SAME READOUT, DIFFERENT ROLLOUT: WHEN TO LET GO

Anonymous authors
Paper under double-blind review

ABSTRACT

Two histories can reach the same complete source state and yield identical, correct model readouts, yet the model predicts different futures under the same actions. We study such pairs after a memory value is deleted or an object returns by a different path. In the tested world-model pairs, every initially correct pair agrees after one action, yet 8.2–16.8% separate within 16; at the first split, one branch is usually source-correct. We test the classical readout-closure requirement on these pairs: an update must preserve the equivalence defined by current answers. Reconstructing one state from those answers restores pair agreement and improves 16-step source accuracy in the world-model tests. Memory results show why agreement and accuracy remain separate. A structured updater illustrates how known identifier and storage correspondences can be preserved through future updates. Correct current answers therefore need not make learned states interchangeable for continued computation.

1 INTRODUCTION

Two histories leave a memory with the same contents. A key held one value along the first history and another along the second. Both values have since been deleted. In one recorded pair, the learned states answer every query about the 64-key table correctly. Two identical copy operations follow, and every answer is still correct. After the next write, their answers begin to differ at two keys that the write should have left unchanged.

An object in a world model takes one of two short paths and returns to the same place. The rest of the scene is unchanged. Both learned states recover every object's position correctly, and their predictions initially agree under the same subsequent actions. A few steps later, they separate. In this environment, the current positions fully determine the effects of those actions. The route back carries no further information the source dynamics need.

In both cases, the histories have reached the same state in the *source*, the system being modeled. Their differences no longer affect its future, but remain visible in what the learned model does next. We ask when agreement at *readout*—the current answers—survives later updates. If it does, the two retained states are interchangeable for those operations. If it does not, a closed rollout can depend on a history the source has already forgotten.

This is a temporal test of interchangeability: substitute one learned state for another, apply the same specified actions, and compare their later answers. In a closed rollout the model receives no new source observation, so a later difference comes from what its retained states still carry.

We examine tasks with a complete, sufficient source state: a table determines future memory operations, and object positions determine future moves in a grid without inertia. Different histories can therefore lose their relevance to the source before losing their influence on the learned dynamics. Figure 1 follows this difference in two neural-memory checkpoints and two contrastive structured world-model checkpoints. The world-model pairs initially agree even after one more action; some separate only under longer continuations.

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.

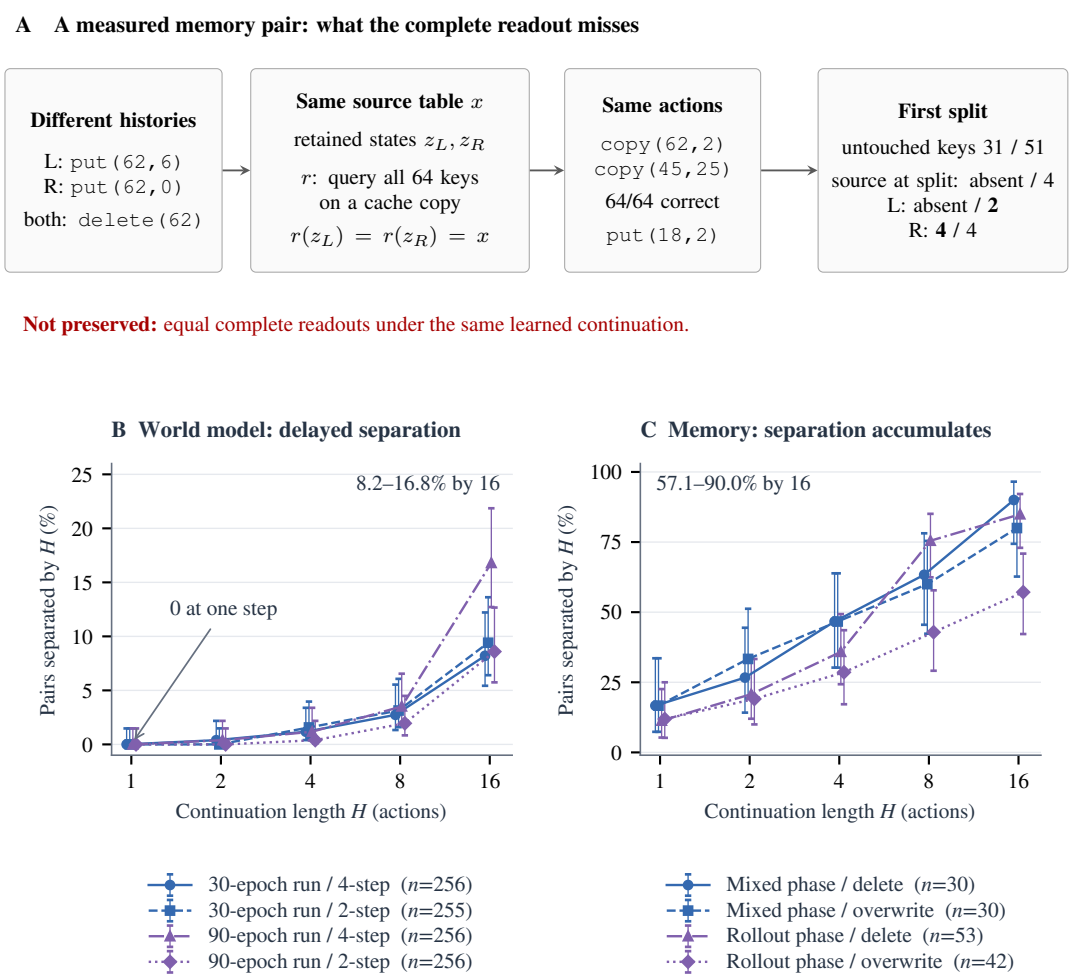

Figure 1: **Correct answers now can diverge after later updates.** (A) A recorded memory pair: deleting key 62 makes the two source tables equal. The declared readout queries all 64 keys on a copy of each retained cache; both complete answers equal the source table. After two common copy actions they remain correct, but the next write makes each branch err on a different untouched key. (B) All initially correct C-SWM pairs agree after one action, but 8.2–16.8% separate by 16 actions. (C) Memory histories show 57.1–90.0% separation. A pair counts once if any of three fixed continuations separates by H. Points include only initially source-correct pairs; whiskers are pointwise Wilson 95% intervals. Horizons are $\log_2$-spaced and vertical scales differ. World-model labels identify separate training runs by their retained epochs; memory labels name the final training phase of each checkpoint. Exact seeds and checkpoint identifiers are in Appendix B. Conditions and counts are in Section 3 and Appendix B.

At the first split in these world-model pairs, one branch usually still predicts the correct source state. Reconstructing once from the current answers improves 16-step source correctness across all four world-model conditions, so the delayed difference bears on the quality of continued prediction.

This matters wherever a representation becomes the starting point for further computation. World models roll predicted states forward to evaluate actions (Assran et al., 2025) and reuse components across settings (Wang & Huang, 2025; Zhang et al., 2026). Agent and associative memories repeatedly revise retained information (Xu et al., 2025; Behrouz et al., 2025); continual learners transport prototypes between representations (Xu & Krawczyk, 2026). At such a handoff, matching current answers is a natural check. Our examples ask whether the next computation also depends on how the model arrived there.

The analysis asks which agreements survive further updates. The classical congruence criterion supplies an analysis tool: the next readout must be determined by the current readout and action. A linear specialization isolates the obstruction: differences hidden from the current readout can flow back into later answers. Our contribution is to test source-equivalent, currently correct learned-state pairs for delayed separation, then compare reconstruction's effects on pair agreement, answer preservation and source accuracy. These paired tests make the closure question observable in a learned system, including where a short successful check stops being informative.

For the separate case of a known correspondence between decoded states, we give an architectural construction. Identifier renaming or record-slot reordering specifies the correspondence; equality tests, pointers and shared set operations give an updater whose predictions transform consistently through every finite rollout. The history-pair test instead starts from retained states with equal current readouts and does not assume such a correspondence. We use this nominal replacement transducer (NRT) to examine how an exact correspondence can be built into learned state updates, alongside direct structured and canonicalized alternatives. Together, the history-pair tests and these constructions connect the meaning of a retained state to the operations that follow it.

Section 2 develops the closure criterion and the reconstruction guarantees. Section 3 tests them on history pairs. Section 4 gives the architectural realizations and their predictive evaluation.

2 From current answers to interchangeable states

2.1 Which current agreements survive an update?

In the opening examples, two histories end at the same source state and both learned states report that state correctly. Their internal states need not be equal. To describe what can happen next, fix a state space Z, an action set $\mathcal{A}$, deterministic updates $\mathcal{F}_a : Z \to Z$, and a readout $r : Z \to Y$, where $Y = r(Z)$. The domain is closed under the updates. State z includes everything the updater retains, including any cache, position counter or other history-dependent input. Parameters and the readout are fixed. We initially use the same action coordinates on both branches; partial actions and coordinate changes are addressed below.

For $w = a_1 \cdots a_h$, write $\mathcal{F}_w = \mathcal{F}_{a_h} \circ \cdots \circ \mathcal{F}_{a_1}$, with $\mathcal{F}_\epsilon = \mathrm{id}_Z$. Define current readout equivalence by

$$z \, E_0 \, z' \quad \Longleftrightarrow \quad r(z) = r(z'). \tag{1}$$

In the experiments r returns all memory values or all object positions. The following is the classical congruence and quotient criterion, specialized to these updates (Burris & Sankappanavar, 2012, Chapter II, Sections 5–6).

Proposition 1 (Readout closure)**.** *The following conditions are equivalent:*

1. *For every $a \in \mathcal{A}$ and every $z \, E_0 \, z'$, the successors satisfy $\mathcal{F}_a(z) \, E_0 \, \mathcal{F}_a(z')$.*
2. *For every a, there is a unique $f_a : Y \to Y$ such that $r \circ \mathcal{F}_a = f_a \circ r$.*
3. *For every $z \, E_0 \, z'$ and every finite action sequence w, $r(\mathcal{F}_w(z)) = r(\mathcal{F}_w(z'))$.*

The first condition makes $f_a(rz) = r(\mathcal{F}_a z)$ independent of the chosen state. Induction gives agreement after any finite continuation; conversely, those continuations include each single action. The law applies to every equal-readout pair, including pairs reached later in a rollout. A short check at the initial state covers only part of this requirement. Source correctness additionally asks whether f_a matches the system being modeled.

For integer horizons $H \geq 0$, let

$$E_H = \bigcap_{|w| \leq H} \ker(r \circ \mathcal{F}_w), \qquad E_\infty = \bigcap_{H \geq 0} E_H. \tag{2}$$

Here $\ker(g)$ denotes pairs with equal values under g, not a linear null space. These equivalences become finer as H increases. Their intersection is the largest update-closed equivalence contained in E_0; the proof is in Appendix A. Longer continuations can therefore expose differences that a shorter check leaves unresolved. Our experiments measure the first separation on a fixed panel of continuations.

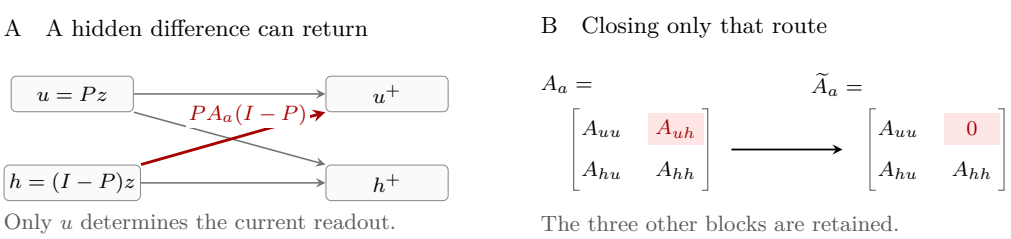

Figure 2: **A hidden difference matters when an update makes it visible.** For the linear specialization, $P = C^+C$ splits a state into visible and hidden coordinates u and h; $Cz = Cu$. The red block carries hidden differences into the next visible state. Readout closure requires this block to vanish, equivalently $CA_aN = 0$. Replacing A_a by $A_a - PA_a(I-P)$ removes just this block and is the smallest Frobenius-norm correction; the other three blocks and b_a are unchanged. The blocks use orthonormal bases of the two subspaces.

What may remain hidden. A linear example makes the condition concrete. Let $r(z) = Cz$ and $\mathcal{F}_a(z) = A_az + b_a$, with C full row rank and columns of N spanning $\ker C$. States with the same answers differ by Nv. Their next answers differ by CA_aNv, so closure is exactly

$$CA_aN = 0 \qquad \text{for every } a.$$

The model may retain distinctions invisible to its readout. What matters is whether an update can bring those distinctions back into the answers. With $P = C^+C$, the nearest closed matrix in Frobenius norm is $A_a - PA_a(I-P)$: only the block carrying hidden differences into the visible part is removed (Figure 2). The other blocks remain. This linear specialization (Appendix A) illustrates the structural requirement without requiring the entire internal state to be reconstructed.

2.2 Restoring Agreement and Preserving Predictions

A deterministic reconstruction can discard the branch-specific difference by choosing a state from the current answers. Let $s : Y \to Z$ depend only on those answers, and define

$$\mathcal{R} = s \circ r, \qquad \widehat{\mathcal{F}}_a = \mathcal{R} \circ \mathcal{F}_a \circ \mathcal{R}. \tag{3}$$

The same answers must select the same complete state, including any cache or clock used in later updates. An implementation requires a computable s.

Proposition 2 (Reconstruction and its separate guarantees)**.** *For any such deterministic s, E_0 is closed under $\widehat{\mathcal{F}}_a$. If s is an exact section, $r \circ s = \mathrm{id}_Y$, then $r\mathcal{R} = r$, $\mathcal{R}^2 = \mathcal{R}$, and*

$$r\widehat{\mathcal{F}}_a = f_a^s r, \qquad f_a^s = r\mathcal{F}_a s. \tag{4}$$

Consequently $r\widehat{\mathcal{F}}_w = f_w^s r$ for every finite $w = a_1 \cdots a_h$, where $f_w^s = f_{a_h}^s \circ \cdots \circ f_{a_1}^s$ and the repaired update is applied at each step. The empty composition is the identity.

Equal readouts give exactly the same reconstructed state, which proves closure. The section identity additionally preserves the current answers. Applying $\mathcal{R}$ only once also makes initially equal pairs agree under subsequent deterministic native updates. The once-only construction gives agreement for those initialized pairs; the per-step construction gives an E_0-closed update on Z. They can have different dynamics: in general $\widehat{\mathcal{F}}_w \neq \mathcal{R}\mathcal{F}_w\mathcal{R}$.

Closing an update that separates an equal-readout pair necessarily changes at least one of its original predictions. This can correct an error or introduce one; the closure condition alone does not determine which. Appendix A gives the prediction-preservation statement and its elementary metric bound.

Source correctness, legal actions and transported actions. To interpret the readout as a source state, it must be sufficient to determine the source's future. If $T_a : Y \to Y$ is that deterministic source update, an exact section and $r\mathcal{F}_a s(y) = T_a(y)$ for every y, a imply $r\widehat{\mathcal{F}}_w(z) = T_w(rz)$. With

a correct initial readout, these conditions give a source-correct rollout. With partial updates $\mathcal{F}_a : L_a \to Z$, a faithful quotient also requires equal-readout states to agree on whether a is executable. The repaired update is naturally defined where $\mathcal{R}z \in L_a$, which need not be the original domain L_a. One can instead make failure explicit and absorbing, but agreement in failure then remains different from legal, correct execution. Appendix A gives these conditions, the full proofs, and an exact linear specialization.

The history-pair tests below use the same actions in the same coordinates. Section 4 then considers identifier and slot transformations, for which actions and query references transform along with the state. In each case, the algebra specifies the correspondence that the next operation must preserve.

3 TWO HISTORIES, ONE CURRENT STATE

Different histories, the same source state. Two histories may reach the same source state without producing the same learned state. In a neural memory, writing different values and then overwriting or deleting them yields the same complete table. In a contrastive structured world model (C-SWM; Kipf et al., 2020), two legal return paths yield the same five-object configuration. We retain two memory checkpoints and two C-SWM checkpoints, with two history families per model and 256 generated histories or scenes per condition. Each pair receives three fixed continuations of length 16, without new source observations. The comparison tests whether the agreements in Proposition 1 survive these continuations.

The memory is a task-trained, four-layer gated-delta recurrent network with 512-dimensional hidden representations. Its retained state contains the convolution and recurrent caches; querying a key reads a copy of that state. The two checkpoints are successive points in one training lineage, not independent initializations. C-SWM uses an object-factored latent transition and a separately fitted coordinate readout. Appendix B specifies both interfaces and the memory training schedule.

The updater and readout remain fixed throughout each comparison. Source states are used to generate the legal action sequences, identify the initially correct cohort and score predictions; they are not supplied as corrective observations to the learned rollout. Reconstruction uses only the model's current readout.

Here z is the retained internal state, a an action, $\mathcal{F}_a$ its update, and $r(z)$ the complete *declared readout*: all 64 key values for memory or the five object positions for C-SWM. Write $E_0 = \{(z, z') : r(z) = r(z')\}$, and let $C_0 \subseteq E_0$ contain pairs whose two current readouts both equal the source. Equality is exact over these discrete outputs. Both source states are sufficient for their deterministic dynamics: the table determines subsequent memory operations, and grid positions determine subsequent moves.

A short passing check can miss a later difference. Figure 1 conditions on C_0: both complete readouts agree with the source before continuation. In C-SWM, no pair gives different readouts after the first action, while 21/256, 24/255, 43/256 and 22/256 scenes separate by action 16 across the four conditions. Memory pairs give different readouts in 27/30, 24/30, 45/53 and 24/42 initially correct history pairs. Each numerator counts a pair once if any of its three fixed continuations has separated by action 16. The first such separation provides a witness to failure of readout closure.

The opening world-model pair illustrates the delay. Its two complete readouts are correct initially and after the first two common actions. At the third action they agree on an incorrect state; at the fourth they first disagree. The memory pair takes a different course: every key remains correct until the third action changes the branches' answers at two untouched keys. Appendix B gives the exact operations and the selection and replay procedure for these examples.

Replaying every step shows where the difference first matters. Among the C-SWM units that separate, 20/21, 18/24, 35/43 and 17/22 remain source-correct at every step before their earliest separating suffix first diverges. At the split, one complete readout is source-correct in 21/21, 22/24, 40/43 and 22/22 units. The corresponding memory clean-onset counts are 16/27, 11/24, 21/45 and 12/24. Thus two states that have answered the same complete source state correctly can lead to different next answers, usually with one branch correct at the first split. Table 5 gives the per-condition timing and first-split counts.

Reconstructing a state from its current answers. Choose a deterministic reconstruction s using only the current readout and set $\mathcal{R} = sr$. Memory reconstructs a cache by a fixed replay of the decoded table; C-SWM renders the decoded positions and re-encodes the image. Compare continuation using the unmodified learned updates (*native*), a single initial reconstruction (R-once), and $\widehat{\mathcal{F}}_a = \mathcal{R}\mathcal{F}_a\mathcal{R}$ at every action (RFR), starting directly from z. Since equal readouts give the same reconstructed state, this deterministic construction preserves agreement under subsequent updates. When $\mathcal{R}$ is not idempotent, adjacent RFR actions apply $\mathcal{R}^2$ at their shared boundary; this is part of the measured operator. An *exact section*, $rs = \mathrm{id}$, reconstructs a state whose readout equals the supplied answers and therefore also preserves the current readout; deterministic reconstruction alone need not do so. These are the distinct guarantees in Proposition 2.

Both RFR and R-once have zero disagreements through $H = 16$ in every tested E_0 cohort. A single reconstruction therefore suffices for the measured pair agreement. Whether this common continuation is accurate is a separate comparison. Figure 3 compares all three modes on the same initially correct pairs and endpoint within each condition: $H = 16$ for C-SWM and $H = 8$ for memory. A pair counts as source-correct only when both complete readouts match the source on all three continuations.

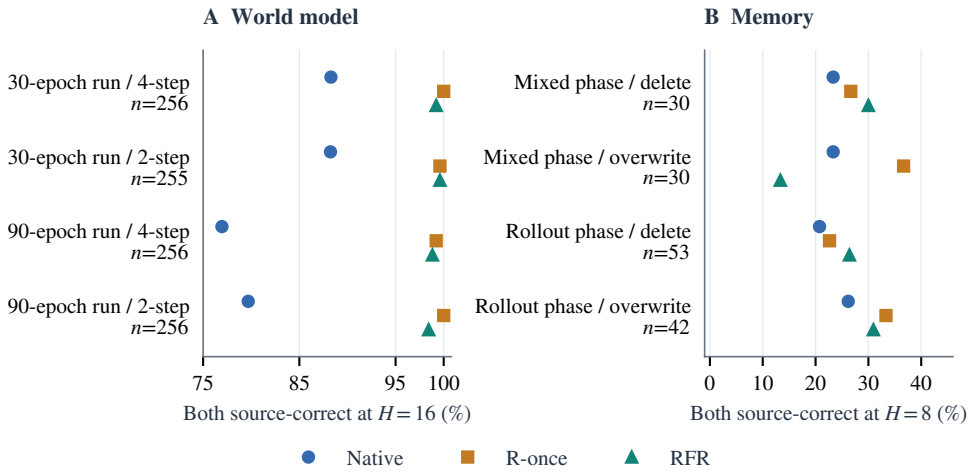

Figure 3: **One reconstruction improves long-horizon source accuracy in C-SWM; agreement alone does not select a good continuation.** Native, R-once and RFR use the same initially correct C_0 pairs within each condition. A unit counts only when both complete readouts are source-correct at the endpoint on all three suffixes. (A) C-SWM endpoint correctness at $H = 16$. (B) Memory at $H = 8$ shows that repeated reconstruction can help or hurt relative to one reconstruction. Both repaired modes give 100% pair agreement through $H = 16$; panels use different horizons and horizontal ranges as labeled. Table 3 gives counts.

At $H = 16$, the four C-SWM conditions have native source-correct rates of 76.95–88.28%, R-once rates of 99.22–100%, and RFR rates of 98.44–99.61%. On the same units, R-once improves/worsens the both-branch endpoint result relative to native updates in 30/0, 29/0, 57/0 and 52/0 cases, respectively; Tables 3 and 4 give the marginal and paired counts. This paired metric combines branch accuracy and error correlation. Even so, in the 90-epoch-run/four-step world-model condition it bounds native mean single-branch accuracy on all three suffix endpoints by 88.48%, below R-once's 99.22% single-branch rate.

In mixed-phase memory with overwrite at $H = 8$, R-once gives 11/30 source-correct pairs, native updates 7/30, and RFR 4/30. Rollout-phase memory with deletion gives a different ordering: 11/53, 12/53 and 14/53 for native, R-once and RFR. At $H = 8$ in C-SWM, native correctness is 94.9–

97.7%, compared with 99.2–100% after one reconstruction and 99.2–100% with RFR. Agreement alone does not select between these updates.

Reconstruction can also change the answers it starts from. In memory, the first reconstruction preserves both complete current readouts in 56.7–78.6% of C_0 pairs. In C-SWM, three tested unit-within-condition incidences reach bilateral absorbing failure, counted as agreement but as incorrect against the source. Appendix B reports current-answer preservation, endpoint accuracy through $H = 16$, and a same-input example where restoring agreement corrects the native errors.

4 BUILDING CORRESPONDENCE INTO THE UPDATER

Reconstruction chooses a common state at the readout interface. Another route is available when the relevant correspondence is known in advance: make each update respect it. Consider a store with records named a, b, and an action that marks a complete. Renaming the records to u, v or exchanging their storage locations changes the encoding. The action and its references change with it, so updating either encoding should produce corresponding stores.

4.1 IDENTIFIER AND STORAGE EQUIVARIANCE

Let X be the typed decoded state space, U the typed action space, and $M_\theta : X \times U \to X$ the learned update, including its deterministic decoder. The predicted state feeds the next update. For $g = (\sigma, \pi)$, where σ permutes record slots and π consistently renames identifiers, the required law is

$$M_\theta(g\cdot x, g\cdot u) = g\cdot M_\theta(x, u). \tag{5}$$

Identifiers have no semantic content, and slots are storage coordinates. Source legality and query references transform along with the state and action. Undoing g therefore makes the two predictions directly comparable. This law concerns known coordinate changes on decoded states. The history-pair tests concern retained latent states with equal readouts, which need not be connected by such a change. In both cases, the next update must preserve the specified relation.

NRT implements this law through three choices. Names enter learned computation through equality tests; output names are copied from the state or current action; and shared set and ordered-pair layers process records and links. A name-decoding tie maps to the fixed null symbol. For every parameter value, these operations give exact joint equivariance in deterministic, exact-arithmetic evaluation. Composition preserves the law through any finite decoded rollout, and each output name comes from the initial state or an intervening action. Appendix C specifies the interface and Theorem 1 states the guarantee; Appendix D proves it.

4.2 PREDICTIVE QUALITY WITH AN EXACT CORRESPONDENCE

We compare NRT with a Symbol-Invariant Transformer (SIT) adaptation (Işık & Li, 2026). SIT already guarantees identifier covariance, meaning that predictions agree after undoing a renaming. Its serialized record positions are trained with random slot permutations. Both models have about 90K parameters and receive identical one-step training examples and minibatch schedules within a seed. We use record-store and tiered-memory families, 256 or 1,024 sampled training transitions, five model seeds, and closed rollouts of 2, 3, 5 or 8 actions. Appendix E gives the transition rules, optimization and evaluation protocol.

Ordinary query accuracy compares a prediction with the source. Whole-state covariance compares the two predictions after undoing the encoding change (Figure 5 in the appendix). Both models achieve 100% identifier covariance. NRT also has 100% slot covariance in all 20 runs, with 97.53–99.97% ordinary query accuracy. The augmented SIT adapter has 67.08–99.38% whole-state slot covariance and 90.70–99.92% ordinary accuracy. NRT meets the prespecified two-percentage-point non-inferiority criterion in every condition. The state-level comparison can expose differences that a sampled query misses: at Records / 1,024, SIT's slot intervention changes the predicted state in 15.00% of episodes and the requested answer in 0.90% of queries.

Two direct alternatives remove absolute slot coordinates (Structured SIT) or canonicalize the full state/action interface (Canonical SIT; Appendix E.5). Table 7 gives the fresh five-seed comparison. All three have 100% observed covariance. Structured SIT exceeds NRT in mean query and

full-state accuracy in all four conditions. At Records / 1,024, query accuracy is 99.85% versus 98.69% and full-state accuracy is 97.50% versus 75.21% (Structured SIT versus NRT). The same correspondence law permits different predictive quality.

The storage effect depends on the transition family. In five-object Blocksworld, both models reach 100% query accuracy, final-state accuracy and both covariances at all tested depths. The architectural guarantee fixes the correspondence regardless of training outcomes; predictive quality remains an empirical property. Full comparisons are in Appendix E.

5 Related work

Equivalence under continued computation. Congruences and quotient algebras characterize equivalences preserved by operations (Burris & Sankappanavar, 2012). State abstraction and bisimulation apply related requirements to action-dependent behavior (Givan et al., 2003). Our experiments use this perspective to compare learned states after histories have merged in a sufficient source state. The distinguishing continuation identifies where agreement at readout stops being enough; reconstruction then separates closure, answer preservation and source correctness.

Compositional world models and memory. Object-oriented world models study algebraic compositional generalization (Zhao et al., 2022), learned group structure (Park et al., 2022), and entity–rule binding (Sehgal et al., 2024). C-SWM learns object-factored latent transitions through a contrastive objective (Kipf et al., 2020). Recent world-model composition, associative memory and persistent agent memory motivate reusing learned states across further operations (Wang & Huang, 2025; Zhang et al., 2026; Behrouz et al., 2025; Xu et al., 2025). Our history-pair comparison holds the checkpoint, current source state and continuation fixed, and tests whether the route to that state still changes the predicted future.

Names, binding and equivariant updates. Interchangeable Token Embeddings study identifier-renaming covariance (Işık et al., 2025); Symbol-Invariant Transformers guarantee it architecturally (Işık & Li, 2026). Nominal automata give state-machine semantics for atom permutations (Moerman et al., 2017). External-memory binding (Webb et al., 2021), relation-based representations (Kerg et al., 2022; Altabaa et al., 2024) and pointer networks (Vinyals et al., 2015) provide complementary mechanisms. NRT combines equality-based binding and supported pointer output with record-slot equivariance, extending the correspondence through decoded state updates. Frame averaging provides a general way to impose a known symmetry on a backbone (Puny et al., 2022); our canonicalized comparator applies this principle to the joint state/action interface, including symmetric encodings and allocation.

6 Discussion

The histories in our examples have already become interchangeable for the source. The model reads their common state correctly, yet can continue differently from each one. In C-SWM, every initially correct pair passes the one-step agreement check; at the first split, one branch is usually source-correct. The answer to a future action can therefore depend on which learned state produced today's correct readout. Closure identifies the missing requirement: updates must preserve the equivalence expressed by that readout. The linear example allows hidden differences to persist, provided that updates never bring them back into the answers.

Reconstruction makes the distinction tangible. Choosing the same state from the same answers restores agreement, even with a single reconstruction at handoff. An exact section also keeps those answers; correct dynamics on its representatives then give correct source rollouts. These are separate conditions. At 16 steps in C-SWM, one reconstruction raises the strict both-branch source-correct rate from 76.95–88.28% to 99.22–100%; repeated reconstruction has mixed effects in memory. The representative chosen for a current answer can change whether its common continuation is correct. In the 90-epoch-run/two-step world-model condition, 36 of 52 R-once gains have no native split on the tested 16-step panel: reconstruction also corrects shared errors.

For deterministic updates and a complete, source-sufficient readout, paired continuation tests whether source-equivalent histories remain interchangeable to the model.

REPRODUCIBILITY STATEMENT

Appendices B and E–H describe the fixed checkpoints, action panels, source simulators, model configurations and per-depth results. The anonymous artifact prepared with this manuscript supplies stored history-pair outputs and timelines, corrected direct-comparison records, and standalone aggregate-reanalysis scripts; its README gives commands and expected outputs. It excludes checkpoints, model and source-simulator implementations, earlier confirmation outputs, and figure-rendering code. Result reanalysis is supported; retraining, checkpoint inference and complete experiment replication require the separately retained materials.

AI USE STATEMENT

Generative AI tools assisted with code implementation, experiment orchestration, literature search, analysis scripts, artifact preparation, and manuscript drafting. All numerical claims are derived from archived executable outputs; the submission authors are responsible for checking the code, sources, analysis, and final text.

REFERENCES

Awni Altabaa, Taylor Webb, Jonathan Cohen, and John Lafferty. Abstractors and relational cross-attention: An inductive bias for explicit relational reasoning in transformers. In *International Conference on Learning Representations*, 2024.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, et al. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025. URL https://arxiv.org/abs/2506.09985.

Ali Behrouz, Meisam Razaviyayn, Peilin Zhong, and Vahab Mirrokni. It's all connected: A journey through test-time memorization, attentional bias, retention, and online optimization. *arXiv preprint arXiv:2504.13173*, 2025.

Stanley Burris and H. P. Sankappanavar. A course in universal algebra. Authors' online edition, 2012. URL https://www.math.uwaterloo.ca/~snburris/htdocs/UALG/univ-algebra2012.pdf. Millennium edition, 2012 update; originally published in 1981.

Robert Givan, Thomas Dean, and Matthew Greig. Equivalence notions and model minimization in Markov decision processes. *Artificial Intelligence*, 2003. URL https://cs.brown.edu/people/tdean/publications/archive/GivanetalAIJ-03.pdf.

Joerg Hoffmann. FF domain collection: Blocksworld-4ops. https://fai.cs.uni-saarland.de/hoffmann/ff-domains.html, 2000. AIPS-2000 four-operator STRIPS domain and generator collection.

İlker Işık and Wenchao Li. Names don't matter: Symbol-invariant transformer for open-vocabulary learning. *arXiv preprint arXiv:2601.23169*, 2026. ICML 2026; official code at https://github.com/BU-DEPEND-Lab/Symbol-Invariant-Transformer.

İlker Işık, Ramazan Gokberk Cinbis, and Ebru Aydin Gol. Interchangeable token embeddings for extendable vocabulary and alpha-equivalence. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 26523–26541, 2025.

Giancarlo Kerg, Sarthak Mittal, David Rolnick, Yoshua Bengio, Blake Richards, and Guillaume Lajoie. On neural architecture inductive biases for relational tasks. *arXiv preprint arXiv:2206.05056*, 2022.

Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1gax6VtDB.

Joshua Moerman, Matteo Sammartino, Alexandra Silva, Bartek Klin, and Michał Szynwelski. Learning nominal automata. In *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages*, pp. 613–625, 2017. doi: 10.1145/3009837.3009879.

Jung Yeon Park, Ondrej Biza, Linfeng Zhao, Jan-Willem Van De Meent, and Robin Walters. Learning symmetric embeddings for equivariant world models. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 17372–17389, 2022.

Omri Puny, Matan Atzmon, Heli Ben-Hamu, Ishan Misra, Aditya Grover, Edward J. Smith, and Yaron Lipman. Frame averaging for invariant and equivariant network design. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2110.03336.

Atharva Sehgal, Arya Grayeli, Jennifer J. Sun, and Swarat Chaudhuri. Neurosymbolic grounding for compositional world models. In *International Conference on Learning Representations*, 2024.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Xinyue Wang and Biwei Huang. Modeling unseen environments with language-guided composable causal components in reinforcement learning. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2505.08361.

Taylor W. Webb, Ishan Sinha, and Jonathan D. Cohen. Emergent symbols through binding in external memory. *arXiv preprint arXiv:2012.14601*, 2021.

Hongye Xu and Bartosz Krawczyk. Two-way is better than one: Bidirectional alignment with cycle consistency for exemplar-free class-incremental learning. In *International Conference on Learning Representations*, 2026.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-Mem: Agentic memory for LLM agents. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Shaowei Zhang, Jiahan Cao, Xunlan Zhou, Shenghua Wan, and De-Chuan Zhan. BRICKS-WM: Building reusability via interface composition kinetics for structured world models. *arXiv preprint arXiv:2606.16489*, 2026.

Linfeng Zhao, Lingzhi Kong, Robin Walters, and Lawson L. S. Wong. Toward compositional generalization in object-oriented world modeling. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 26841–26864, 2022.

A Readout closure: proofs and conditions

The results below use the fixed deterministic interface of Section 2.1. We first prove the closure and reconstruction statements, then specialize them to linear readouts and partial actions.

Proof of Proposition 1. Suppose condition 1 holds. For $y \in Y$, choose any z with $rz = y$ and set $f_a(y) = r\mathcal{F}_a(z)$. Closure makes the choice irrelevant. Since $Y = r(Z)$, this defines f_a everywhere and uniquely, proving 2. Conversely, 2 maps any equal-readout pair to equal readouts, proving 1. Induction on word length gives 3 from 1; the empty word is the initial equality. Taking words of length one in 3 gives 1.

Finite and unbounded continuations. Each E_H in Eq. 2 is an equivalence, since it is an intersection of equal-output relations. Including all words of length at most H makes $E_{H+1} \subseteq E_H$, and

$$E_{H+1} = E_0 \cap \bigcap_{a \in \mathcal{A}} (\mathcal{F}_a \times \mathcal{F}_a)^{-1}(E_H). \tag{6}$$

If $(z, z') \in E_\infty$, then every continuation after a common first action also agrees; hence $(\mathcal{F}_a z, \mathcal{F}_a z') \in E_\infty$. Conversely, any update-closed equivalence $E \subseteq E_0$ preserves every finite continuation by induction, so $E \subseteq E_\infty$. This proves the largest-closed-subrelation claim.

For a minimal separation, let $Z = \{0,1\}^2$, $r(x,h) = x$ and $\mathcal{F}(x,h) = (h,h)$. States $(0,0)$ and $(0,1)$ agree now and disagree after one action. For a delayed version, shift (x, h_1, h_2) to (h_1, h_2, h_2) and read its first coordinate: $(0,0,0)$ and $(0,0,1)$ agree for one step and separate at the second. The second example exposes a hidden difference one action later.

Proof of Proposition 2. If $rz = rz'$, then $\mathcal{R}z = \mathcal{R}z'$. Applying $\mathcal{R}\mathcal{F}_a$ gives equal successor states under the repaired update. This already proves closure without a section identity. With $rs = \mathrm{id}_Y$,

$$r\mathcal{R} = rsr = r, \qquad \mathcal{R}^2 = srsr = \mathcal{R}, \tag{7}$$

$$\widehat{\mathcal{F}}_a = s(r\mathcal{F}_a s)r = sf_a^s r, \qquad r\widehat{\mathcal{F}}_a = f_a^s r. \tag{8}$$

Repeated substitution proves the word identity, with the empty word interpreted as the unmodified initial readout. Without an exact section, the repaired readout update is instead rsf_a^s, so neither current-answer preservation nor idempotence follows. A usable implementation also needs s to be computable; existence as a set map is not an algorithmic guarantee.

Reconstruction once and reconstruction at each step have different dynamics. For example, take $Z = \{0, 1, 2\}$, readouts $(r(0), r(1), r(2)) = (0, 0, 1)$, section $s(0) = 0, s(1) = 2$, and update $(\mathcal{F}(0), \mathcal{F}(1), \mathcal{F}(2)) = (1, 2, 2)$. From state 0, two repaired steps remain at 0, whereas $\mathcal{R}\mathcal{F}^2\mathcal{R}(0) = 2$. Nevertheless, applying $\mathcal{R}$ once to an equal-readout pair makes its two starting states equal, so subsequent native updates preserve agreement between those branches. The R-once comparison measures agreement after this shared initialization.

Proposition 3 (Preserving the original predictions)**.** *There exists an E_0-closed update family $\mathcal{G}_a : Z \to Z$ satisfying $r\mathcal{G}_a(z) = r\mathcal{F}_a(z)$ for every z, a if and only if $\mathcal{F}_a$ is already E_0-closed. If an equal-readout pair has native successors $p = r\mathcal{F}_a(z)$ and $p' = r\mathcal{F}_a(z')$, every closed repair gives a common readout q with*

$$\max\{d(p, q), d(p', q)\} \geq \tfrac{1}{2} d(p, p') \tag{9}$$

for any metric d on Y.

Preserving every original prediction would transfer the repair's closure to the original update. The bound follows from the triangle inequality. It measures prediction changes from the same two input states. Source accuracy can rise or fall depending on the common output selected by the repair. The obstruction is to preserving all native predictions while closing the update.

Proof of Proposition 3. Suppose a closed $\mathcal{G}$ preserves all native one-step predictions. For every equal-readout pair and every action,

$$r\mathcal{F}_a z = r\mathcal{G}_a z = r\mathcal{G}_a z' = r\mathcal{F}_a z'.$$

Thus the native family is closed. The reverse implication takes $\mathcal{G} = \mathcal{F}$. If the native family is closed and s is an exact section, then $r\mathcal{F}_a sr = r\mathcal{F}_a$; the reconstruction also preserves all native readout rollouts by induction. For the metric bound, closure gives one common repaired readout q, and $d(p, p') \leq d(p, q) + d(q, p') \leq 2\max\{d(p, q), d(p', q)\}$. The same statement applies to a fixed word, using its original input pair.

Linear readouts and the smallest matrix correction. Let $Z = \mathbb{R}^n$, $r(z) = Cz$ with full-row-rank $C \in \mathbb{R}^{m \times n}$, and $\mathcal{F}_a(z) = A_a z + b_a$. Let N have orthonormal columns spanning $\ker C$. Two states agree at readout exactly when their difference is Nv. Their next difference is CA_aNv; the affine offset cancels. Therefore

$$E_0 \text{ is closed} \quad \Longleftrightarrow \quad CA_aN = 0 \quad \text{for every } a. \tag{10}$$

For any $\delta \in \ker C$ and $w = a_1 \cdots a_h$,

$$r(\mathcal{F}_w(z + \delta)) - r(\mathcal{F}_w(z)) = CA_{a_h} \cdots A_{a_1}\delta. \tag{11}$$

The difference subspaces corresponding to E_H obey $V_0 = \ker C$ and $V_{H+1} = V_0 \cap \bigcap_a A_a^{-1}(V_H)$. Their dimensions can strictly decrease at most $n - m$ times; if two consecutive subspaces coincide, the recursion has reached a fixed point.

Write $P = C^+C$, the orthogonal projector onto the row space of C. Since $CP = C$ and C is injective on that row space, Eq. 10 is equivalent to $PA_a(I - P) = 0$. The unique nearest matrix in Frobenius norm satisfying this constraint is

$$A_a^{\mathrm{closed}} = A_a - PA_a(I - P). \tag{12}$$

Indeed, every matrix is the sum of its four mutually orthogonal blocks under the row/kernel decomposition. The constraint sets only the block from the kernel into the row space to zero; the other three remain unchanged at the minimum. The affine offset may stay b_a.

For the section $s(y) = C^+y$, reconstruction instead gives the full-state map $\widehat{\mathcal{F}}_a(z) = PA_aPz+Pb_a$. It has the same one-step readout as the correction above, but generally a different internal-state map. The Frobenius-minimal correction and canonical reconstruction should therefore not be identified as the same operator.

Source correctness requires a sufficient readout. Assume the source has deterministic total updates $T_a : Y \to Y$ and $r\mathcal{F}_a s = T_a$ for an exact section. Then Eq. 4 gives $r\widehat{\mathcal{F}}_a = T_a r$, and induction gives $r\widehat{\mathcal{F}}_w = T_w r$. Interpreting this as accuracy requires the initial readout to match the actual source state. If the desired next output, considered as a function of the full state z, differs inside one r-fiber, no closed updater depending on those current answers alone can realize both outputs. Position without velocity is a familiar example of an insufficient readout. The complete memory table and the no-inertia grid positions used here are sufficient for their source rules. Their equal source states consequently have identical futures under a common legal continuation.

Partial actions. For $\mathcal{F}_a : L_a \to Z$, a quotient with the same executable interface exists exactly when L_a is a union of E_0-classes and equal-readout legal states have equal successor readouts. The first condition makes the quotient domain $r(L_a)$ well-defined for every representative, and the second makes its output well-defined. Necessity follows by lifting either a domain or output discrepancy; sufficiency defines the quotient action on that domain. The analogous induction then compares the same legal action sequences.

The natural domain of $\mathcal{R}\mathcal{F}_a\mathcal{R}$ is $\{z : \mathcal{R}z \in L_a\}$, not automatically L_a. If the section is exact and L_a is a union of readout classes, reconstruction preserves membership in L_a. For a merely deterministic reconstruction, even this domain guarantee can fail. Source correctness for partial actions further requires that the learned and source action domains correspond and that the representative updates are correct on those domains.

Alternatively, adjoin an absorbing state † with a distinct observable failure readout. Set every undefined update to †, and extend reconstruction to keep † fixed. The total statements then apply on the enlarged space. A pair reaching † agrees there but has not preserved successful execution. The experimental source-correctness metric counts this outcome as incorrect.

Transformation-generated correspondences. Under a known coordinate change g, compare a state and action (x, u) with $(g\cdot x, g\cdot u)$ and undo g on outputs. Source legality and query semantics must be transported too. With unchanged action coordinates and an invariant readout, equivariance can preserve the pairs generated by that transformation. It does not imply that every equal-readout pair is generated by one of those transformations.

For a finite illustration, let $Z = \{0,1\}^3$, $r(x,h,c) = x$, and $\mathcal{F}(x,h,c) = (h,h,c)$. Flipping c defines a nontrivial symmetry: $\mathcal{F}$ commutes with it and r is invariant. Yet $(0,0,0)$ and $(0,1,0)$ have equal current readouts and unequal next readouts. The example separates preservation of a transformation-generated correspondence from preservation of every readout class.

B HISTORY-PAIR CONSTRUCTION AND COMPLETE RESULTS

Figures 1 and 3 use the original-start evaluation described below. Each condition has 256 generated units and three fixed suffixes per unit. C_0 requires both complete current readouts to equal the source; E_0 requires only equality with each other. Filtering is at the initial state, before continuation outcomes. A cumulative separation counts a unit once when at least one of its three suffixes has separated by the displayed horizon. The three suffixes are repeated measurements of a unit. Checkpoints can share source fixtures, and all intervals are calculated within conditions.

B.1 MEMORY MODEL AND TRAINING LINEAGE

The four-block gated-delta memory has 512-wide hidden states, 1,024-wide RMS-normalized MLPs after each block, eight 64-wide heads and a width-four convolution. It is trained from scratch on table operations with the Qwen3-Next gated-delta layer in Transformers 4.57.6. Writes, copies and deletions use four tokens. The retained state is the convolution and recurrent caches.

Table 1: Exact figure counts. Native separation uses C_0; repaired separation uses E_0 and is zero for both RFR and R-once. "Keep" means both current readouts are preserved by the first $\mathcal{R}$. "Correct" means RFR's two branches match the source at $H = 8$ on all three suffixes.

			Native separation		Repair		
Condition	$\lvert C_0\rvert$	$\lvert E_0\rvert$	$H = 1$	$H = 16$	$H = 16$	Keep	Correct
M120 / delete	30	189	5/30	27/30	0/189	17/30	9/30
M120 / overwrite	30	175	5/30	24/30	0/175	20/30	4/30
H240 / delete	53	206	6/53	45/53	0/206	36/53	14/53
H240 / overwrite	42	187	5/42	24/42	0/187	33/42	13/42
S1 / 4-step	256	256	0/256	21/256	0/256	256/256	256/256
S1 / 2-step	255	255	0/255	24/255	0/255	255/255	254/255
S2 / 4-step	256	256	0/256	43/256	0/256	256/256	256/256
S2 / 2-step	256	256	0/256	22/256	0/256	256/256	254/256

For each of 64 key reads, a cloned cache takes a four-token query; a nine-logit linear head predicts 0–7 or absent, leaving the retained cache unchanged.

Training uses simulated tables and balanced live/absent queries. The H240 and M120 phases use a hierarchical loss that adds a binary presence loss to a value-classification loss on present keys; inference takes the largest of the nine logits. State supervision reads a table after its history; copy supervision adds a copy operation; rollout supervision reads at action depths 0, 1, 8 and 32. The mixed phase alternates state supervision with a stability objective that supervises copied and untouched values at depths 0, 3, 8, 16 and 32. AdamW uses weight decay 0.01, gradient clipping at norm one, and the implemented warmup/cosine schedule. Each row of Table 2 continues the preceding row's weights with a new optimizer. H240 and M120 are successive checkpoints of this lineage. Their names refer to the final phase lengths.

Table 2: Memory training phases. Steps are additional updates in each phase; learning rates are the phase's maximum settings. All phases use the same four-layer architecture.

Objective	Steps	Batch	Learning rate	Seed
State	300	8	3×10^{-4}	2101
State	1,200	8	3×10^{-4}	2101
State	1,500	8	1.5×10^{-4}	2101
State	1,500	8	1.2×10^{-4}	2101
Rollout	600	2	5×10^{-5}	2101
Copy	800	8	8×10^{-5}	4601
State (160 events)	1,000	8	6×10^{-5}	4701
Rollout (H240)	240	4	3×10^{-5}	20260921
Mixed (M120)	120	8	10^{-5}	20260929

B.2 Paired histories and retained checkpoints

In the main figures, "mixed phase" denotes the fixed M120 memory checkpoint `mixed120` after 120 final mixed-objective updates; "rollout phase" denotes H240, `rollouthier240`, after 240 final rollout-objective updates. They are successive checkpoints in one training lineage. A pair shares a 128-event prefix, then writes different values to the same key before the same overwrite or deletion. Readout exhausts all 64 keys, with values 0–7 and one absent category. Reconstruction starts a new cache and makes two deterministic passes over all 64 decoded key values. The main figures' "30-epoch run" label is S1, C-SWM seed 31001 at retained epoch 30; "90-epoch run" is S2, a separate run with seed 31002 at retained epoch 90. Each pair shares an eight-action prefix, then follows two legal two-step or four-step return paths for the same object in a five-object 5×5 grid. The fixed coordinate decoder is calibrated separately and frozen before the correction evaluation. The latent state is retained between updates, and the probe reads its output.

C-SWM uses the original small image encoder and a graph transition with hidden width 512, five object slots and two latent coordinates per slot. Training uses 2,000 episodes of 50 actions, Adam at 5×10^{-4} and batch size 1,024. The coordinate readout flattens the ten latent entries and uses an MLP with width 128 and five 25-class outputs. It is fitted on separate calibration scenes, with an 80/20 scene-level split, Adam at 10^{-3}, batch size 512 and up to 5,000 steps. Checkpoints at epochs 30 and 90 were selected by earlier eight-step development accuracy. These learned coordinates are decoded to grid cells only for reading and reconstruction; native updates retain the latent state.

B.3 Source accuracy and timing

Table 3: Source-correct unit counts on the same initially correct cohort. Each column requires both complete readouts to equal the source at its endpoint on all three suffixes. A trajectory may have made an earlier error and recovered. Every count in a row has denominator $|C_0|$.

		$H = 8$			$H = 16$		
Condition	$\lvert C_0\rvert$	Native	R-once	RFR	Native	R-once	RFR
M120 / delete	30	7	8	9	0	5	4
M120 / overwrite	30	7	11	4	2	5	2
H240 / delete	53	11	12	14	1	5	11
H240 / overwrite	42	11	14	13	4	5	7
S1 / 4-step	256	249	255	256	226	256	254
S1 / 2-step	255	249	253	254	225	254	254
S2 / 4-step	256	243	256	256	197	254	253
S2 / 2-step	256	250	256	254	204	256	252

Table 4: Paired $H = 16$ C-SWM endpoint outcomes for R-once versus Native. "Gain" and "Loss" count initially correct units that change the strict both-branch, all-suffix correctness result. "Gain, no split" is a subset of Gain with no Native branch disagreement on any of the three tested suffixes through 16 actions. These are descriptive counts on the same C_0 units as Table 3.

Condition	$\lvert C_0\rvert$	Gain	Loss	Gain, no split
S1 / 4-step	256	30	0	16
S1 / 2-step	255	29	0	10
S2 / 4-step	256	57	0	26
S2 / 2-step	256	52	0	36

For a cohort of size n with B native units correct on both branches, mean single-branch correctness on the same three suffix endpoints is at most $(n + B)/(2n)$: each of the other $n - B$ units can contribute at most one correct branch. In S2/4-step at $H = 16$, $B = 197$ and $n = 256$, giving 88.48%; R-once's identical branches are correct in 254/256 units, or 99.22% per branch.

To distinguish an initially correct pair that separates at its first error from a pair that has already made a common error, we replay all original native trajectories and retain the readout at each action. Let t_e be the first source error on either branch and t_s their first separation. A separating pair necessarily has $t_e \leq t_s$. When $t_e = t_s$, both branches were correct at every preceding step. When $t_e < t_s$, they first made a common error. Endpoint accuracy alone does not determine this order.

Table 5 counts each C_0 unit once. For a separating unit, we use the suffix with the earliest separation, breaking ties by suffix index. For a unit without separation, we check for any error on its three suffixes. The first-split column counts selected separating units for which exactly one branch still gives the complete source state at that split. The four timing categories partition the cohort. The replay preserves the original memory batches of 32 and C-SWM batches of 256. All 92,160 comparisons with stored initial flags, first-separation times and horizon metrics match exactly. The timelines add a diagnostic on the same units.

Table 5: Error and separation timing through 16 actions. "At split" means all previous answers on the selected suffix were correct; "Earlier" means a shared error preceded separation. The next two columns have no separation on any suffix: at least one error, or all answers correct throughout. "One correct" counts separating units with exactly one source-correct branch at the split.

		Separation		No separation		
Condition	$\lvert C_0\rvert$	At split	Earlier	Error	Correct	One correct
M120 / delete	30	16	11	3	0	19
M120 / overwrite	30	11	13	6	0	16
H240 / delete	53	21	24	8	0	28
H240 / overwrite	42	12	12	17	1	17
S1 / 4-step	256	20	1	26	209	21
S1 / 2-step	255	18	6	16	215	22
S2 / 4-step	256	35	8	48	165	40
S2 / 2-step	256	17	5	45	189	22

B.4 Reconstruction and local prediction changes

The original-start RFR measurement is $\widehat{\mathcal{F}}_{a_H} \cdots \widehat{\mathcal{F}}_{a_1}(z)$. An earlier evaluation applied an additional initial $\mathcal{R}$. We recomputed the intended operator on the same 6,144 unit–suffix rows; these are paired corrections of previously evaluated units. Eight memory E_0 units change readout on a second reconstruction, so the extra application matters to the operator definition. The corrected evaluation retains the same closure and $H = 8$ source-correctness counts.

The prediction-change calculation compares the original (native) and repaired updates from the *same local pre-action states*. The 1,900 local records correspond to 1,013 unit-within-condition incidences. At the original handoff, 1,572 records start in E_0 and 350 in C_0. All 1,900 have equal readouts at the selected local pre-action state; 321 also have both readouts source-correct there. Sixteen records are tight, of which only one starts in C_0. That retained H240/delete example is unit 248, suffix 0, depth 3; both complete readouts are also correct immediately before separation. It is the memory pair described at the start of the Introduction. After a common prefix, its branches execute `put(62,6); delete(62)` and `put(62,0); delete(62)`. Both then execute `copy(62,2)`, `copy(45,25)`, and `put(18,2)`, where `copy(a,b)` writes the value at key a to key b. All 64 answers remain correct after each of the first two continuation actions. The third action should leave keys 31 and 51 at (absent,4), but the branches read (absent,2) and (4,4), respectively. Every other key is correct in both branches. Its smallest retained relevant margin is 0.01887 and largest full-run versus replay logit change is 8.58×10^{-6}.

The return-path example in the Introduction is C-SWM seed 31002, two-step paths, unit 70, suffix 0. Object 4 returns to the same position after moving right then left, or up then down. Both complete position readouts are correct at this point and after each of the first two common continuation actions. At the third action they still agree but are jointly wrong; at the fourth they first differ, predicting object 0 at $(0, 1)$ and $(1, 1)$, respectively. The true position is $(0, 1)$. Coordinates are (row, column). This selected example illustrates both a common error and a later disagreement along one continuation.

An invalid reconstructed C-SWM configuration enters an explicit absorbing failure state, observable as such, independently on each branch. The three observed unit-within-condition failure incidences are bilateral. One local prediction-change record is bilateral failure; its bound uses the totalized readout space. Source correctness in Figure 3 counts failure as incorrect.

The numerical diagnostic replays 35 trajectories selected by a frozen rule, including each condition's earliest C_0 separation, all tight records, non-idempotent reconstructions and failure cases. All discrete results agree over three repeats. Thirty-one trajectories meet the strict margin/variation criterion; one additional trajectory is discrete-stable without saved separation logits; three C-SWM trajectories are sensitive to the batch path. The preselected earliest C_0 witness is strictly stable in seven of eight conditions. The common threshold 1.2398×10^{-4} was calibrated on memory batch sizes 16 and 32; no separate C-SWM floor was calibrated.

The native curves use pointwise Wilson intervals within conditions; repair points are descriptive proportions with exact counts in Table 1. The plotting script recomputes these counts from the retained records; the main-paper reproducibility statement describes artifact availability.

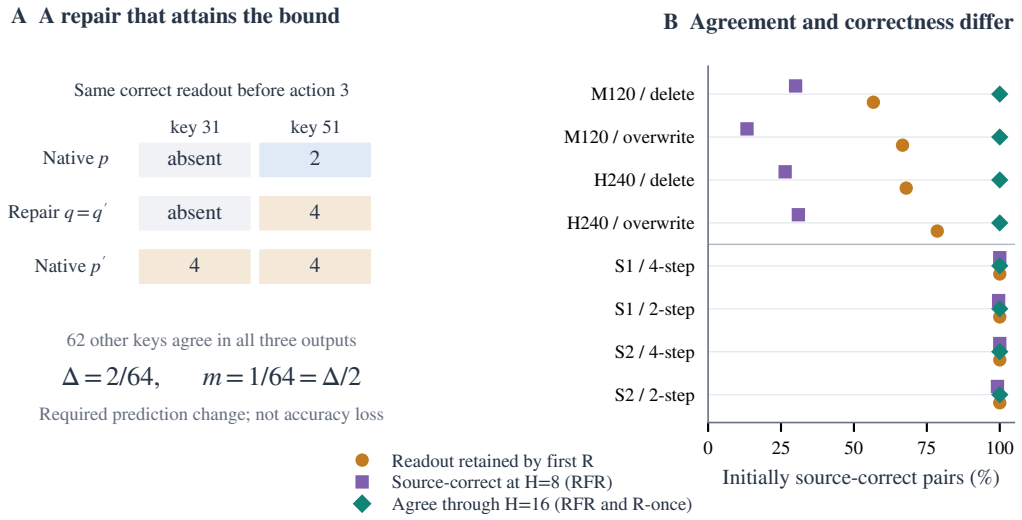

Figure 4: **Agreement, answer preservation and correctness separate.** (A) At action 3, the native memory branches disagree on two of 64 keys; the common repaired output changes one key relative to each. All other keys agree. "Absent" is a deleted association; 2 and 4 are values. This selected example is the only initially source-correct witness among 16 tight records. (B) Initially source-correct pairs: current answers preserved by the first reconstruction; both branches source-correct on all three suffixes at $H = 8$; and no separation on any suffix through $H = 16$. RFR reconstructs at every update, R-once only at handoff. Both reach 100% agreement, including bilateral C-SWM failures. Points are descriptive proportions; Table 1 gives exact counts.

Restoring agreement changes at least one prediction. At an action where native updates first produce different readouts from equal pre-action readouts, let the two native predictions be p, p' and let a closed repair give their common prediction q. We recompute this repaired step from those same two native pre-action states, rather than reading q from an independently evolved repaired trajectory. For a readout metric d, the triangle inequality gives

$$m := \max\{d(p,q), d(p',q)\} \;\geq\; \tfrac{1}{2}d(p,p') =: \Delta/2. \tag{13}$$

All 1,900 retained local records satisfy this bound under normalized Hamming distance, with one record interpreted in the space containing the failure state. These records evaluate the same-input bound of Proposition 3. Sixteen memory records attain the lower bound exactly. The source-correct example in Figure 4A has $\Delta = 2/64$ and $m = 1/64$, and is stable under numerical replay. Here the common repaired output is source-correct: changing one key relative to each native prediction removes their different errors. The bound concerns the unavoidable change in predictions; its effect on source accuracy depends on which common output the repair selects.

C Nominal replacement transducer: interface and construction

C.1 Identifier and slot transformations

Fix a record capacity R, and let X be the corresponding typed decoded state space, U a set of action descriptions, and Q a family of queries. The source implements partial transitions $T_u : X \rightharpoonup X$ and query functions $q : X \to Y_q$, where Y_q is the answer space for q. The learned one-step network, with parameters θ, produces successor-state logits $F_\theta(x, u)$, and a deterministic decoder δ

maps those logits back into X. The network accepts typed (x, u) even if a prediction has made u illegal under the source rules; the source's partial transition and the network's defined update are not identified. We write

$$M_\theta(x, u) = \delta(F_\theta(x, u)).$$

Closed execution feeds each decoded prediction back into the next step. For a source-legal action sequence (word) $w = u_1 \cdots u_d$, define $x_0 = x$ and $x_t = M_\theta(x_{t-1}, u_t)$, and write $M_\theta^w(x) = x_d$; define $T_w(x)$ analogously for the exact source. For the discrete queries used here, prediction error is

$$\ell_q(x, w; \theta) = \mathbf{1}\{q(M_\theta^w(x)) \neq q(T_w(x))\}, \qquad q \in Q. \tag{14}$$

Ordinary query accuracy is one minus the mean of this indicator over the evaluated queries. Prediction receives no successor observation from the source after the initial state is given. Thus late-query evaluation reads the retained decoded state with the same query semantics as the source; there is no additional learned query decoder in this comparison.

We treat identifier strings as nominal atoms: their spelling carries no semantic category. Let A be a countably infinite atom set, let $\perp \notin A$ denote the null name, and let G be the group of finite permutations of A, each fixing $\perp$. Consistent renaming induces actions on states and actions. Internal record slots are also arbitrary. Let S_R be the group of record-slot permutations; it reindexes every record axis and any allocation-slot coordinate. For $g = (\sigma, \pi) \in S_R \times G$, σ changes storage coordinates while π renames public identifiers. The semantic requirement on a decoded one-step update is

$$M_\theta(g \cdot x, g \cdot u) = g \cdot M_\theta(x, u). \tag{15}$$

This property is *joint equivariance* under identifier and slot permutations. Equation 15 says that either representation may be used to execute the update and then undone without changing the represented successor state.

For a word $w = u_1 \cdots u_d$, let $g \cdot w = (g \cdot u_1) \cdots (g \cdot u_d)$. A paired state intervention checks whether

$$g^{-1} \cdot M_\theta^{g \cdot w}(g \cdot x) = M_\theta^w(x).$$

We evaluate identifier renaming and slot reindexing separately: the former renames all identifiers in the initial state and actions, while the latter reindexes record axes and action-allocation coordinates. After undoing the transformation, we apply the same original $q \in Q$ to both retained states and check $q(g^{-1} \cdot M_\theta^{g \cdot w}(g \cdot x)) = q(M_\theta^w(x))$. Whole-state covariance is the fraction of paired episodes satisfying the state equality above; query covariance is the fraction of paired query instances satisfying the answer equality. The distinction matters because a wrong retained store can still answer a particular query correctly. Whole-state covariance compares the entire predicted state; query covariance compares only the answer requested by q. Neither metric measures ordinary predictive correctness against the exact source, which is evaluated separately by Eq. 14.

Source legality is determined along the ground-truth trajectory when episodes are generated; it is not a promise that a learned trajectory remains source-valid. For corresponding inputs, the source rules satisfy $T_{g \cdot u}(g \cdot x) = g \cdot T_u(x)$ whenever either side is defined, and legality agrees on both sides. A transported query q^g obeys $q^g(g \cdot x) = q(x)$. These conditions explain why undoing the representation change preserves the ground-truth answers used in the paired test.

C.2 Architecture

Equation 15 can fail through two representation shortcuts: a learner can use the numeric encoding of an identifier, or the absolute position of a serialized record. NRT removes the first shortcut by exposing names only through equality and pointers to available names, and the second by using shared set and pairwise operations without record positions.

The state has a fixed number of record slots. Each record has an existence flag, one public name, finite categorical attributes, and directed links. An action contains an operation kind from a finite set, up to three identifier arguments, optional finite key/value fields, and an allocation-slot indicator that reindexes with the record slots.

Identifier-free learned features. The model never embeds a numeric identifier token. For each record it receives existence and finite attributes plus four Boolean relations: equality of the record

name with each of three identifier arguments, and equality of the record slot with the action's allocation slot. Global action features contain operation/key/value fields, equality among identifier arguments, and allocation status. The absolute allocation coordinate is withheld.

A Transformer encoder without record positions processes the record set. Shared local heads predict existence and finite attributes. Ordered-pair heads receive contextual target/source features, links in both directions, target/source binding features, the target-source slot diagonal, and global action features. They predict links and scores for pointers to names in the previous state.

Names copied from the state or action. For each target record, NRT scores candidate name sources

$$\{\bot\} \cup \{\text{prior record names}\} \cup \{\text{current action-name arguments}\}. \tag{16}$$

Probabilities of pointers carrying the same identifier are summed into categorical name probabilities. A new identifier for create or rename must therefore be supplied by the action. Writing $\mathrm{supp}(z)$ for the non-null identifier atoms occurring in a state or action z, the output obeys

$$\mathrm{supp}(M_\theta(x,u)) \subseteq \mathrm{supp}(x) \cup \mathrm{supp}(u), \tag{17}$$

When one name category has a unique maximum, deterministic decoding selects it. A tie between maximal name logits maps to null $\bot$, the fixed point of atom permutations. Numeric `argmax` would otherwise break covariance by selecting the lowest token index in a symmetry-induced tie.

The following statement uses deterministic evaluation in exact arithmetic on the typed interface $X \times U$. It concerns the decoded model update, whether or not that prediction is a semantically valid source state.

Theorem 1 (Exact product equivariance)**.** *For every θ, $\sigma \in S_R$, $\pi \in G$, state $x \in X$, and typed action $u \in U$, the decoded update M_θ satisfies*

$$M_\theta((\sigma,\pi)\cdot x, (\sigma,\pi)\cdot u) = (\sigma,\pi)\cdot M_\theta(x,u). \tag{18}$$

It also satisfies Eq. 17. Consequently, for every finite source-legal word $w = u_1 \cdots u_d$ and $g = (\sigma,\pi)$,

$$M_\theta^{g\cdot w}(g\cdot x) = g\cdot M_\theta^w(x), \qquad \mathrm{supp}(M_\theta^w(x)) \subseteq \mathrm{supp}(x) \cup \bigcup_{t=1}^{d} \mathrm{supp}(u_t).$$

Finite semantic features and all name-equality predicates are fixed by atom permutations. Shared record layers are slot equivariant. Pair tensors are reindexed along both record axes and expose no absolute slot index. Pointer scores are therefore atom invariant and slot equivariant; renaming their sources permutes the induced non-null categorical logits. Unique maxima follow that permutation, while tied maxima decode to the fixed null symbol. Every non-null output originates from a state or action pointer. Composition proves the rollout result. Appendix D gives the full argument.

D Proof of the joint equivariance guarantee

Write a state as (e, n, a, ℓ) for existence, name atoms, finite attributes, and links. The G action changes only n and identifier arguments of u. Every name-dependent learned input is a predicate $[n_i = u_j]$ or $[u_j = u_k]$, hence is fixed under simultaneous $\pi \in G$. Other semantic inputs are finite categorical fields fixed by G. The S_R action permutes record axes, both axes of ℓ, and the allocation indicator.

The input projection is shared over records. Self-attention without record positions, the feed-forward blocks, layer normalization, and shared local heads commute with S_R. Pair construction is equivariant under S_R: left/right contextual tensors, ℓ_{ij}, ℓ_{ji}, the target/source binding predicates, and the diagonal $[i = j]$ all transform by $(i,j) \mapsto (\sigma i, \sigma j)$. The shared pair heads therefore commute with S_R. All of these logits are fixed by G.

For target i, let s range over the null source, prior-record name sources, and current action-name sources. Let p_{is} be the score for source s and $c_{x,u}(s) \in A \cup \{\bot\}$ the name carried by that source. The categorical probability for atom b is

$$P_i(b) = \sum_{s : c_{x,u}(s) = b} \mathrm{softmax}(p_i)_s.$$

The scores do not change under π, while $c_{\pi x,\pi u}(s) = \pi(c_{x,u}(s))$. Thus $P_i^{\pi x,\pi u}(\pi b) = P_i^{x,u}(b)$. A unique maximum follows π. If maxima tie, the decoder emits $\perp$, fixed by G. Every non-null atom with positive probability therefore has a pointer source carrying it in the prior state or current action, proving the support law. Under σ, target and prior-record source axes are permuted together, so the same pointer construction is slot equivariant. The existence mask, finite attributes, and links commute with the relevant actions, including removal of self-links. Induction over the decoded-feedback recurrence defined in Section C.1 proves both rollout statements.

The calculation uses the typed state/action fields, not source legality, so the one-step correspondence also holds at an incorrect intermediate prediction. This is the step needed to apply induction along the learned trajectory. The algebraic guarantee concerns deterministic evaluation in exact arithmetic. The decoder preserves the typed tensor interface, but does not certify source invariants such as semantic validity of every predicted record. The fixed null tie rule, existence masking and removal of self-links all commute with the declared transformations. The empirical covariance tests separately check the floating-point implementation on the registered inputs.

E ARCHITECTURAL EVALUATION: DOMAINS AND PROTOCOL

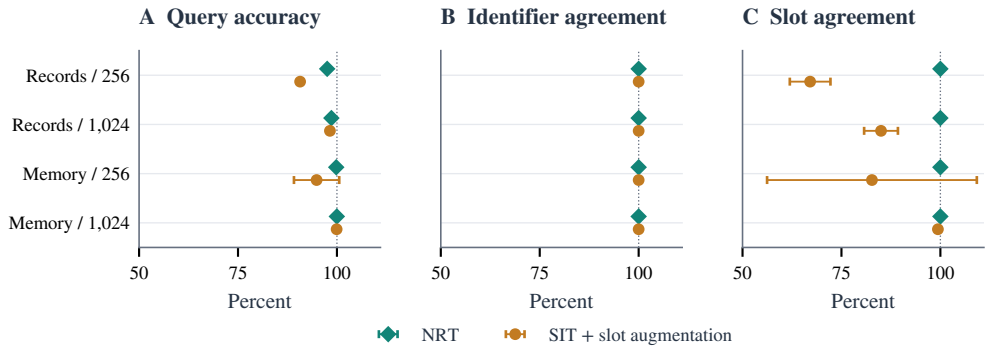

Figure 5: **Identifier agreement does not guarantee storage agreement.** Points show means and whiskers one sample standard deviation over five prespecified model seeds. Both models retain high ordinary query accuracy and exact identifier covariance. Covariance measures agreement after undoing the corresponding transformation. Under record-slot reindexing, NRT remains exact by construction, while the slot-augmented sequence adapter remains inexact in these two families. Agreement uses the whole decoded state; query accuracy uses source answers. Horizontal axes display 50–111% to retain the full descriptive standard-deviation whiskers; 100% is the maximum possible observed rate.

A separate fixed-vocabulary control provides a simple accuracy contrast. In Memory / 1,024, its source-query accuracy is 99.96% under the original identifier codes and 69.58% after code renaming; NRT retains 99.97% in both conditions. The intervention acts after the identifier codec, so it tests dependence on the numeric codes actually used by the updater. Appendix F reports the control comparison.

The storage difference also depends on the transition family. On five-object AIPS-2000 Blocksworld with 64 training transitions, both models reach 100% query accuracy, final-state accuracy and covariance under both interventions, at every tested depth. Here slot augmentation learns the required agreement. The architectural law supplies the same correspondence independently of training outcomes; source accuracy remains an empirical property of the learned update.

E.1 EXACT STATE FAMILIES

The main families exercise both identifier codes and record positions through repeated updates: public names can change while record allocation, deletion, links, and tier membership make storage state consequential across later actions.

The **record-store** family contains records with stable identities and mutable public names, typed attributes, and directed links. Programs cover rename, set, reset, link, unlink, delete, create, and copy. Queries ask existence, attributes, link membership, linked-name sets, and attribute counts.

The **tiered-memory** family contains key/value entries in core and archival tiers. Programs use core/archival add, remove, and replace, core clear, and promote. Queries retrieve per-tier values and keys or count entries by tier. In both families, action order, targets, values, initial relations, allocation roles, and names vary by seed.

Training examples are individual one-step transitions, called *primitive transitions* in the protocol. In the main comparison, $N \in \{256, 1024\}$ denotes the number of sampled one-step training transitions. The corresponding unique counts are 254/1,012 for records and 256/1,004 for memory. Rollout-evaluation episodes compose 2, 3, 5, or 8 actions on fresh states. The model receives the full encoded initial state once; after each action, its own decoded successor becomes the input state for the next action. The exact source advances in parallel only to generate ground truth and never supplies a successor observation to the model during rollout.

To test another domain, we implement the exact four-operator AIPS-2000 Blocksworld semantics from the FF Domain Collection (Hoffmann, 2000). The source uses `pickup`, `putdown`, `stack`, and `unstack`; late queries exhaust the five predicate families instantiated with specific objects. Models receive actions with object arguments filled in and predict their effects on the state. Its evaluation protocol is fixed separately from the two-family comparison.

E.2 MODELS AND OPTIMIZATION

The main comparison uses two approximately 90K-parameter models. **NRT** is the proposed set/pointer updater with joint identifier and slot equivariance. **Symbol-Invariant Transformer (SIT) adapter** adapts the public construction of Işık & Li (2026): a shared Transformer pipeline views the serialization from the perspective of each identifier, symmetric aggregation predicts finite fields, and each identifier's pipeline supplies its output logit. Names absent from the state and action are masked. This makes identifier covariance exact. We add random record-slot permutation at every training step as the direct remedy for its positional serialization. The adapter uses structured successor-state heads so both models predict the same state fields. This is a task adaptation of the predecessor's mechanism, whose original encoder–decoder targets linear temporal logic (LTL).

Within a seed, models receive identical transitions and minibatch indices. Parameter spread is below 0.7%. Record models train for 6,000 Adam steps; memory models train for 4,000. The sequence baseline's record learning rate, selected on separate development data, is 7.5×10^{-4}; all others use 10^{-3}. Batch size is 64 with the same changed-component loss weight. We use nested sets of 256 and 1,024 transitions and require adequate accuracy on components changed by a one-step transition (*primitive fit*) before comparing closed rollouts. Appendix F reports a separate comparison with a model using a fixed name vocabulary, plain set attention, and a flat model.

E.3 EVALUATION PROTOCOL

Architecture and optimizer choices used separate development runs. Before the final comparison, we fixed data seeds, one unseen identifier-permutation seed, one slot-permutation seed, five model seeds, and 96 episodes per combination of state family and training-set size (a *cell*). We also fixed the metrics, thresholds, and a no-rerun policy. The primary success rule requires: every primitive fit is at least 0.95; both models have exact identifier covariance; NRT has exact slot covariance; NRT's mean ordinary accuracy is no more than two percentage points below the sequence baseline in every cell; and the augmented baseline's mean whole-state slot covariance is at most 0.99 in both low-data cells. We report ordinary-accuracy differences descriptively.

Empirical covariance is the proportion of paired evaluations with equal outputs after undoing the transformation. Theorem 1 provides the corresponding all-input guarantee for NRT in exact arithmetic.

Protocol development and the earlier comparison are recorded in Appendix H.

E.4 Per-condition results

Table 6: Final comparison in percent (mean ± sample standard deviation; five seeds). Covariance is exact agreement of the whole decoded state after undoing the paired intervention. SIT denotes the slot-augmented symbol-invariant sequence adapter.

Family	N	Ordinary query		Identifier state cov.		Slot state cov.	
		NRT	SIT	NRT	SIT	NRT	SIT
Records	256	97.53±.20	90.70±.24	100±0	100±0	100±0	67.08±5.13
Records	1,024	98.60±.32	98.23±.63	100±0	100±0	100±0	85.00±4.27
Memory	256	99.82±.14	94.86±5.72	100±0	100±0	100±0	82.71±26.50
Memory	1,024	99.97±.04	99.92±.03	100±0	100±0	100±0	99.38±.57

The minimum changed-component training fit is 99.79% for NRT and 97.76% for SIT. SIT's query covariance under slot reindexing is 96.92%, 99.10%, 97.62% and 99.96% in table order. The paired ordinary-accuracy differences (NRT minus SIT) are $6.84\pm.20$, $.37\pm.71$, 4.96 ± 5.78 and $.05\pm.03$ percentage points.

Final record models have 91,095 (NRT) and 90,578 (SIT) parameters; memory models have 89,807 and 90,350. Each condition evaluates 24 episodes at each depth in $\{2, 3, 5, 8\}$. The identifier and slot interventions use the same 96 underlying episodes as ordinary evaluation. Across four conditions and five seeds, this gives 1,920 paired episodes and 61,210 late queries per intervention. The separate Blocksworld evaluation has 480 paired episodes and 17,280 queries per intervention over five fresh seeds.

E.5 Direct structured and canonicalized alternatives

The preceding comparison isolates a positional sequence model trained with slot augmentation. We additionally compare two implementations that preserve the known correspondence by construction. All three updaters predict the same state fields and receive the same state and action information.

Structured SIT. This adaptation retains SIT's focal-symbol pipelines and output heads. It removes record-position embeddings, ordered adjacency-vector features and the absolute allocation one-hot vector. Each record instead receives four local role indicators: whether its name matches each of the three action arguments and whether it is the allocation target. The action carries only allocation status. A shared linear layer combines incoming and outgoing graph contractions of the record embeddings, after which a Transformer without positions processes the record and action tokens. Finite fields aggregate over symbol pipelines; name logits come from their respective pipelines, with unsupported names masked. An ordered-pair head predicts links from the two record embeddings and their current forward/backward links. Thus attributes, action roles and directed relations remain available without absolute record coordinates.

Canonical SIT. The sequence backbone retains its positions and ordered features. At every rollout step, we jointly canonicalize the state and action, predict in those coordinates, and map the output back. Canonicalization covers identifier arguments, the allocation coordinate and both link axes. It orders by typed content, relations and equality patterns, rather than by numeric identifier values. Color refinement partitions records using these invariant signatures. We search all remaining orders within color classes, assign canonical names by first occurrence in the ordered action arguments and records, and select the smallest resulting state/action encoding. The training target is transported using the same input-derived map. No successor information chooses the map.

A symmetric input can have several minimizing maps. Selecting one and simply inverting its output would leave this ambiguity unresolved. We therefore average output coordinates over the canonical input's stabilizer, following the standard frame-averaging principle (Puny et al., 2022). Names are averaged as probabilities; finite-field and link outputs are averaged as logits. For these permutation actions, averaging each coordinate orbit is equivalent to averaging the full stabilizer. Identical null slots are handled by adjacent-swap generators rather than factorial enumeration. Unsupported names have zero probability. This requires one backbone evaluation per canonical input, although searching the remaining record orders has factorial worst-case cost.

Paired training and evaluation. The new comparison trains all three models on the same fresh examples and minibatch schedules, with five model seeds, two data sizes and both domains. It retains the earlier Adam settings: 6,000 updates for records, 4,000 for memory, batch size 64 and changed-component weight 5. The SIT variants use hidden width 56, four heads and two layers. Canonical SIT retains feedforward widths 168/192; Structured SIT uses 142/167 to offset its graph projection when matching parameter counts. Records have 91,095/90,582/90,578 parameters and memory has 89,807/90,468/90,350 for NRT/Structured SIT/Canonical SIT, respectively. No augmentation is needed by these three constructions.

Here N counts sampled transitions. The two record sets contain 256/1,011 unique transitions and the memory sets 255/1,007, out of $N = 256/1,024$; all methods share these repetitions. Training and evaluation seeds are 39601/39611 for records and 49601/49611 for memory, model seeds are 370–374, and name/slot transformation seeds are 59601/59611. Each domain shares its 96 evaluation fixtures across models, data sizes and seeds, with 24 at each depth in $\{2, 3, 5, 8\}$, under identity, name, slot and joint transformations. Successors are always the model's own decoded predictions. Source states supply training labels and evaluation truth only.

Before these runs, separate apparatus fixtures tested joint transport and symmetric cases, including duplicate names, cycles and allocation into empty slots. FP32 reduction order can perturb mathematically tied name scores. Every method in this follow-up uses the same 10^{-5} absolute name-logit tie band; a tied maximum decodes to null. This comparison uses the original 95% changed-component training-fit threshold and retains every completed seed.

Source-state accuracy compares the full decoded array in the recorded allocation coordinates. The scoring codec binds each newly created source identity to its allocation slot immediately, including identities deleted before the final query; these scoring observations are never model inputs.

Table 7: Direct alternatives on fresh shared fixtures. Percent, mean ± sample standard deviation over five model seeds. Query and full-state accuracy compare with the source; joint covariance compares paired predictions after undoing both transformations. Separate name and slot covariances are also 100% in every run. Min. fit is the minimum changed-component training accuracy across the five seeds.

Family	N	Updater	Query accuracy	State accuracy	Joint cov.	Min. fit
Records	256	NRT	97.37±0.20	62.92±2.51	100	100.00
Records	256	Structured SIT	98.30±0.23	72.71±2.89	100	100.00
Records	256	Canonical SIT	86.15±0.69	11.67±1.86	100	100.00
Records	1,024	NRT	98.69±0.15	75.21±1.36	100	99.95
Records	1,024	Structured SIT	99.85±0.07	97.50±0.93	100	100.00
Records	1,024	Canonical SIT	95.99±0.79	56.25±4.17	100	100.00
Memory	256	NRT	99.24±0.32	91.46±2.70	100	100.00
Memory	256	Structured SIT	99.94±0.00	98.96±0.00	100	100.00
Memory	256	Canonical SIT	86.31±1.49	28.96±3.49	100	100.00
Memory	1,024	NRT	99.85±0.06	97.92±0.00	100	100.00
Memory	1,024	Structured SIT	100.00±0.00	100.00±0.00	100	100.00
Memory	1,024	Canonical SIT	99.54±0.16	94.17±2.28	100	100.00

All 60 runs pass the training-fit gate. Structured SIT improves mean query accuracy over NRT by 0.94, 1.16, 0.70 and 0.15 percentage points in table order; it also has higher mean full-state accuracy in each condition. Canonical SIT maintains the same correspondence, while its predictive accuracy depends more strongly on the training-set size in this recipe. These comparisons provide multiple realizations of the interface law, with different predictive behavior.

F CONVENTIONAL CONTROLS

Table 8 is the separate control comparison against controls with a fixed name vocabulary (vocabulary-bound controls). Matched set receives the same equality indicators as NRT but predicts names with a classifier over a fixed vocabulary. Plain set removes the binding indicators; flat serializes the state. All use about 90K parameters, identical training examples and within-seed schedules, 4,000 steps, and model seeds 40–44.

Table 8: Vocabulary-bound control comparison in percent (mean ± sample standard deviation over five seeds). Covariances use the identifier intervention.

Family	N	Ordinary query		Query cov.		State cov.	
		NRT	Matched	NRT	Matched	NRT	Matched
Records	256	96.75±.22	94.55±.34	100±0	75.24±.46	100±0	0±0
Records	1,024	98.60±.26	98.50±.25	100±0	75.31±.45	100±0	0±0
Memory	256	99.61±.12	99.19±.16	100±0	68.66±.52	100±0	2.08±0
Memory	1,024	99.97±.04	99.96±.04	100±0	69.58±.25	100±0	2.29±.47

The same control comparison includes plain set and flat controls. NRT minus plain-set ordinary accuracy is 20.25±.89, 18.29±.65, 17.66±1.75, and 15.75±2.22 points in table order.

G DEPTH-STRATIFIED ORDINARY ACCURACY

Table 9: Final-comparison query accuracy by closed-rollout depth (percent means over five seeds).

Family	N	Model	$d = 2$	$d = 3$	$d = 5$	$d = 8$
Records	256	NRT	97.97	98.06	97.20	97.26
		SIT	92.50	92.74	90.06	89.02
Records	1,024	NRT	98.81	99.09	98.92	97.95
		SIT	98.71	97.97	98.81	97.66
Memory	256	NRT	99.95	99.89	99.95	99.54
		SIT	96.98	97.07	93.20	92.73
Memory	1,024	NRT	100	100	100	99.91
		SIT	100	100	100	99.72

H PROTOCOL DEVELOPMENT

An earlier architectural comparison failed its combined success rule. One record-$N = 256$ SIT seed had 93.39% changed-component training accuracy, below the 95% eligibility threshold. The memory-$N = 256$ ordinary-accuracy advantage was 2.015 points, below the required 2.5-point advantage in both families. Subsequent development runs selected a stable SIT record learning rate. The final comparison then fixed a non-inferiority and joint-covariance criterion before evaluation, using disjoint data, model and permutation seeds. It uses data seeds 6901/7901 and 8901/9901, model seeds 70–74, identifier-permutation seed 10901 and slot-permutation seed 11901. Earlier outcomes are excluded from its reported results.