43.09	40.42	-2.68	43.92	$+0.83$
Static (1,899)	60.66	61.40	$+0.74$	62.24	$+1.58$

Accuracies and differences are percentage points. Difference columns compare Naive and PhysEvolve with Base. Task, subtype, and ability levels overlap; rows within each top-level partition do not.

Table 16: Complete top-level PhysBench task and ability partitions for the matched Base and PhysEvolve models. The dynamics-subtype rows are nested inside Dynamics and therefore do not form a complete partition.

Taxonomy level	View (n)	Base	PhysEvolve	Δ
Task	Dynamics (2,932)	44.37	45.29	+0.92
	Relationships (2,088)	61.97	63.46	+1.48
	Scene (2,240)	45.13	45.62	+0.49
	Property (2,542)	62.20	61.61	-0.59
Dynamics subtype	Throwing (410)	46.10	54.88	+8.78
	Collision (650)	38.62	40.77	+2.15
	Motion (1,433)	54.71	56.45	+1.74
Ability	Perception (919)	60.39	56.04	-4.35
	Dynamic (189)	75.13	75.66	+0.53
	Static (1,899)	60.66	62.24	+1.58
	Comparison (1,423)	45.89	44.91	-0.98
	Reasoning (2,056)	43.09	43.92	+0.83
	Identify (1,119)	82.93	82.84	-0.09
	Judgment (767)	34.94	40.03	+5.08
	Prediction (1,430)	42.17	43.71	+1.54

G.4 Joint Proposer–Solver updates

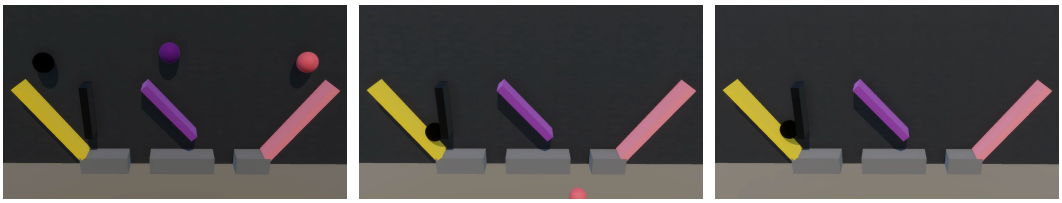
An additional run removed Proposer-role rows from the shared-policy update while retaining Proposer generation, the executor, and the family set. Its training answer parse rate remains valid through step 64 and becomes zero for steps 65–100. Table 3 reports endpoint accuracy and answer parse rates for the joint and Solver-only update configurations.

G.5 Task-level transfer and paired answer changes

Table 19 gives the exact accuracies underlying Figure 3. The CausalPhys Naive and branch-disabled results use the same video-packing protocol as the Base and full-model evaluations. All comparisons retain the same valid-gold evaluation set.

Table 20 uses the official PhysBench answer-matching rule to partition the matched questions by terminal-checkpoint correctness. The difference between the two discordant counts, divided by the row's sample size, recovers the accuracy change. Throwing and Motion are taxonomy subtypes; Judgment is an ability view and can overlap the subtype views. Counts are therefore interpreted within each row.

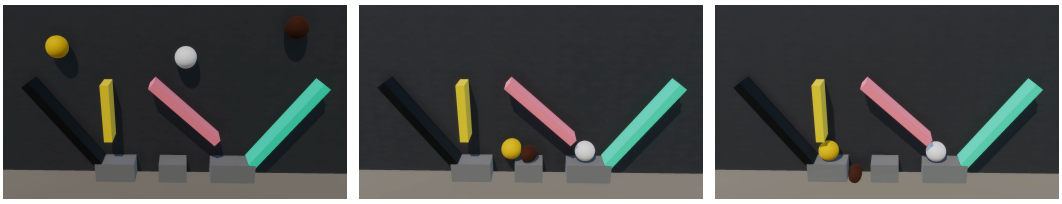
Counterfactual Removal *ContPhy soft:00086*



If we removed the black floating wall and other balls, which pit would the purple ball drop into?
A. The left pit
B. The right pit
C. None of the above

Base: A / Naive: A / w/o branch: A / PhysEvolve: B **Gold: B**

Goal-Pit *ContPhy soft:00038*



What can we do to make the brown ball drop into the right pit?
A. Remove the pink floating wall and other balls
B. Remove the yellow floating wall and other balls
C. None of the above

Base: A B / Naive: A B / w/o branch: A / PhysEvolve: C **Gold: C**

Figure 8: Matched ContPhy examples for Counterfactual Removal and Goal-Pit. Each example displays three of the ten evaluation input frames, the original question and options, the extracted model answers, and the reference answer. The examples illustrate reasoning about removal and a requested destination in held-out soft-body scenes. The Goal-Pit reference answer is "None of the above."

Table 17: Full accuracy (%) table for open-weight context and matched 8B models.

	PhysBench								CausalPhys			ContPhy		
Model	Ovr.	Dyn.	Throw.	Coll.	Motion	Pred.	Judg.	Reason.	Ovr.	Antic.	Interv.	Ovr.	CF-Rm.	Goal-Pit
InternVL2.5-8B	43.8	42.1	40.0	38.9	27.7	43.0	32.7	27.4	50.2	51.5	42.4	31.4	48.4	23.3
Llama-3.2-11B-Vision	38.7	43.2	36.6	35.5	27.2	44.1	30.8	29.6	44.2	44.1	40.1	20.7	0.0	20.0
Pixtral-12B	46.9	44.6	41.2	39.2	35.2	46.9	31.0	35.1	50.2	53.0	44.5	21.5	40.2	13.3
Qwen2.5-VL-7B	49.4	44.3	44.4	36.5	50.6	44.1	33.1	36.5	52.6	56.5	45.7	25.9	11.5	24.4
Qwen3-VL-32B	60.9	48.7	47.1	46.0	83.5	48.0	44.3	51.4	58.9	61.9	52.9	41.0	45.9	26.7
Qwen3-VL-8B Base	52.9	44.4	46.1	38.6	54.7	42.2	34.9	43.1	53.9	55.9	42.8	37.1	44.3	20.0
Naive self-play	52.4	44.4	45.9	38.8	55.5	42.9	35.2	40.4	55.8	57.6	45.9	37.4	43.4	25.6
PhysEvolve	53.5	45.3	54.9	40.8	56.5	43.7	40.0	43.9	54.5	58.4	45.0	39.5	51.6	44.4

Table 18: Matched cross-benchmark accuracy and complete category/domain breakdowns.

Benchmark view (n)	Base	PhysEvolve	Δ
CausalPhys Overall (3,057)	53.94	54.50	$+0.56$
Anticipation (896)	55.92	58.37	$+2.46$
Goal orientation (637)	50.71	49.61	-1.10
Intervention (573)	42.76	45.03	$+2.27$
Perception (951)	60.99	59.83	-1.16
ContPhy Overall (1,950)	37.08	39.54	$+2.46$
Cloth (450)	48.00	47.11	-0.89
Fluid (450)	24.89	28.44	$+3.55$
Rope (600)	30.83	32.17	$+1.33$
Soft body (450)	46.67	52.89	$+6.22$
Counterfactual removal (122)	44.26	51.64	$+7.38$
Goal-Pit (90)	20.00	44.44	$+24.44$

Table 19: Complete accuracies for the task-level transfer comparison (%). CausalPhys controls use the matched video-packing protocol.

Task	n	Base	Naive	w/o branch	PhysEvolve
ContPhy					
Counterfactual Removal	122	44.26	43.44	43.44	**51.64**
Goal-Pit	90	20.00	25.56	21.11	**44.44**
CausalPhys / Anticipation					
Collision Prediction	300	53.67	50.67	**54.67**	51.67
Fluid Flow	200	47.50	**60.50**	35.50	58.50
Intention Speculation	196	45.92	**52.55**	42.35	48.47
Deformation	200	77.50	70.00	**78.00**	**78.00**
CausalPhys / Intervention					
Action Substitution	99	38.38	46.46	44.44	**47.47**
Spatial Manipulation	151	43.05	**46.36**	41.06	45.03
Temporal Shifting	149	54.36	**59.73**	50.34	57.05
Viewpoint Transformation	174	**35.06**	33.33	33.33	33.33

Table 20: Paired terminal-checkpoint correctness on PhysBench. Each row partitions the same evaluation items by Base and PhysEvolve correctness.

Task	n	Both correct	Ours only	Base only	Neither	Δ (pp)
Overall	9802	4487	754	700	3861	+0.55
Throwing	410	156	69	33	152	+8.78
Judgment	767	220	87	48	412	+5.08
Motion	1433	730	79	54	570	+1.74