MECHANISM-GUIDED ITERATION FOR EXECUTION-GROUNDED GAME DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Language-model agents can revise executable game designs, but successful execution alone does not establish meaningful mechanism improvement or transferable design knowledge. The challenge is to connect execution feedback with specific rule changes and reusable experience. We propose mechanism-guided iteration, a frozen-backbone framework that couples rule-level diagnosis with Verified Mechanism Memory. Candidates are represented as canonical GameSpecs and executed by a fixed interpreter. Diagnosis identifies a local revision target, while bounded Component–Mechanism–Skill retrieval supplies typed components, executable rules, and replay-verified solution traces. A separate, method-blind evaluator assesses predictive compression, initial-state transfer, compositional novelty, mechanism interaction, and discovered strategy diversity. Matched controls separate diagnosis and memory from feedback and repeated sampling; an optional slow-policy extension is evaluated separately. The framework makes mechanism reuse an explicit, testable design choice without equating executable validity, surface novelty, or illustrative rule changes with creative quality.

1 INTRODUCTION

Creative game design rarely starts entirely from scratch. New designs can reuse familiar components and mechanics while recombining them into new interactions and solution structures, consistent with the view of creativity as recombination of existing elements (Boden, 1994). For language-model agents, this raises a basic question: can past experience become reusable design knowledge without reducing future designs to variations of previously seen templates? This question is especially important in executable game design, where a design may run successfully yet still contain disconnected mechanics, collapse to a single trivial strategy, or work only under the conditions seen during development. A useful design process should therefore go beyond achieving executable validity and support the recombination of known mechanisms into behaviorally distinct structures that remain effective under changed conditions.

Iterative feedback and external memory are natural tools for this task. Self-Refine and Reflexion study refinement through feedback (Madaan et al., 2023; Shinn et al., 2023), while Skill-Pro studies the reuse of skills derived from prior experience (Mi et al., 2026). What remains unclear is whether reusable mechanism knowledge provides value beyond iterative feedback itself. In particular, *given the same execution feedback and resource ceiling, does diagnosing a rule-level failure and retrieving grounded rule fragments improve the next design?* Answering this question requires controls that separate the effects of feedback, diagnosis, and memory: a method that receives richer feedback or additional revision opportunities cannot isolate the contribution of reusable knowledge.

Our hypothesis is that diagnosis and memory play complementary roles in iterative design. Execution traces provide evidence of *what happened*; diagnosis identifies *what needs to change*; and reusable rule fragments provide grounded mechanism knowledge that can inform *how to revise the design*. We operationalize this hypothesis as a bounded revision procedure: execute a candidate, diagnose its development profile, retrieve relevant entries from a "Verified Mechanism Memory (VMM)", and request a local edit. VMM stores typed state components, executable rule fragments, and replay-verified solution traces extracted from prior successful designs. The main method keeps the diagnosis policy fixed and adapts only the current design and bounded external memory, while the model parameters, evaluator, candidate selector, and resource ceilings remain unchanged. Figure 1 previews

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

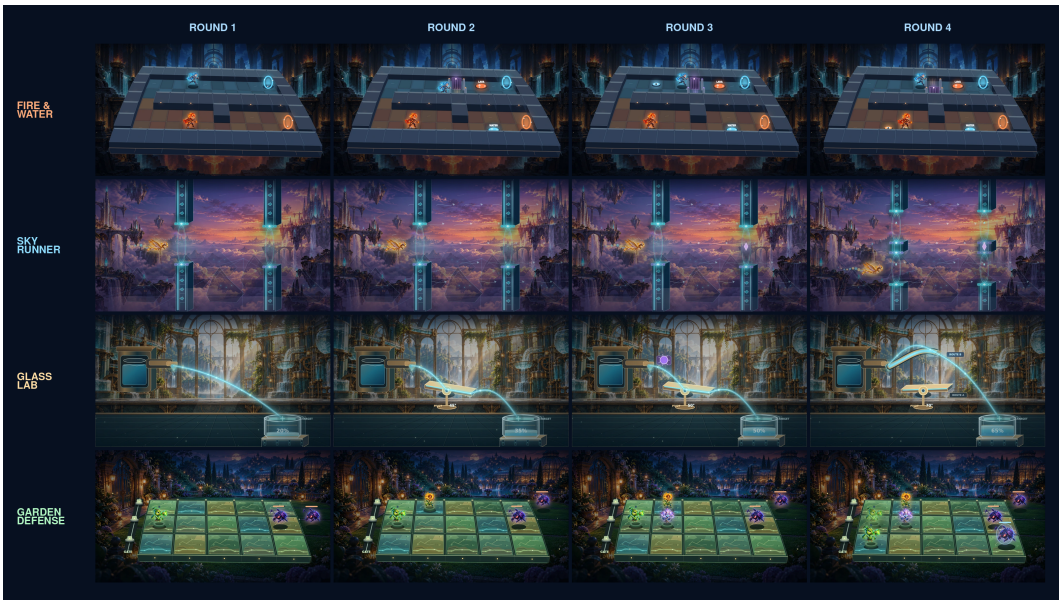


Figure 1: Illustrative presentation of four browser-game lineages across four revision rounds (columns). The rows correspond to Fireboy and Watergirl (Fire & Water), Flappy Bird (Sky Runner), Happy Glass (Glass Lab), and Plants vs. Zombies (Garden Defense). Section 5.2.5 summarizes their rule-level evolution; Appendix E gives the version sequences. The panels are qualitative illustrations, not quantitative evidence of a memory benefit.

four illustrative revision sequences; Section 5.2.5 and Appendix E describe the rule changes behind their similar-looking screenshots.

To test whether mechanism reuse improves more than executable validity, we evaluate each design along several complementary behavioral dimensions, including predictive transfer, effective interaction among mechanics, compositional novelty, and discovered strategy diversity. In particular, we use predictive compression to ask whether explanatory rules induced from past executions continue to predict behavior under changed initial states. Inspired by minimum description length (Rissanen, 1978; Grünwald, 2007), this diagnostic compares an observation-induced mechanism predictor against a past-fitted sham predictor. We treat compression as one probe of transferable structure rather than as a definition of creativity. All methods are evaluated with the same method-blind evaluator, and matched component controls isolate the effects of diagnosis, memory representation, retrieval, and persistence under shared feedback and resource ceilings.

Our contributions are threefold. First, we formulate mechanism-guided iteration as a bounded design loop linking executable rule-level diagnosis to retrieval from Verified Mechanism Memory. Its Component–Mechanism–Skill representation connects reusable rule fragments to typed state components and replay-verified solution traces. Second, we specify a method-blind evaluation protocol that separates retrospective compression, initial-state transfer, interaction, novelty, and discovered strategy diversity. Third, we organize matched comparisons around diagnosis, memory representation, retrieval, persistence, and optional slow-policy adaptation under shared feedback and resource ceilings. Qualitative evolution cases complement these comparisons by tracing how familiar game actions acquire new rule-level consequences. Together, these contributions support controlled analysis of mechanism reuse rather than assuming that additional memory or revision necessarily improves design quality.

2 Related Work

Iterative language-agent refinement. ReAct interleaves reasoning and environment actions (Yao et al., 2022). Self-Refine uses self-feedback, while Reflexion stores verbal feedback from prior attempts (Madaan et al., 2023; Shinn et al., 2023). We test whether rule-level diagnosis and execution-

verified memory help under matched feedback and budgets, using feedback-only and diagnosis-without-memory controls.

Reusable skills and self-improving agent systems. Voyager stores executable skills, ExpeL retrieves experience-derived insights, and DreamCoder learns reusable program libraries (Wang et al., 2023; Zhao et al., 2024; Ellis et al., 2021). Skill-Pro learns reusable skills from agent experience (Mi et al., 2026). Automated Design of Agentic Systems and the Darwin Gödel Machine instead search agentic systems or modify agent implementations (Hu et al., 2025; Zhang et al., 2026). Our frozen-backbone method changes game specifications and bounded C–M–S memory, not agent code, model weights, or the evaluator. Comparison rows are protocol-matched adaptations, not official-system reproductions.

Game generation and quality–diversity search. Search-based and learning-based procedural generation study content representations and evaluation (Togelius et al., 2011; Summerville et al., 2018); CreativeGame targets mechanic-aware generation (Ma et al., 2026). Quality–diversity search separates behavioral coverage from solution quality (Mouret & Clune, 2015; Cully & Demiris, 2017). Our bounded archive is not a MAP-Elites grid, and discovered strategy diversity does not establish human enjoyment.

Mechanism-centered evaluation and predictive compression. Alchemy, XLand-MiniGrid, Craftax, I-PHYRE, and Multiverse Mechanica expose controlled transformations, rules, or transfer conditions (Wang et al., 2021; Nikulin et al., 2024; Matthews et al., 2024; Li et al., 2024; Osazuwa Ness et al., 2026). They are adapter targets, not interchangeable game-design benchmarks. Our compression measures draw on minimum description length (Rissanen, 1978; Grünwald, 2007). Unlike an MDL training objective, RMCP and TMCP are post-generation diagnostics comparing an observation-induced predictor with a past-fitted sham. We therefore report metric saturation and adapter-specific failure modes separately.

3 Structured Designs, Explanations, and Memory

3.1 Task and fixed constraints

Given a brief x, the designer returns a design document and a canonical GameSpec

$$g = (\mathcal{E}_g, \mathcal{X}_g, \mathcal{U}_g, \mathcal{F}_g, \mathcal{O}_g),$$

containing entities, typed state variables, public actions, executable mechanic rules, and objectives. A fixed interpreter induces the transition kernel $P_g(s' \mid s, a)$ for state s and public action a. Let $\ell_{\mathrm{art}}(g)$ be the canonical serialized length of the artifact and B_{art} its shared maximum; specification growth is constrained by

$$\ell_{\mathrm{art}}(g) \le B_{\mathrm{art}}. \tag{1}$$

Schema validation, bounded execution, and a solver-discovered winning trajectory define executable validity. The current complete-profile selector additionally requires all five diagnostics to be finite and applicable. Executable validity and metric applicability are reported separately; absence of a supported transfer probe is not assigned a zero score.

Model parameters θ remain frozen. Adaptation changes the current design and external memory, not the model weights or interpreter. Each experiment freezes the prompts, scenario constructors, learner, metrics, reference library, budgets, and selector in a manifest. Changes create a new comparison; scores from different evaluators are not pooled.

3.2 Reusable memory is distinct from an explanatory model

Verified Mechanism Memory (VMM) uses the Component–Mechanism–Skill representation $\Omega_t = (\mathcal{C}_t, \mathcal{M}_t, \mathcal{S}_t)$. Components in $\mathcal{C}_t$ are typed state-variable definitions. Mechanisms in $\mathcal{M}_t$ are executable rules with preconditions, action triggers, effects, and scope. Skills in $\mathcal{S}_t$ are successful action sequences with their source initial conditions and terminal goal. These are concrete records, not additional learned model parameters.

A candidate enters memory only after complete-profile validity and replay of a winning trajectory. Only rules activated by that replay are retained as mechanisms. Entries carry source identifiers and execution evidence; deduplication, bounded capacities, and lexical retrieval control storage and

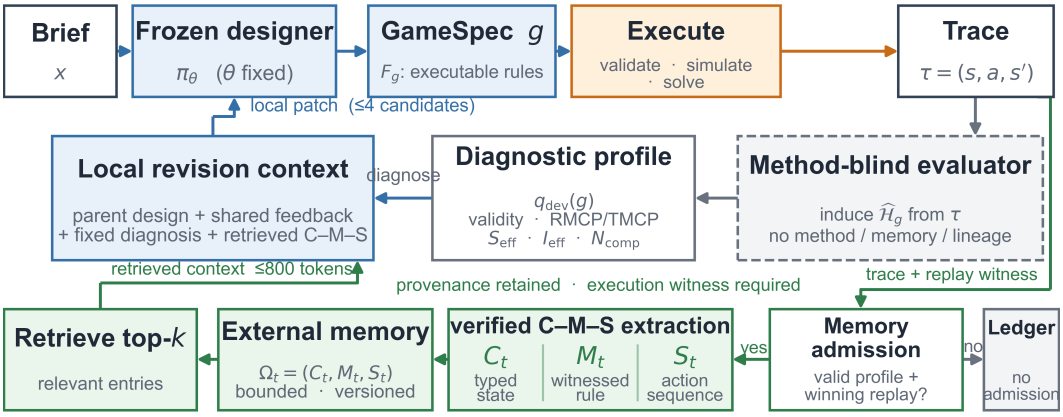


Figure 2: Mechanism-guided game design evolution across iterative revision rounds

prompt size. Retrieved mechanisms carry their referenced typed components. A verified source-game skill is not assumed to remain executable after transplantation; the target game is executed again. No per-rule compression threshold or unimplemented MDL admission gate is claimed.

VMM retains up to 24 components, 24 mechanisms, 12 skills, and eight non-dominated archive entries. Canonical duplicates are replaced and oldest excess entries removed. Lexical retrieval uses the brief, diagnosis, edit operator, and target mechanic IDs, returning mechanisms with their typed components. Retrieval defaults to three entries (configurable from one to four) and at most 800 input tokens. Candidate and API logs preserve provenance after eviction.

The shared evaluator separately constructs an explanatory set $\widehat{\mathcal{H}}_g$ from observations. Thus $\mathcal{F}_g$ denotes executable game rules, $\mathcal{M}_t$ denotes remembered rule fragments, and $\widehat{\mathcal{H}}_g$ denotes observation-induced explanations. Giving the evaluator the method's own memory would confound design quality with privileged explanatory access.

3.3 Observation-only induction and predictive transfer

A fixed encoder maps executed transitions to events $y_h = \psi(s_h, a_h, s_{h+1})$. Numeric changes are deltas; the preceding state and action remain the conditioning context. The evaluator collects past records D_{past}, groups them by action, and fits a bounded conditional tree over observed state fields. Its leaves specify conditions and permitted event sets. The learner receives neither executable rule code nor transfer records. Appendix B specifies collection and induction.

For a fixed predictor p_η, where η denotes all frozen evaluator parameters, define

$$L(D; \mathcal{H}) = -\sum_{(s_h, a_h, y_h) \in D} \log_2 p_\eta(y_h \mid s_h, a_h, \mathcal{H}). \tag{2}$$

The bounded reporting evaluator fits a sham $q_0(y \mid a)$ on D_{past} using action-conditioned event frequencies mixed with a normalized finite-byte-string tail. The sham deliberately ignores state. The mechanism predictor $q_H(y \mid s, a, \widehat{\mathcal{H}}_g)$ uses the same tail. After induction, both q_0 and $\widehat{\mathcal{H}}_g$ are frozen. Let $L_0(D)$ and $L_H(D)$ denote their respective code lengths. We report

$$\text{RMCP}(g) = \frac{L_0(D_{\text{past}})}{L_0(D_{\text{past}}) + L_H(D_{\text{past}}) + \varepsilon_L}, \tag{3}$$

$$\text{TMCP}(g) = \frac{L_0(D_{\text{init}})}{L_0(D_{\text{init}}) + L_H(D_{\text{init}}) + \varepsilon_L}, \tag{4}$$

where D_{init} contains records from changed initial values with rules, objectives, and public interfaces preserved, and $\varepsilon_L = 10^{-12}$ is a numerical division guard. All observed code lengths are positive, so the guard is below reporting precision. Both scores lie strictly in $(0, 1)$: up to that guard, .5 is equal code length, $> .5$ favors the mechanism predictor, and $< .5$ favors the sham. A missing D_{init}

makes TMCP unavailable. Rule-removal records use the same frozen predictors but are excluded from Equation 4.

Initial-state probes independently perturb up to four used variables in identifier order, excluding those in win/lose conditions. Numeric values change by ± 1 within bounds, Booleans toggle, and enums use their first two alternatives. Rules, objectives, and interfaces remain fixed. The original forward witness is replayed for at most 24 actions at seed 2713, retaining probes that fail to win.

Boundedness does not establish construct validity or make differences linear in bits. Scores still depend on serialization, learner capacity, and sampled trajectories. We therefore retain raw bit savings, encoded event length, and the original normalized scores in the audit appendix. Compression is a predictive diagnostic, not an optimal-code guarantee or a direct measure of human creativity.

4 Mechanism-Guided Design Iteration

4.1 Fixed diagnosis and local revision

The generation-time evaluator used the prespecified normalized scores, whereas the bounded RMCP and TMCP in Equations 3–4 were computed only after candidate selection. To keep these roles distinct, let $\mathrm{nRMCP}_{\mathrm{dev}}$ and $\mathrm{nTMCP}_{\mathrm{dev}}$ denote the scores available to the designer. The development evaluator returns validity, probe outcomes, and

$$\mathbf{q}_{\mathrm{dev}}(g) = [N_{\mathrm{comp}}(g), \mathrm{nRMCP}_{\mathrm{dev}}(g), \mathrm{nTMCP}_{\mathrm{dev}}(g), S_{\mathrm{eff}}(g), I_{\mathrm{eff}}(g)]_{\mathrm{dev}}. \tag{5}$$

Invalid designs receive repair instructions. Otherwise, the deterministic diagnosis returns both an edit category—transfer, strategy, interaction, novelty, or compression—and the target GameSpec rule IDs. The model is instructed to revise the located rules rather than choose a target from the aggregate score alone. The transfer deficit uses the generation-time $\mathrm{nRMCP}_{\mathrm{dev}}$–$\mathrm{nTMCP}_{\mathrm{dev}}$ gap under unchanged rules; the bounded reporting scores were never fed back to the model or selector. Rule-removal failures are separate diagnostics that can motivate an independent backup route. The deficit heuristic and tie order appear in Appendix B; they are not an estimated causal attribution model.

Using the retrieval contract in Section 3.2, each revision request combines the current design, common execution feedback, and retrieved C–M–S memory. It asks the model to edit the located rules with the smallest relevant change, preserve the brief and existing successful routes, and keep narrative, GameSpec, and example playthrough in agreement. The model returns a complete first design or an incremental patch to a parent identified by a digest. Every resolved patch is validated and executed. Preservation instructions do not prove semantic equivalence or guarantee monotonic improvement.

4.2 Execution, consolidation, and shared selection

Within a brief, the main method proposes, executes, and revises at most four candidates. Each revision uses the immediately preceding candidate as its parent; if that candidate is invalid, the next call repairs it using its validation errors. Valid intermediate candidates can contribute verified memory during revision, while all attempts remain in the outcome ledger. Standard comparison cells start from empty memory; only the ordered reuse study carries assigned memory across briefs (Appendix B). Only after iteration stops does the shared selector retain complete-profile valid candidates, discard Pareto-dominated candidates, and compare sorted per-metric ranks lexicographically from the weakest rank. Ties favor a shorter canonical artifact and then an earlier proposal; if none passes, the fallback uses design validity, GameSpec validity, and generation order without marking the outcome successful. The main method keeps diagnosis priority, retrieval count, edit scope, and verification instructions fixed. An optional slow extension can update only these fields after two-fixture validation; Appendix B.4 gives its gate and charged costs, and fixture acceptance does not establish improvement on future briefs.

4.3 Shared diagnostics and their limits

The evaluator receives only a canonical GameSpec, not a method label, parent, memory, or lineage. Every arm uses the same learner and scenario construction. All iterative component controls also receive identical feedback fields. Optimizing a diagnostic does not independently validate it; model selection can overfit its evaluation criterion (Cawley & Talbot, 2010).

Effective interaction. For each ordered pair of active mechanics (f_i, f_j), the evaluator replays fixed actions with and without f_i, preserving the public state and action spaces. Let K_g be the number of such active mechanics. A target-local distance $\Delta_{i\to j}$ combines changes in applicability, activation, outcomes, reachable states, and reward at target activation. With operational edge indicator $\chi_{i\to j}$,

$$I_{\text{eff}}(g) = \frac{1}{K_g(K_g-1)} \sum_{i\neq j} \chi_{i\to j}\Delta_{i\to j}, \tag{6}$$

for $K_g \geq 2$, and zero otherwise. The reward term is not a replanned action-value estimate. The threshold is a fixed default; formal sham calibration remains necessary.

Compositional novelty. N_{comp} compares connected typed motifs of sizes two to four in the effective interaction graph with a frozen reference library, weighting motifs by successful-trace support. The pilot library has five hand-authored motifs per size. This measures distance to that basis, not novelty relative to a representative corpus of games.

Strategic richness. A fixed solver ensemble discovers successful near-optimal trajectories. Clustering their mechanic-activation signatures yields solver-balanced class frequencies p_c and

$$S_{\text{eff}}(g) = \exp\left(-\sum_c p_c \log p_c\right). \tag{7}$$

This measures discovered strategy diversity, not exhaustive strategic depth. Appendix A gives complete calculations. All five diagnostics are accompanied by validity, applicability, artifact size, and model usage; none substitutes for human assessment of design appeal.

5 Evaluation and Results

We compare mechanism-guided iteration with matched controls in CreativeGame, test whether memory persists across related cases, and examine the same diagnostics in external environments. The calibrated compression scores were defined after inspecting development results, so the comparisons below describe observed behavior rather than a sealed confirmation. Appendix C.1 gives the sampling and analysis units.

5.1 Experimental setup

Research questions. RQ1. Does rule-level diagnosis improve revision under the same feedback, with and without structured memory? **RQ2.** Does verified C–M–S memory improve outcomes over no memory or raw traces, and which memory components matter? **RQ3.** Do the observed differences persist under calibrated scoring and across external environments? **RQ4.** Does optional slow-policy adaptation improve outcomes enough to justify its cost?

Benchmarks. CreativeGame is the primary comparison: each method generates executable GameSpecs from the same briefs. A separate ordered-case study tests whether memory carried between related briefs helps later designs. Five external adapters test whether the diagnostics remain informative across different rule systems; we report their outcomes separately rather than combining scores across environments.

Baselines. Independent sampling controls for repeated proposals, while feedback-only revision tests the value of iteration without explicit diagnosis or memory. Diagnostic no-memory adds targeted edits, and fixed-revision C–M–S supplies structured memory with generic edits; together these controls isolate diagnosis with and without memory. Diagnostic raw-memory retrieves verified source trajectories instead of C–M–S entries. The full method, labeled **Ours (Diagnostic C–M–S)**, combines diagnosis with verified C–M–S; its optional slow-policy variant additionally permits bounded policy updates. Tables 1, 2, and 3 separate the main comparison, component controls, and slow-policy extension, respectively. Random retrieval and unverified memory test relevance and execution-gated admission. We also include protocol-matched Self-Refine/Reflexion (Madaan et al., 2023; Shinn et al., 2023) and Skill-Pro (Mi et al., 2026) adaptations, without claiming to reproduce their original systems. Appendix C specifies the component controls and Appendix D the literature adaptations.

Measures and implementation. Transfer compression (TMCP) is the primary diagnostic; retrospective compression (RMCP), validity, profile coverage, novelty, interaction, and strategy diversity

provide complementary views. All CreativeGame methods use the same frozen `gpt-5.4` backbone, evaluator, final selector, and resource ceilings; methods that revise candidates receive the same feedback fields. The bounded compression rescore uses stored selected candidates without new model calls or reselection. Quality means include only applicable profiles, and paired intervals respect the case or curriculum as the independent unit. Budget, applicability, and error-handling details appear in Appendix B.

5.2 Results

5.2.1 Does mechanism memory improve design quality?

On CreativeGame, the main method reaches TMCP .500, effectively the sham tie point, compared with .518 for feedback-only revision and .516 for diagnostic no-memory (Tables 1 and 2). Its RMCP is also below feedback-only revision (.752 versus .817). Across the eleven distinct conditions in Tables 1–3, the main method has the highest observed strategy diversity, $S_{\text{eff}} = 2.000$, but the primary compression scores do not show a transfer advantage from structured memory.

Table 1: Main-method comparison on CreativeGame. Each method has 500 attempts; values summarize case means, with quality metrics computed only where applicable. RMCP and TMCP lie in $(0, 1)$; .5 denotes equal mechanism and sham code lengths.

Method	Valid	Coverage	N_{comp}	RMCP	TMCP	S_{eff}	I_{eff}
Independent	.69(0)	.70(0)	.50(.0079)	.76(.076)	.50(.00099)	1.6(.23)	.059(.020)
Feedback-only	.80(.44)	.80(.45)	.49(.0081)	.81(.045)	.52(.034)	1.1(.24)	.074(.016)
Self-Refine	.77(.20)	.73(0)	.54(.13)	.81(.061)	.51(.011)	1.7(1.1)	.049(.016)
Reflexion	.77(.010)	.73(0)	.44(.13)	.78(.061)	.53(.011)	1.9(1.1)	.049(.016)
Skill-Pro adaptation	.81(.28)	.73(.28)	.42(.10)	.78(.021)	.50(.0050)	1.8(.75)	.052(.020)
Ours (Diagnostic C–M–S)	.83(.17)	.81(.18)	.58(.041)	.86(.069)	.55(.03)	2.0(.68)	.061(.19)

Table 2: Component controls on CreativeGame, separated from the main baselines.

Method	Valid	Coverage	N_{comp}	RMCP	TMCP	S_{eff}	I_{eff}
Diagnostic, no memory	.86(.18)	.86(.18)	.44(.11)	.78(.098)	.52(.036)	1.6(.50)	.053(.014)
C–M–S, fixed revision	.67(.23)	.67(.24)	.50(.018)	.76(.090)	.50(.0010)	1.8(.69)	.055(.013)
Diagnostic, raw memory	.86(.18)	.87(.18)	.47(.028)	.76(.080)	.51(.014)	1.4(.11)	.049(.020)
Random-retrieval C–M–S	.73(.28)	.73(.28)	.49(.038)	.81(.039)	.50(.0010)	1.4(.44)	.049(.016)
Unverified C–M–S	.87(.18)	.87(.18)	.46(.074)	.77(.056)	.50(0)	1.5(.42)	.048(.013)
Ours (Diagnostic C–M–S)	.90(.18)	.88(.18)	.58(.041)	.86(.069)	.57(.0030)	2.0(.68)	.059(.019)

Table 3: Optional slow-policy extension on CreativeGame. Ours is the fixed-policy full method; Ours + slow-policy update is an extension, not the main method. Each has 15 attempts.

Method	Valid	Coverage	N_{comp}	RMCP	TMCP	S_{eff}	I_{eff}
Ours (Diagnostic C–M–S)	.79(.18)	.80(.18)	.47(.041)	.75(.068)	.50(.0030)	2.0(.68)	.049(.019)
Ours + slow-policy update	.60(.43)	.60(.43)	.49(.0030)	.75(.041)	.50(.0010)	1.7(.19)	.062(.032)

Paired comparisons give the same picture (Table 4). The main-minus-fixed-revision TMCP difference is $+.001[-.0003, .004]$; differences against no memory and raw memory are $-.033[-.106, .006]$ and $-.083[-.031, .002]$. All three intervals include zero. Strategy diversity is higher than with raw memory ($+.703[.042, 1.449]$) or unverified C–M–S ($+.515[.026, .930]$). These secondary differences support a change in the generated design profile, not a gain in predictive transfer.

Table 4: Paired CreativeGame contrasts, Ours (Diagnostic C–M–S) minus comparator. Intervals resample cases and nested repeats; quality comparisons use jointly applicable profiles.

Comparator	Δ TMCP [95% interval]	Δ validity [95% interval]	Δ RMCP [95% interval]	ΔS_{eff} [95% interval]
C–M–S, fixed revision	$+.0012$ $[-.00030, .0041]$	$+.13$ $[-.20, .53]$	$+.021$ $[-.031, .078]$	$+.13$ $[-.67, .44]$
Diagnostic, no memory	$-.033$ $[-.11, .0060]$	$-.067$ $[-.33, .20]$	$-.0079$ $[-.032, .022]$	$+.40$ $[-.25, 1.1]$
Diagnostic, raw memory	$+.083$ $[-.031, .0020]$	$+.068$ $[-.33, .20]$	$-.0060$ $[-.065, .080]$	$+.70$ $[.042, 1.4]$
Random-retrieval C–M–S	$+.0034$ $[-.00060, .0078]$	$+.067$ $[-.27, .40]$	$+.055$ $[-.11, .036]$	$+.78$ $[-.11, 1.6]$
Unverified C–M–S	$+.0021$ $[-.00070, .0064]$	$-.067$ $[-.40, .26]$	$-.00030$ $[-.066, .064]$	$+.51$ $[.026, .93]$
Self-Refine / Reflexion	$-.0072$ $[-.027, .0040]$	$+.20$ $[-.47, 0]$	$-.079$ $[-.13, .014]$	$+.35$ $[-.74, 1.2]$
Skill-Pro adaptation	$+.0033$ $[-.0014, .0073]$	$+.067$ $[-.34, .47]$	$+.086$ $[-.22, .022]$	$+.25$ $[-1.3, 1.1]$

5.2.2 Does persistent memory help across cases?

We compare persistent C–M–S with reset C–M–S, persistent raw memory, and no memory on later cases in each curriculum. Persistent memory has lower mean TMCP than reset memory (.512 versus .520; Table 5). The paired persistent-minus-reset contrast is $-.026[-.052, -.0002]$, and persistent-minus-no-memory is $-.022[-.044, -.0001]$. Only two curricula contribute complete paired transfer profiles, making these differences fragile. Persistent C–M–S nevertheless has higher paired strategy diversity than no memory ($+.669[.159, 1.261]$) and higher interaction than reset memory ($+.013[.001, .029]$). The result is an observed quality–diversity trade-off, not evidence that persistence improves transfer.

Table 5: Cross-case reuse on later curriculum cases. Metric means use applicable profiles; token totals cover all 75 game executions per arm. Paired intervals, reported in the text, require joint applicability.

Arm	RMCP	TMCP	Valid	S_{eff}	I_{eff}	Tokens (M)
Persistent C–M–S	.882	.556	.920	1.689	.073	2.035
Reset C–M–S	.755	.520	.883	1.627	.058	2.076
Persistent raw memory	.731	.497	.833	1.613	.057	2.049
No memory	.724	.510	.733	1.493	.059	2.093

5.2.3 Do the diagnostics distinguish external methods?

Table 6 shows TMCP variation across methods rather than uniform sham ties. CreativeGame ranges from .53 to .72, whereas Multiverse Mechanica spans .06–1.00, crossing the .5 sham point. The other four external environments have lower endpoints between .55 and .71 and upper endpoints of 1.00. These reported endpoints alone do not establish ceiling saturation. The ranges are not confidence intervals and do not identify which method attains either endpoint. Coverage also varies within every environment. We therefore report environment-specific outcomes without inferring an overall method ranking or pooling scores.

Table 6: Benchmark-specific adapter outcomes. Completed counts are environment-level totals; ranges span eleven methods in CreativeGame and three in each external environment, not confidence intervals. Endpoint methods are not identified.

Environment	Completed	Validity range	Coverage range	TMCP range	TMCP interpretation
CreativeGame	165	.60–1.00	.60–1.00	.53–.72	Above sham
Multiverse Mechanica	45	.81–1.00	.72–1.00	.06–1.00	Spans sham point
Symbolic DM Alchemy	50	.67–1.00	.65–1.00	.71–1.00	Above sham
XLand-MiniGrid	50	.82–1.00	.56–1.00	.61–1.00	Above sham
I-PHYRE	50	.77–1.00	.66–1.00	.58–1.00	Above sham
Craftax	50	.63–1.00	.54–1.00	.55–1.00	Above sham

5.2.4 Does slow adaptation justify its cost?

The slow variant used 435,000 model tokens, compared with 413,000 for the fixed main method, yet its TMCP remained near .500 and its validity was lower (.600 versus .800; Table 3). Persistent C–M–S likewise used slightly more tokens than no memory in the reuse study (2.135 million versus

2.093 million). These development results do not show a quality gain that offsets the extra model use. Appendix C.1 reports the complete usage accounting.

5.2.5 WHAT CHANGES IN THE ILLUSTRATED DESIGNS?

Figure 1 summarizes four game lineages across four successive versions, covering platform coordination, one-button flight, fluid routing, and lane defense. In *Fireboy and Watergirl*, parked avatars, replay ghosts, sensors, and gravity-dependent movement organize character switching around a memory-relay puzzle. The *Flappy Bird* lineage combines clean-pass edits to later gates, failed-run echoes, and timed phase states within one-button flight. These interactions already appear in its first version; later versions add wind-crease interactions and connect these features through progression objectives. In *Happy Glass*, early droplet storage develops into delayed gravity programming, followed by a final design that separates directional release from limited manual gravity changes. In *Plants vs. Zombies*, blocked allied shots initially disappear, but later versions store them as charge and use lane bending to strengthen subsequent discharge.

The shared pattern is a change in how familiar actions connect across time: switching, passing, drawing, and blocking acquire consequences for later states. Appendix E gives the version sequences and archive identifiers in the figure's row order, distinguishing retained mechanics from later additions. These cases describe implemented rule changes, not measured improvements in player experience or evidence that memory caused those changes. They are separate from the controlled GameSpec comparisons.

6 DISCUSSION AND LIMITATIONS

The method builds on iterative refinement and skill memory (Madaan et al., 2023; Shinn et al., 2023; Mi et al., 2026). Its question is whether fixed mechanism-level diagnosis improves how grounded memory is used for game-rule revision. It does not introduce feedback, skill storage, or archives as new ideas. Agentic-system search and self-modifying agents are related settings (Hu et al., 2025; Zhang et al., 2026); our main method does not rewrite its agent code or depend on recursive improvement of its own learning procedure. The extension shows why component contrasts are necessary: a mechanism-memory design can improve discovered strategy diversity while failing to improve calibrated retrospective compression or transfer.

Several boundaries remain substantive. The CreativeGame panel has 15 attempts per method and the reuse panel 75 game runs per arm, but at most five independent cases or curricula contribute to each comparison; their intervals remain imprecise development evidence. One Skill-Pro adaptation is not an official-system reproduction. GameSpec solvability does not establish fidelity to every natural-language constraint, and preservation prompts do not enforce all intended semantics. A normalized code can still favor long or easily memorized events. The novelty basis is small, interaction calibration incomplete, and solver coverage limited. All five diagnostics are exposed during search, so blinded human assessment and a sealed evaluator are needed to distinguish better designs from better optimization of the measurement system. The updated adapter ranges show score variation, but do not identify a superior method or establish comparable calibration across environments. Cross-backbone claims, cross-environment superiority, and a positive transfer claim therefore require more independent cases and calibrated untouched experiments.

7 CONCLUSION

We introduced an execution-grounded protocol for testing whether rule-level diagnosis and verified C–M–S memory improve iterative game design. The updated adapter summaries show TMCP above the sham point in five environments and spanning it in Multiverse Mechanica. These ranges do not identify a winning method or establish a benefit from mechanism reuse. Component and curriculum comparisons remain necessary to isolate diagnosis, memory representation, and persistence under shared feedback. Positive secondary contrasts in discovered strategy diversity should not be interpreted as a general transfer advantage. Stronger conclusions require method-resolved paired comparisons, calibrated held-out evaluation, and independent human judgments.

REPRODUCIBILITY STATEMENT

Our evaluation focuses on structured, executable game specifications and does not fully capture playability, visual quality, or human enjoyment. The experiments use a single language-model backbone, so generalization to other models remains untested. The updated external summaries show varying compression scores, but omit method-resolved contrasts and uncertainty; they do not establish cross-environment superiority. Moreover, the diagnostics guide revision, so improvements may partly reflect adaptation to the evaluation criteria rather than broader design quality. Future work should test additional models and held-out cases, calibrate the external metrics, and include blinded human evaluation.

AI USE STATEMENT

Language models were used to assist with polishing the language of the manuscript.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

Margaret A Boden. Précis of the creative mind: Myths and mechanisms. *Behavioral and brain sciences*, 17(3):519–531, 1994.

Gavin C Cawley and Nicola LC Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *The Journal of Machine Learning Research*, 11:2079–2107, 2010.

Antoine Cully and Yiannis Demiris. Quality and diversity optimization: A unifying modular framework. *IEEE Transactions on Evolutionary Computation*, 22(2):245–259, 2017.

Kevin Ellis, Catherine Wong, Maxwell Nye, Mathias Sablé-Meyer, Lucas Morales, Luke Hewitt, Luc Cary, Armando Solar-Lezama, and Joshua B Tenenbaum. Dreamcoder: Bootstrapping inductive program synthesis with wake-sleep library learning. In *Proceedings of the 42nd acm sigplan international conference on programming language design and implementation*, pp. 835–850, 2021.

Peter D Grünwald. *The minimum description length principle*. The MIT press, 2007.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *International Conference on Learning Representations*, volume 2025, pp. 21344–21377, 2025.

Shiqian Li, Kewen Wu, Chi Zhang, and Yixin Zhu. I-phyre: Interactive physical reasoning. In *International Conference on Learning Representations*, volume 2024, pp. 28195–28215, 2024.

Hongnan Ma, Han Wang, Shenglin Wang, Tieyue Yin, Yiwei Shi, Yucong Huang, Yingtian Zou, Muning Wen, and Mengyue Yang. Creativegame: Toward mechanic-aware creative game generation. *arXiv preprint arXiv:2604.19926*, 2026.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023.

Michael Matthews, Michael Beukman, Benjamin Ellis, Mikayel Samvelyan, Matthew Jackson, Samuel Coward, and Jakob Foerster. Craftax: A lightning-fast benchmark for open-ended reinforcement learning. *arXiv preprint arXiv:2402.16801*, 2024.

Qirui Mi, Zhijian Ma, Mengyue Yang, Haoxuan Li, Yisen Wang, Haifeng Zhang, and Jun Wang. Skill-pro: Learning reusable skills from experience via non-parametric ppo for llm agents. *arXiv preprint arXiv:2602.01869*, 2026.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

Alexander Nikulin, Vladislav Kurenkov, Ilya Zisman, Artem Agarkov, Viacheslav Sinii, and Sergey Kolesnikov. Xland-minigrid: Scalable meta-reinforcement learning environments in jax. *Advances in Neural Information Processing Systems*, 37:43809–43835, 2024.

R Osazuwa Ness, Ricardo Cannizzaro, Yunshu Wu, and Lars Kunze. Multiverse mechanica: a testbed for learning game mechanics via counterfactual worlds. International Conference on Learning Representations, 2026.

Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Adam Summerville, Sam Snodgrass, Matthew Guzdial, Christoffer Holmgård, Amy K Hoover, Aaron Isaksen, Andy Nealen, and Julian Togelius. Procedural content generation via machine learning (pcgml). *IEEE Transactions on Games*, 10(3):257–270, 2018.

Julian Togelius, Georgios N Yannakakis, Kenneth O Stanley, and Cameron Browne. Search-based procedural content generation: A taxonomy and survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(3):172–186, 2011.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. *URL https://arxiv. org/abs/2305.16291*, 2(11), 2023.

Jane X Wang, Michael King, Nicolas Porcel, Zeb Kurth-Nelson, Tina Zhu, Charlie Deck, Peter Choy, Mary Cassin, Malcolm Reynolds, Francis Song, et al. Alchemy: A benchmark and analysis toolkit for meta-reinforcement learning agents. *arXiv preprint arXiv:2102.02926*, 2021.

Zefeng Wang, Minxi Yan, Jinhe Bi, Sikuan Yan, Volker Tresp, and Yunpu Ma. Metaskill-evolve: Recursive self-improvement of llm agents via two-timescale meta-skill evolution. *arXiv preprint arXiv:2607.05297*, 2026.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: open-ended evolution of self-improving agents. In *International Conference on Learning Representations*, volume 2026, pp. 104223–104294, 2026.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Tailin Zhou. Hierarchical self-improvement: A framework for task-specific evolvable agent harnesses, 2026. URL `https://arxiv.org/abs/2608.08466`.

A METRIC OPERATIONALIZATION

A.1 SHARED INPUTS, ADAPTERS, AND EXECUTION ORDER

For an artifact g, let $\mathcal{F}_g^{\mathrm{act}} = \{f_1, \ldots, f_{K_g}\}$ be the executable mechanics that can be activated under the evaluation scenarios. This set is distinct from the explanatory mechanism set $\widehat{\mathcal{H}}_g$ induced from observations. A trajectory is $\tau = (s_0, a_0, s_1, \ldots, s_{T_\tau})$, where T_τ is its action length. Each environment adapter deterministically provides: (i) the executable-mechanic representation, (ii) the event encoder ψ, (iii) the neutralizing intervention $\mathrm{do}(f_i = 0)$, and (iv) the mechanic-level strategy encoder. The compression learner sees only the preregistered event interface and never reads hidden transition code.

The general adapter interface freezes these objects together with the public action and observation spaces. The environment-specific contracts in Table 8 are implemented by the external extension runner, while the CreativeGame driver executes GameSpec rules and neutralizes designated mechanic IDs without removing public actions or state variables. Successful execution does not establish useful calibration. Table 6 reports varying TMCP ranges, but not method identities at the endpoints or paired uncertainty.

Within the pilot, the evaluator collects past and development transfer evidence, induces $\widehat{\mathcal{H}}_g$ from past records only, freezes it, and computes the five-metric profile. Collection also produces the fixed intervention and solver traces used below. More report-stage resamples do not create a held-out split. Future confirmation requires untouched briefs and unopened test scenarios, with no prompt, threshold, or policy update after their results are revealed.

A.2 RETROSPECTIVE AND TRANSFER COMPRESSION

The event encoder maps each transition to $y_h = \psi(s_h, a_h, s_{h+1})$, with context $x_h = (s_h, a_h)$. In this appendix, η collects the frozen smoothing, background-code, and induced-rule parameters. For discrete events, the fixed predictor in Equation 2 uses the smoothed distribution

$$p_\eta(y \mid x, \mathcal{H}) = (1 - \varepsilon_{\mathrm{pred}}) q_{\mathcal{H}}(y \mid x) + \varepsilon_{\mathrm{pred}} p_0(y \mid x), \tag{8}$$

where $q_{\mathcal{H}}$ is uniform over the union of events permitted by applicable rules; the background p_0 is independent of x and is used alone when no rule applies. The generation-time alphabet is the set of *all finite byte strings*, including strings that are not valid UTF-8. For byte sequence $b = \mathrm{UTF8}(y)$ of length n bytes, the frozen background is

$$p_0(b) = 2^{-(n+1)} 256^{-n} = 2^{-(9n+1)}. \tag{9}$$

This samples a geometric length starting at zero and independent uniform bytes. It is normalized because

$$\sum_b p_0(b) = \sum_{n=0}^{\infty} 256^n 2^{-(9n+1)} = \sum_{n=0}^{\infty} 2^{-(n+1)} = 1.$$

UTF-8 encoded event strings are an injective subset of this fixed alphabet; no observed-vocabulary fitting or renormalization on test events is performed. The implied background code uses $9n + 1$ bits. Equation 8 remains normalized because the rule distribution and background are normalized on the same alphabet. The score includes numeric deltas already serialized in the event, with no second Gaussian channel. This is a valid reference code, not a claim of optimal compression or representation invariance.

The explanatory set is $\widehat{\mathcal{H}}_g = \mathrm{Induce}(D_{\mathrm{past}})$. Let L_{rule} be its canonical rule-description length and B_M the shared explanation budget; the evaluator requires $L_{\mathrm{rule}}(\widehat{\mathcal{H}}_g) \le B_M$. The bounded reporting scorer also estimates action-conditioned event counts $c_a(y)$ and totals $C_a = \sum_y c_a(y)$ from D_{past} only. With $\varepsilon_{\mathrm{sham}} = .05$, its state-blind sham is

$$q_0(y \mid a) = \begin{cases} (1 - \varepsilon_{\mathrm{sham}}) c_a(y)/C_a + \varepsilon_{\mathrm{sham}} p_0(y), & C_a > 0, \\ p_0(y), & C_a = 0, \end{cases} \tag{10}$$

where an unobserved event has $c_a(y) = 0$ and therefore retains positive byte-tail probability. The mechanism predictor uses

$$q_H(y \mid s, a) = (1 - \varepsilon_{\mathrm{pred}}) q_{\widehat{\mathcal{H}}_g}(y \mid s, a) + \varepsilon_{\mathrm{pred}} q_0(y \mid a), \tag{11}$$

with $\varepsilon_{\text{pred}} = .01$; when no explanatory rule applies, it reverts exactly to q_0. Both q_0 and $\widehat{\mathcal{H}}_g$ are frozen before transfer scoring. Define

$$L_0(D) = -\sum_{(s,a,y)\in D} \log_2 q_0(y \mid a), \qquad L_H(D) = -\sum_{(s,a,y)\in D} \log_2 q_H(y \mid s, a).$$

The complete bounded-score calculations are

$$\text{RMCP}(g) = \frac{L_0(D_{\text{past}})}{L_0(D_{\text{past}}) + L_H(D_{\text{past}}) + \varepsilon_L}, \tag{12}$$

$$\text{TMCP}(g) = \frac{L_0(D_{\text{init}})}{L_0(D_{\text{init}}) + L_H(D_{\text{init}}) + \varepsilon_L}. \tag{13}$$

The report uses $\varepsilon_L = 10^{-12}$ only to guard division; all observed code lengths are positive. Ignoring this below-precision guard, .5 is $L_0 = L_H$, and TMCP $> .5$ iff $L_H < L_0$. The ratio is monotone in L_H/L_0 but is not a linear bit-effect scale.

For audit, the bounded scorer also retains raw bit savings per transition,

$$\text{RMCP}_{\text{raw}}(g) = \frac{L_0(D_{\text{past}}) - L_H(D_{\text{past}})}{|D_{\text{past}}|}, \tag{14}$$

$$\text{TMCP}_{\text{raw}}(g) = \frac{L_0(D_{\text{init}}) - L_H(D_{\text{init}})}{|D_{\text{init}}|}. \tag{15}$$

Mean UTF-8 event lengths accompany these raw values because coding length can confound cross-artifact comparisons. The generation-time score instead used the byte prior directly as its background and reported

$$\text{nTMCP}_{\text{dev}}(g) = \frac{L_{0,\text{dev}}(D_{\text{init}}) - L_{H,\text{dev}}(D_{\text{init}})}{L_{0,\text{dev}}(D_{\text{init}}) + \varepsilon_L}, \tag{16}$$

with the analogous nRMCP definition on D_{past}. This generation-time value is not pooled with the bounded rescore.

D_{init} is the full bounded collection of initial-state probe records and is not truncated according to the number of rule-removal records. Rule-removal scores use the corresponding frozen predictors on their own records and are reported as robustness, not added to TMCP. The retained pooled-family value is a compatibility diagnostic only. Missing families are not assigned zero. Development records remain available to guide the next design; increasing report-stage resamples does not turn them into an unopened test.

A.3 Original-score audit

Table 7 preserves the originally reported endpoint rather than silently replacing it. Its method ranking differs from the calibrated bounded rescore because the original byte prior rewards prediction of long serialized events. Under the original score, the main-minus-fixed, main-minus-no-memory, main-minus-raw-memory, and main-minus-Skill-Pro contrasts were respectively $-.026[-.189, .105]$, $+.015[-.132, .190]$, $-.029[-.160, .126]$, and $-.152[-.323, -.009]$. These are historical development results, not additional confirmatory tests.

Table 7: Original mean nTMCP. Each CreativeGame method attempted 15 executions across five case IDs and three repeats; quality means use applicable case means, which may number fewer than five. This audit table records the prespecified normalization; the main results use the explicitly post-hoc bounded rescore.

Method	$\text{nTMCP}_{\text{dev}}$	Method	$\text{nTMCP}_{\text{dev}}$
Independent	.736	Feedback-only	.743
Diagnostic, no memory	.664	C–M–S, fixed revision	.751
Diagnostic, raw memory	.712	Ours (Diagnostic C–M–S)	.675
Ours + slow-policy update	.661	Self-Refine / Reflexion	.750
Skill-Pro adaptation	.832	Random-retrieval C–M–S	.686
Unverified C–M–S	.686		

A.4 EFFECTIVE CAUSAL INTERACTION

For every ordered pair $i \neq j$, the pilot replays matched actions from the original winning trajectory in g and $g^{-i} = \mathrm{do}(f_i = 0)$ at the same seeds. It does not perform closed-loop replanning to estimate action values. Let App_j and Act_j be the applicability and activation indicators of target rule f_j, and let Y_j be its recorded outcome. Outcome and occupancy histograms are projected to variables read or written by f_j, using all recorded transitions. Define

$$d^{\mathrm{app}}_{i\to j} = |\Pr_g(\mathrm{App}_j) - \Pr_{g^{-i}}(\mathrm{App}_j)|, \tag{17}$$

$$d^{\mathrm{act}}_{i\to j} = |\Pr_g(\mathrm{Act}_j) - \Pr_{g^{-i}}(\mathrm{Act}_j)|, \tag{18}$$

$$d^{\mathrm{out}}_{i\to j} = \mathrm{JSD}\big(P_g(Y_j), P_{g^{-i}}(Y_j)\big), \tag{19}$$

$$d^{\mathrm{reach}}_{i\to j} = \mathrm{JSD}\Big(\rho^j_g, \rho^j_{g^{-i}}\Big). \tag{20}$$

Here ρ^j is the occupancy distribution over target-local states, and base-2 Jensen–Shannon divergence lies in $[0, 1]$. With per-step reward r_h, fixed return bounds $J_{\min}$ and $J_{\max}$, and numerical guard ε_J, the pilot uses a trajectory-reward proxy rather than a counterfactual action-value estimate:

$$v_j(\tau) = \mathrm{clip}_{[-1,1]}\left(\sum_{h: f_j \text{ activated at } h} r_h\right), \qquad d^{\mathrm{value}}_{i\to j} = \min\left\{1, \frac{|\overline{v}_j(g) - \overline{v}_j(g^{-i})|}{J_{\max} - J_{\min} + \varepsilon_J}\right\}. \tag{21}$$

The bar denotes the mean across matched recorded trajectories. This proxy attributes reward at activation steps, not the marginal causal value of an action.

The directional pairwise effect is the fixed equal-weight mean

$$\Delta_{i\to j} = \frac{1}{5}\left(d^{\mathrm{app}}_{i\to j} + d^{\mathrm{act}}_{i\to j} + d^{\mathrm{out}}_{i\to j} + d^{\mathrm{reach}}_{i\to j} + d^{\mathrm{value}}_{i\to j}\right). \tag{22}$$

The pilot fixes $\varepsilon_I = 0.05$. A threshold calibrated from the 95th percentile of per-artifact maximum effects under sham interventions is a planned replacement, not a completed calibration. With paired bootstrap resampling over recorded scenario seeds, let $\mathrm{LCB}_{0.95}$ denote the lower endpoint of the resulting 95% interval and define

$$\chi_{i\to j} = \mathbb{I}[\Delta_{i\to j} > \varepsilon_I \ \wedge\ \mathrm{LCB}_{0.95}(\Delta_{i\to j}) > 0]. \tag{23}$$

Substitution of Equation 23 into Equation 6 gives the final game-level score. This normalization counts both the density and strength of directed interactions; $I_{\mathrm{eff}} = 0$ when $K_g < 2$.

A.5 COMPOSITIONAL NOVELTY

The effective mechanic graph is

$$G^{\mathrm{eff}}_g = (\mathcal{F}^{\mathrm{act}}_g, E^{\mathrm{eff}}_g), \qquad E^{\mathrm{eff}}_g = \{(f_i, f_j) : \chi_{i\to j} = 1\}. \tag{24}$$

Node and edge labels are canonical functional roles; colors, surface names, natural-language wording, and environment-specific identifiers are discarded. Let $\mathcal{T}_{\mathrm{succ}}(g)$ be the collected successful trajectories. For $h \in \{2, 3, 4\}$, let $\mathcal{Z}_h(g)$ contain connected typed motifs with h nodes that meet both support criteria

$$\mathrm{supp}(z) = \frac{\#\{\tau \in \mathcal{T}_{\mathrm{succ}}(g) : z \text{ is activated in } \tau\}}{|\mathcal{T}_{\mathrm{succ}}(g)|} \geq s_{\min}, \qquad \mathrm{seeds}(z) \geq r_{\min}. \tag{25}$$

where $\mathrm{seeds}(z)$ counts distinct scenario seeds supporting motif z. For a frozen external reference library $\mathcal{R}_h$, typed graph-edit distance is

$$d_{\mathrm{GED}}(z, r) = \frac{\mathrm{GED}(z, r)}{|V_z| + |V_r| + |E_z| + |E_r|} \in [0, 1], \tag{26}$$

where V_z and E_z are the vertex and edge sets of motif z; unit insertion, deletion, and type-mismatch costs are used. A motif's novelty and the size-specific aggregate are

$$n(z) = \frac{1}{k_{\mathrm{NN}}} \sum_{r \in \mathrm{kNN}_{k_{\mathrm{NN}}}(z, \mathcal{R}_h)} d_{\mathrm{GED}}(z, r), \tag{27}$$

$$N_h(g) = \begin{cases} \dfrac{\sum_{z \in \mathcal{Z}_h(g)} \mathrm{supp}(z) n(z)}{\sum_{z \in \mathcal{Z}_h(g)} \mathrm{supp}(z)}, & \mathcal{Z}_h(g) \neq \varnothing, \\ 0, & \mathcal{Z}_h(g) = \varnothing. \end{cases} \tag{28}$$

The complete score is

$$N_{\mathrm{comp}}(g) = \frac{1}{3} \sum_{h=2}^{4} N_h(g). \tag{29}$$

Thus $N_{\mathrm{comp}} \in [0, 1]$, and an artifact cannot obtain a high score from a single novel pair while lacking higher-order compositions. If no successful trajectory exists, the artifact is recorded as a solvability failure and N_{comp} is not estimated rather than assigned an artificial zero. The reference libraries exclude all test artifacts, method archives, and intermediate test-time generations.

A.6 Strategic richness

Let the fixed solver ensemble be $\mathcal{P} = \{A_1, \ldots, A_P\}$, and let $J(\tau)$ be trajectory return. After normalizing return as

$$\bar{J}(\tau) = \frac{J(\tau) - J_{\min}}{J_{\max} - J_{\min} + \varepsilon_J}, \tag{30}$$

let $\bar{J}^\star$ be the largest normalized return found for g and retain the successful near-optimal set

$$\mathcal{T}_\delta(g) = \{\tau : \mathrm{Success}(\tau) = 1,\ \bar{J}(\tau) \geq \bar{J}^\star - \delta\}. \tag{31}$$

The pilot strategy encoder uses the recorded mechanic-activation sequence, filters explicitly ignored labels such as no-op and cosmetic markers, and removes consecutive duplicate activations; it does not prove that every remaining activation is strategically consequential. We denote this signature by $\varphi(\tau)$, distinct from the event encoder ψ, and denote its set of activated mechanic labels by $F(\tau)$. With Lev denoting Levenshtein edit distance, two retained trajectories have

$$d_{\mathrm{seq}}(\tau_i, \tau_j) = \frac{\mathrm{Lev}(\varphi(\tau_i), \varphi(\tau_j))}{\max(|\varphi(\tau_i)|, |\varphi(\tau_j)|)}, \tag{32}$$

$$d_{\mathrm{set}}(\tau_i, \tau_j) = 1 - \frac{|F(\tau_i) \cap F(\tau_j)|}{|F(\tau_i) \cup F(\tau_j)|}, \tag{33}$$

$$d_{\mathrm{strategy}}(\tau_i, \tau_j) = \tfrac{1}{2} d_{\mathrm{seq}} + \tfrac{1}{2} d_{\mathrm{set}}. \tag{34}$$

Complete-linkage agglomerative clustering with fixed pilot cut η_{str} produces clusters $C_1, \ldots, C_C$. Let $n_{p,c}$ be the number of retained trajectories from solver A_p in cluster c, and let $\mathcal{P}_g^+$ contain solvers with at least one retained trajectory. To prevent a high-yield solver from dominating,

$$p_c = \frac{1}{|\mathcal{P}_g^+|} \sum_{p \in \mathcal{P}_g^+} \frac{n_{p,c}}{\sum_{c'} n_{p,c'}}, \qquad S_{\mathrm{eff}}(g) = \exp\left(- \sum_{c=1}^{C} p_c \log p_c \right). \tag{35}$$

When more than B_{succ} successful trajectories are available, we repeat stratified rarefaction without replacement R_{rare} times and report

$$\widehat{S}_{\mathrm{eff}}(g) = \frac{1}{R_{\mathrm{rare}}} \sum_{r=1}^{R_{\mathrm{rare}}} S_{\mathrm{eff}}^{(r)}(g). \tag{36}$$

If fewer are available, all retained trajectories are used and their count is reported. No successful trajectory is a solvability failure; an environment with no meaningful goal-conditioned solution classes receives "not applicable", not zero.

A.7 Default calibration and cross-environment aggregation

The current pilot uses $k_{\mathrm{NN}} = 5$, $s_{\min} = 0.05$, $r_{\min} = 3$ recorded seeds, $\varepsilon_{\mathrm{pred}} = 0.01$, $\delta = 0.10$, $B_{\mathrm{succ}} = 100$, $\eta_{\mathrm{str}} = 0.2$, $J_{\min} = -1$, $J_{\max} = 1$, and $\varepsilon_L = \varepsilon_J = 10^{-12}$. Search uses 200 paired-bootstrap resamples and 20 rarefaction repeats; report-stage scoring uses 10,000 and 100. The interaction threshold is a fixed default, not a fitted validation estimate. The generation-time score uses joint encoded events and no continuous residual variance. Three deterministic seed replays do not constitute three independent realizations, even when they satisfy the current support-count gate.

For future cross-environment analysis, raw values should not be averaged directly. Let $F^{\mathrm{val}}_{e,m}$ be a separately constructed, frozen validation cumulative distribution function for metric m in environment e, and let $m_e(g)$ be the raw metric value of artifact g. Once such calibration data exist, each test artifact can be converted to

$$\widetilde{m}_e(g) = F^{\mathrm{val}}_{e,m}(m_e(g)) \in [0, 1]. \tag{37}$$

For the set $\mathcal{A}_m$ of environments in which m is applicable, the cross-environment summary is

$$\mathrm{CrossEnv}(m) = \frac{1}{|\mathcal{A}_m|} \sum_{e \in \mathcal{A}_m} \mathrm{Median}_{g \in e}\, \widetilde{m}_e(g). \tag{38}$$

The five metrics remain separate. Equation 38 is not computed because the updated environment-level ranges do not supply the required frozen validation distributions or method-resolved applicable profiles. A reported upper endpoint of 1.00 alone does not establish saturation; unrounded distributions are needed to assess ceiling effects. Per-environment sample counts, unsupported metrics, and calibration constants must accompany any future aggregate.

Table 8: Implemented environment-specific adapter contracts used in the development matrix. Successful execution does not by itself establish a discriminative comparison.

Environment	Mechanism unit	Principal transfer probes
Multiverse Mechanica	Ground-truth causal game rule	Counterfactual worlds and unseen rule compositions
Symbolic DM Alchemy	Latent potion/stone transformation	New symbols, stones, chemistries, and compositions
XLand-MiniGrid	Rule–object–goal relation	Held-out layouts, goals, and rule combinations
Craftax	Resource–recipe–capability dependency	New seeds, biomes, progression contexts, and compositions
I-PHYRE	Support/collision/intervention relation	New geometry and counterfactual object interventions

B CreativeGame Pilot Implementation Contract

The generation and selection run and the offline bounded rescore retain their exact evaluator and control identifiers in the stored manifests. Manifests freeze method, prompt, configuration, source, case, and repeat identifiers. The rescore reads the selected stored GameSpec, reruns the deterministic evidence collector, and writes a linked result with `candidate_reselected=false` and `llm_calls=0`; it never overwrites the generation ledger. Standard comparison cells start from empty memory; methods within an environment–case–repeat tuple share the same environment seed, while the three repeats use distinct seeds and model requests. Only the explicit reuse study carries assigned memory across ordered epochs, with separate state per curriculum, repeat, and arm. No game-code or asset-generation stage is invoked. Historical scoring paths remain available for replay and must not be pooled with bounded scores.

B.1 Past observations and conditional induction

The bounded breadth-first solver uses at most 100,000 explored states per solve and the GameSpec episode horizon. The fixed scenario seeds are 1709, 2713, and 3907. Past evidence begins with the successful forward-order trajectory at the first successful seed. For the seed-1709 witness, at

prefix depths zero through $\min(8, T_\tau) - 1$, the collector tries the first eight enumerated actions. Each branch replays the prefix, executes one action, and contributes its final transition if that transition exists. Failures are recorded rather than filtered by score. This adds at most 64 event records, although replaying prefixes requires additional transitions. Repeated records are retained and are not independent samples.

The inducer receives only these past records. It processes action groups and state fields in sorted order, excluding dotted field names, missing values, and list/dictionary-valued fields. Numeric predicates use midpoints of sorted distinct observed values, with complementary $\leq$ and $>$ branches; other scalar predicates use equality and inequality. At most the first 32 tests per field are considered. Let R be the records at a node, R_+ and R_- the two children, and $\widehat{p}_R(y)$ the empirical frequency of event y in R. The split gain is

$$H(R) - \frac{|R_+|}{|R|} H(R_+) - \frac{|R_-|}{|R|} H(R_-), \qquad H(R) = -\sum_y \widehat{p}_R(y) \log_2 \widehat{p}_R(y). \tag{39}$$

Both children must contain at least two records, and the greatest gain must exceed 0.1 bits. Strict comparison retains the first encountered split on ties. Recursion stops at depth two or when no split qualifies. Each leaf returns the conjunction of its path predicates, its action trigger, and the set of observed event strings. Continuous-field means are retained for schema compatibility but are not scored during generation. A leaf may retain multiple outcomes. The fixed rule budget is checked after induction; the procedure is not an optimizer of a penalized MDL objective.

B.2 Event scoring and transfer construction

Let $b_h = 9|\,\mathrm{UTF8}(y_h)| + 1$ and let U_h be the union of events permitted by applicable rules. The generation-time joint-event code length is

$$\ell_h = \begin{cases} b_h, & \text{no rule applies,} \\ b_h - \log_2 \varepsilon_{\mathrm{pred}}, & y_h \notin U_h, \\ -\log_2\!\big((1-\varepsilon_{\mathrm{pred}})/|U_h| + \varepsilon_{\mathrm{pred}} 2^{-b_h}\big), & y_h \in U_h. \end{cases} \tag{40}$$

The last branch uses base-2 log-sum-exp; long events are never replaced by a probability floor. Thus an applicable rule that excludes the observation pays $-\log_2 \varepsilon_{\mathrm{pred}}$ additional bits, whereas an inapplicable rule reverts to the background. The bounded reporting score uses the same three logical branches but replaces 2^{-b_h} by the frozen $q_0(y_h \mid a_h)$ in Equation 10. No Gaussian score is added for numeric changes already present in the event. The predictor digest includes its scoring specification, smoothing probabilities, byte prior, past action/event counts, and disabled residual channel. Scores from the generation-time and bounded evaluators are not pooled.

The initial-state probe construction is specified in Section 3.3; the rule-removal probes below use a separate intervention family.

For rule-removal probes, the first and last active mechanic IDs in sorted order are neutralized separately, deduplicating when there is only one ID. The same original witness is replayed at seed 3907, again with horizon 24 and failures retained. Intervention identity is not supplied as an additional explanatory-rule condition. These probes therefore measure prediction under an unannounced transition change, not invariant transfer.

Events are concatenated within each family and retained in full under the fixed probe limits; no cross-family truncation is applied. Retrospective scores use all past records, and transfer scores use only initial-state records. Rule-removal scores and win/loss diagnostics are separate. A missing initial-state family makes TMCP unavailable even when rule-removal evidence exists. Solver failure or absent evidence is flagged, never repaired by invented observations.

B.3 Diagnosis, selection, and memory

For a complete-profile candidate, let $r_{\mathrm{dev}} = \mathrm{nRMCP}_{\mathrm{dev}}(g)$ and $t_{\mathrm{dev}} = \mathrm{nTMCP}_{\mathrm{dev}}(g)$, with t_{dev} restricted to initial-state transfer. These are the generation-time scores in Equation 5, not the post-hoc bounded RMCP and TMCP. A missing initial-state family cannot be replaced by a rule-removal

aggregate; it fails the complete-profile gate. The implemented deficit scores are

$$d_{\text{transfer}} = \max\{0, (r_{\text{dev}} - t_{\text{dev}})/\max(|r_{\text{dev}}|, 1)\}, \tag{41}$$

$$d_{\text{strategy}} = 1/\max(S_{\text{eff}}, 1), \tag{42}$$

$$d_{\text{interaction}} = 1 - I_{\text{eff}}, \qquad d_{\text{novelty}} = 1 - N_{\text{comp}}, \tag{43}$$

$$d_{\text{compression}} = \begin{cases} 1, & r_{\text{dev}} \le 0, \\ 1/(1 + r_{\text{dev}}), & r_{\text{dev}} > 0. \end{cases} \tag{44}$$

Operators within 0.25 of the maximum deficit are eligible; the first eligible operator in the current priority list is selected. The default order is transfer, strategy, interaction, novelty, compression. The diagnosis returns this edit operator together with target GameSpec mechanic IDs for the next revision request; the model edits the identified rules. These heuristic scores select a revision instruction, not a combined quality metric. Preservation instructions do not implement a separate route-preservation verifier.

All methods share the final selector. For each hard-valid candidate and metric, its rank is the fraction of hard-valid candidates with a strictly lower value. Pareto-dominated candidates are excluded. The remaining candidates' five ranks are sorted ascending and compared lexicographically; larger is better. Ties prefer a shorter canonical artifact, then an earlier candidate. If no candidate is hard-valid, the fallback prioritizes design validity, then GameSpec validity, then generation order, without reclassifying the result as successful.

Memory capacities, retrieval, and eviction follow Section 3.2.

B.4 Optional slow-policy extension and budgets

The main arm keeps its policy fixed. Only `diagnostic_cms_slow` may change diagnosis priority, retrieval count, edit scope, and verification instructions. This narrow policy search is an extension, not unrestricted recursive improvement. No evaluator, model weight, resource limit, or final selector is editable.

Table 9: Current per-method, per-game pilot ceilings. Output caps are included in the shared 35,000-token input/output allowance, not additional allocations.

Quantity	Ceiling
Total input and output model tokens	35,000
All API calls / candidate calls	8 / 4
Initial design output	9,000
Valid-design revision output	3,500
Invalid-design repair output	6,000
Policy proposal output	800
Validation output per policy (both fixtures)	3,000
Retrieved memory input	800
Canonical artifact / explanatory-rule budget (bits)	262,144 / 262,144

In the slow extension only, a trial is considered on every third game after two candidates, only when at least 20,000 tokens remain and the call allowance reserves validation and a subsequent task revision. One proposal call is followed, for a changed policy, by two validation calls: one for the current policy and one for the proposed policy, each returning revisions to both fixed puzzles. An identical policy skips validation. The proposal may change only priority, retrieval count, edit scope, and verification instructions.

For each fixture, the executable gate preserves the objectives, original action IDs, and existing state-variable definitions including initial values. It does not prove semantic equivalence of every revised action or enforce every natural-language preservation instruction. With a valid current artifact, the per-metric tolerance vector in the generation-time order $(N_{\text{comp}}, \text{nRMCP}_{\text{dev}}, \text{nTMCP}_{\text{dev}}, S_{\text{eff}}, I_{\text{eff}})$ is

$$(0.02,\ 0.2,\ 0.2,\ 0.1,\ 0.02).$$

The proposed artifact must be hard-valid, remain above the current value minus tolerance on every metric, and exceed the current value plus tolerance on at least one metric. A valid proposal also

passes against an invalid current artifact. Both fixtures must pass before the policy changes. A failed, skipped, or unchanged trial is not counted as an accepted update.

Model calls stop at token, call, or candidate limits, and can stop after three consecutive hard-valid candidates with identical executable GameSpecs. Best-of-N uses independent full proposals rather than revision feedback. Completed low-quality attempts remain in the outcome ledger. Infrastructure and quota errors halt the batch and enter a separate error ledger; they are neither successful results nor game-validity failures. Checkpoint replay reuses a response only for a matching request, preventing duplicate successful calls during resume.

B.5 Interaction, strategy, and remaining validation

Target-local interaction histograms project each transition to variables read or written by the target mechanic. The reward proxy sums rewards only on target-activation steps and clips the total to $[-1, 1]$. No replanned action-value estimate is computed. The shared solver ensemble uses forward, reverse, rotated, and three fixed shuffled action orders (19, 37, 71), together with runs excluding each of the first three action IDs. Solver traces are bounded; the resulting strategic richness measures discovered, not exhaustive, solution classes.

The reference motif basis contains five hand-authored motifs at each size two, three, and four. It is frozen across methods but is not evidence that a design is novel relative to the wider game literature. Deterministic evaluator replays are cached and are not independent experimental replicates; the model repeat index in the development manifest is separate from those replays. The past-fitted sham is implemented and unit-tested, but it was selected after observing the original scores. A prospectively frozen sham, a corpus-derived novelty basis, more independent cases, and untouched briefs/scenarios remain requirements for broader confirmatory claims. All seven literature-inspired adaptations are supplementary proxies, not official-repository reproductions.

C Component Controls and Submission Evidence Gates

C.1 Evaluation scope and analysis units

The initial study completed 630 runs in six runnable environments; its ledger retains 735 cell IDs, including 105 excluded at preflight. The archived extension comprised 165 CreativeGame runs, 300 reuse executions, and 225 external-adapter runs, originally using five case IDs and three repeats per method. The updated Table 6 reports 165 CreativeGame attempts and 245 external attempts: 45 in Multiverse Mechanica and 50 in each other external environment. These updated totals do not specify allocation by method, case, or repeat; they must not be interpreted as 15 attempts per environment–method pair. Each reuse arm has five curricula of five ordered cases, with three complete repetitions per curriculum. Its 75 game executions therefore represent 15 complete curriculum trajectories. Case IDs are the independent units within each case–method comparison; curricula are the independent units for reuse. Repeats and ordered cases are nested within those units, and paired intervals resample at the corresponding level. Quality estimates use only applicable profiles, so their effective denominators can be smaller than the planned attempts. These are development-sample uncertainty summaries, not population significance tests. Reporting uncertainty rather than point estimates alone follows guidance for limited-run RL evaluation (Agarwal et al., 2021).

The archived 690-run extension ledger records 7,876,242 input tokens and 5,318,085 output tokens, or 13,194,327 model tokens across 1,954 logged model calls. At the rates used in the run ledger, this is approximately $99.46. This historical usage total does not establish the cost of the updated Table 6. It excludes preflight probes and a disconnected request absent from successful result rows. CreativeGame paired intervals use 2,000 hierarchical bootstrap resamples of cases and nested repeats; the reuse analysis preserves curriculum and repeat boundaries.

C.2 Research-question contrasts and interpretation

RQ1: Diagnosis under identical feedback. Compare diagnostic C–M–S (the main method) with C–M–S plus a fixed generic revision instruction. Both see the same parent, probe feedback, and memory cap; only the main method receives an explicit diagnosis. Compare diagnostic no-memory

with feedback-only revision as a second contrast. Agreement would support a diagnosis effect; a gain confined to privileged feedback would not.

RQ2: Structured memory beyond additional context. Compare the main method with diagnostic no-memory and diagnostic raw-memory. Raw memory contains verified source trajectories, not unrelated context, and shares the 800-token retrieval ceiling. Targeted controls additionally randomize C–M–S retrieval order or admit structured entries without execution-gated witnesses. A separate curriculum study compares persistent C–M–S memory with reset C–M–S, persistent raw memory, and no memory at matched epochs. These contrasts test the C–M–S package, relevance, verification, and state persistence without treating additional context as equivalent to reusable knowledge.

RQ3: Evaluation and transfer controls. Compare the bounded rescore with the original development score, then inspect each external adapter separately for metric ties and ceilings. Initial-state prediction remains separate from rule-removal robustness. Behavior-preserving rewrites, surface renaming, inactive-rule insertion, and fixed size/length controls would further test representation sensitivity. Untouched briefs, unopened scenarios, and independent assessment remain necessary for generalization claims.

RQ4: Adaptation value and cost. Compare the main method with its slow-policy extension on observed quality and realized token use. The present panel is descriptive; establishing a policy benefit would also require accepted-change counts, charged validation costs, subsequent-brief outcomes, and order sensitivity. A lower ceiling is not a measured saving, and success on policy fixtures is not the endpoint.

C.3 Control implementation and evidence gates

Table 10: Implemented controls. Diagnosis denotes the explicit rule selecting a targeted edit, not access to additional evaluation data. Only the slow extension can update bounded policy fields.

Arm	Feedback	Memory	Diagnosis	Slow update
Independent sampling	No	None	No	No
Feedback-only revision	Yes	None	No	No
Diagnostic no-memory	Yes	None	Yes	No
C–M–S, fixed generic revision	Yes	C–M–S	No	No
Diagnostic raw-memory	Yes	Raw traces	Yes	No
Ours (Diagnostic C–M–S)	Yes	C–M–S	Yes	No
Ours + slow-policy update	Yes	C–M–S	Yes	Yes

Prompt isolation. All iterative controls receive the same complete-profile feedback and compact transfer diagnostics through the same base prompt. The comparison arm name is removed before calling the model. Explicit diagnosis is present only in diagnosis-enabled arms; memory is present only in the assigned representation. The independent arm receives neither a parent nor feedback. With fresh empty memory, all seven arms have identical initial prompts. Feedback-only and diagnostic no-memory retain no C–M–S or raw-experience archive.

Memory replacement. The C–M–S arms use the capacities and retrieval procedure above, including at most two archive-profile entries inside the 800-token cap. The raw-memory arm stores at most 24 source-winning trajectories, canonically deduplicated by GameSpec digest. It orders them by the same lexical relevance principle and retrieves at most the configured number of source entries. Complete raw transitions are added only while the serialized memory fits the input cap; a retrieved prefix is not claimed to be a complete successful route. Raw retrieval carries source provenance but no induced C–M–S entries or metric archive. Both representations exclude exact current-parent duplicates. The cap is equal, not necessarily the realized token count or information content.

Targeted memory ablations. The random-retrieval arm keeps the verified C–M–S representation, entry count, and 800-token cap but shuffles the retrieved entries with the run seed before prompting. It tests relevance without replacing structured memory by unrelated text. The unverified arm retains typed components, mechanisms, and skills without requiring a replayed execution witness at admission. It tests the execution gate while preserving the structured representation. Both arms share the main method's diagnosis, evaluator, selector, cases, and seeds.

Cross-case reuse contract. Each reuse arm completes 15 curriculum trajectories, totaling 75 game executions across five frozen curricula of five ordered cases. Within a curriculum, a named state carrier remains fixed while genre, trigger, and feedback vary. The four arms are persistent C–M–S, reset C–M–S, persistent raw memory, and diagnostic no-memory. There are three independently requested runs per curriculum and arm. Persistent state flows only from epoch e to $e + 1$ within the same curriculum, repeat, and arm; reset C–M–S reconstructs empty state before every epoch. Reports treat curriculum as the independent unit, repeat as nested, and epoch as an ordered within-run factor. The primary reuse contrast averages epochs two through five and pairs arms by curriculum, repeat, and epoch.

Development run contract. The runner defaults to planning and invokes no API without an explicit execution flag. Each cell has a separate response cache and one shared append-only outcome ledger. Checkpoint resumes require matching source, configuration, and case manifests and request digests. The report groups by environment, method, case, and repeat; validates the execution-contract digest; exposes failed, missing, unavailable, and not-applicable values; and does not interpret quota or connection errors as invalid games. Method summaries first average repeats within case, and paired intervals resample cases before repeats. The offline combiner rejects incomplete source ledgers. The bounded rescore script consumes only stored artifacts, preserves original source-cell identifiers, emits separate manifests and 100 paired metric contrasts, and makes no model calls.

Conditions for a stronger claim. Normalization and unit tests address implementation correctness, not the adequacy of all metrics. Before confirmation, the evaluation must pass surface-renaming, equivalent-rewrite, inactive-rule, and representation-length checks, and report failures rather than change the scorer after viewing method rankings. A sham-calibrated interaction threshold and an independent motif corpus are required for stronger interaction and novelty interpretations. External assessors must score brief fidelity and meaningful choices without seeing method identity or optimizer scores. These assessments and unopened test scenarios are not implemented by merely resampling development traces.

Table 11: Evidence required for the central claims. Each negative outcome limits the corresponding claim; no claim requires winning every metric.

Claim	Identifying comparison	Result that would weaken it
Explicit diagnosis helps	Main vs. C–M–S fixed generic revision, identical feedback	Gain disappears when feedback is equal
Structured memory helps	Main vs. diagnostic raw-memory and no-memory	Raw traces match the gain at comparable cost
Transfer improves	Unopened initial states/briefs, normalized and raw scores	Gain exists only on development probes or length-sensitive coding
Slow adaptation adds value	Main vs. slow extension, charged validation costs	Fixture improvement does not carry to later briefs

The frozen extension activates the weakening condition for the broad transfer claim. In the bounded rescore, diagnosis does not clearly improve paired TMCP over fixed generic C–M–S revision, structured memory does not improve paired TMCP over no memory or raw memory, and persistent C–M–S is lower than reset C–M–S and no memory in the sparse complete-profile curriculum contrasts. Positive secondary contrasts therefore support only the narrower claim that verified structured memory changes discovered strategy diversity.

D Supplementary Literature-Inspired Comparisons

Table 12 documents the literature-inspired adaptations available in the development implementation. The stored Self-Refine/Reflexion and Skill-Pro comparisons use the common generation protocol, cases, and ceilings, alongside the matched controls described in Appendix C. Budget-matched independent sampling is also a main comparison. The archive- and policy-based adaptations listed below retain different feedback or state contracts and are not treated as matched main-table baselines. No superiority over an official system is inferred from any prompt-level adaptation.

Table 12: Baseline adaptations in the design-only pilot. The table describes the experimental proxies, not full reproductions of the cited systems.

Method	Mutable state	Purpose of comparison
Budget-matched best-of-N	None; independent GameSpecs	Controls for independent sampling under the same candidate and token ceilings.
Self-Refine / Reflexion (Madaan et al., 2023; Shinn et al., 2023)	Current artifact and textual feedback	Tests sequential revision without a persistent skill library, population archive, or adaptive improver.
Skill-Pro (Mi et al., 2026)	Structured skill memory	Skill-memory adaptation inspired by Skill-Pro; no claim of official implementation equivalence.
Meta Agent Search (ADAS) (Hu et al., 2025)	Prompt-defined agent state and archive	Archive-conditioned design-search adaptation, not executable agent-code search.
Darwin Gödel Machine (Zhang et al., 2026)	External proposal state and archive	Archive-based self-revision proxy; generated agent-code self-modification is outside this runner.
MetaSkill-Evolve (Wang et al., 2026)	Task-skill and meta-skill representations	Two-level skill-state adaptation without Ours-specific C–M–S memory.
Hierarchical Self-Improvement (Zhou, 2026)	Task and improvement prompt state	Hierarchical prompt-state adaptation rather than unrestricted harness rewriting.

E Four Browser-Game Evolution Lineages

The four lineages below expand Section 5.2.5 in the row order of Figure 1. Each contains four successive archived designs. The figure uses the display names Fire & Water, Sky Runner, Glass Lab, and Garden Defense for the source-game lineages described below. The version descriptions refer to stored design records and executable rules, not to player-experience measurements. These browser-game artifacts are separate from the frozen GameSpec comparisons; neither the illustrations nor the version sequences isolate a causal effect of memory.

Fireboy and Watergirl: memory relay. The archive `lin_1776695727_003` contains *Echo Relay Temple* (v1), *Relay Glyph Temple* (v2), *Relay Echo Temple* (v3), and *Memory Relay* (v4, titled *Memory Relay Temple* in its HTML artifact). The first version already combines parked-avatar activation with replay ghosts and sensors; these are not later inventions. The second version records the avatar's gravity orientation; the third mirrors imprinted replays onto a ceiling path and links sensor progress to parked-avatar activation. The final design retains parked states and replay activation as parts of a memory-relay route. The sequence illustrates revisions to an existing interaction structure rather than a progression from character switching without memory to memory appearing for the first time.

Flappy Bird: route writing. The archive `lin_1776687736_000` contains *Pulse Morph Run* (v1), *Crease Choir* (v2), *Neon Route Weave* (v3), and *Neon Route Echo* (v4). One-button flight and obstacle traversal persist throughout the sequence. Clean-pass changes to future gates, phase charging, and assistance from previous failed-run echoes already occur in v1. The second version adds hold-and-release wind creases, echo refraction, and a route-credit objective. The third and fourth connect gate changes, echoes, and phase states through progression objectives. Thus route writing describes the lineage's rule structure, not a capability first introduced in v2. The v2 name follows its node record and executable artifact rather than the archive tree's generic label.

Happy Glass: conditional fluid routing. The archive `lin_1776692493_002` contains *Ink Ritual Cup* (v1), *Echo Ink Ritual* (v2), *Ink Chain Codex* (v3), and *Ritual Ink Cup* (v4). The first version already supports delayed droplet storage and release, but uses fixed downward gravity. The second and third versions extend delayed release with gravity programming; v3 also links charged droplets and neighboring strokes. In v4, an absorb stroke releases a droplet in its stored direction, whereas limited manual turns change global gravity separately. Solid strokes route droplets, and ritual state distinguishes charged fill from ordinary fill. The sequence changes the relation between geometry, stored fluid, and state transitions rather than simply accumulating all earlier mechanisms.

Plants vs. Zombies: charged interception. The archive `lin_1776689871_001` contains *Fireline Garden* (v1), *Neon Bent Lanes* (v2), and two successive versions titled *Resonance Garden* (v3 and v4). The first version includes wave forecasts and placement-based obstruction: a plant in front of a shooter can block and delete an allied projectile. The second version introduces lane bending, and the third makes intercepted shots useful by storing charge in generators for later discharge. In v4, reaching full charge triggers discharge automatically; a bend can strengthen the next discharge once per lane per wave. Forecast-guided placement therefore coordinates immediate defense with preparation for a later attack. Forecasts are retained from the first version, not introduced in the last.