How a World Model Forms: Separating World Knowledge from Behavioral Competence in Sokoban Transformers

Anonymous authors
Paper under double-blind review

Abstract

Does teaching a policy to represent its environment more accurately make it better at acting? We study this question in Sokoban Transformers that receive an initial board and move history, without updated observations. Eight training conditions vary supervision of state, action effects, and goal occupancy while sharing the same action targets and trajectory corpus. We measure what probes recover, which state distinctions guide decisions, and whether the policy solves. Action prediction alone yields accurate action-effect readouts and decisions that distinguish remote states, despite incomplete state reconstruction. Across three seeds, supervising all three components raises complete-state recovery from 0.208 to 0.955, while solving on held-out puzzles with optimal distances of 32–60 changes from 0.745 to 0.735. Supervision does help when policies must continue from supplied detour histories, but doubling training in both conditions reduces this advantage, leaving training efficiency as a plausible contributor. Together, these results show why complete-state recovery cannot stand in for behaviorally useful world knowledge. Its value must be tested through the decisions and continuations it is intended to support.

1 Introduction

World-model research approaches the link between knowledge and action from two directions. Learned environment models support control by predicting what happens next (Ha & Schmidhuber, 2018; Hafner et al., 2019). Studies of sequence models ask whether such knowledge emerges without explicit environment-prediction targets: Othello and chess models can encode board states from move histories (Li et al., 2023; Nanda et al., 2023b; Toshniwal et al., 2022; Karvonen, 2024). These findings raise a further question: when a policy already acts well, does teaching it to represent the environment more explicitly improve its behavior?

An accurate board readout is evidence about the representation, but its value for action requires a separate test. Earlier Othello studies already use interventions to connect board representations to move predictions (Li et al., 2023; Nanda et al., 2023b). Conversely, models such as MuZero learn quantities useful for planning without reconstructing observations (Schrittwieser et al., 2020). These results leave open how explicit knowledge supervision changes a policy's ability to solve a task, and when a gain in recoverable knowledge translates into a behavioral gain. We examine this relationship across training objectives and evaluation settings.

Sokoban makes the comparison concrete. The policy sees only an initial board and its move history; a misplaced push can make a puzzle irreversibly unsolvable. We use *world knowledge* for information about the environment's state, action consequences, and goal configuration. Our measurements address three specific components: **state** (player and box positions), **action effects** (walk, push, or stay), and **goal occupancy**. Probes test whether these quantities can be recovered from hidden representations. Behavioral tests ask whether the policy distinguishes relevant states and solves puzzles, including after detours. Neither measurement defines all the knowledge the policy might use.

The experiment varies component supervision in a complete 2^3 design with three seeds per condition (Figure 2). Every policy learns the same optimal-action targets from paired optimal and successful-detour trajectories. This shared corpus exposes different histories while preserving a valid target at

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.

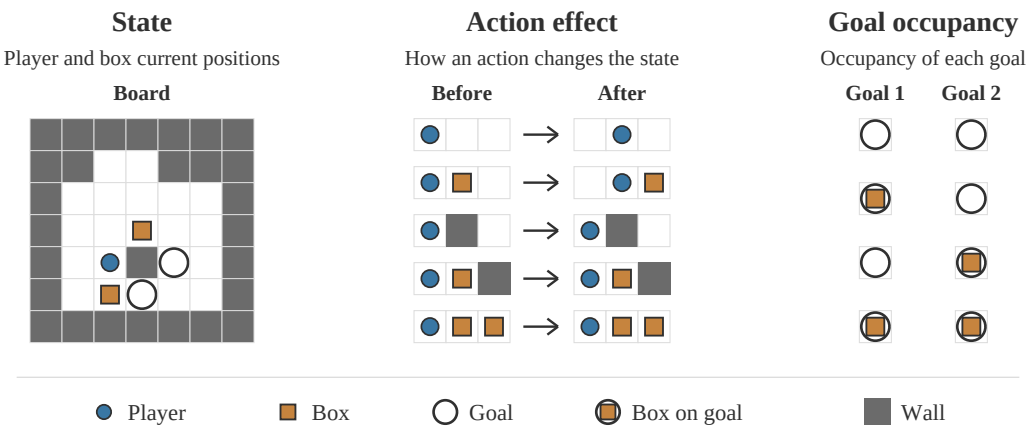

Figure 1: Three measured components: state (current player/box positions), action effects, and goal occupancy. The model receives only the initial board and action history. Before/after pairs show a rightward action in five physical contexts: walking, pushing, and three blocked cases yielding stay. The goal panel shows four independent occupancy combinations for Goal 1 and Goal 2; both occupied means solved. Boxes outside the goal cells are omitted.

every position. Architecture and update count are held fixed; auxiliary computation and gradients are not. We then compare readouts, state-sensitive decisions, and solving across original starts and supplied histories.

The resulting picture is conditional. Supervision sharply improves complete-state recovery, yet does not consistently improve solving from original starts. It does improve continuation after supplied detour histories, although longer training reduces that advantage. Action-only policies distinguish remote states despite imperfect recovery. Our contribution is this controlled account of *where knowledge supervision helps and where recovery alone is misleading*, rather than a claim that world knowledge is unnecessary. The results motivate evaluating a proposed world representation through the behavior it is meant to support (Sections 5 and 6).

2 Setting

Environment. Boards are 7×7 with a walled border, two boxes and two goals. The four directional actions have three effects across the five physical contexts in Figure 1: WALK, PUSH, or STAY. A puzzle is solved when both boxes occupy goals. A vectorized GPU engine enumerates the state graph to obtain exact transitions and optimal distances, which we check against a scalar reference (Section A).

Model and input. A 16-block, 50.5M-parameter decoder-only Transformer reads the 49 tiles of the *initial* board followed by the executed actions. It must use this history to act as the board changes; updated boards are never supplied. Section B specifies the architecture and tokenization.

Time alignment. After k moves, including blocked attempts, we measure the current state s_k from the representation that predicts the next action a_k. It can see only the initial board and $a_0, \ldots, a_{k-1}$. Readouts cover embeddings (layer 0) and all 16 block outputs.

3 Learning optimal actions from varied histories

To study knowledge beyond a single optimal plan, training must include varied histories. Unrestricted exploration, however, can make a puzzle unsolvable, leaving no optimal next action. We construct successful detours that preserve solvability and pair them with shortest solutions (Figure 7).

Paired trajectories. Each accepted start supplies an *optimal* trajectory A and a successful *detour* trajectory B; both enter training. A follows a fixed tie-broken shortest path. B takes a solvable, budget-feasible non-optimal action with probability $q = 0.5$ when available, otherwise an optimal action. It must solve, be longer than A, and contain a *state-changing* detour; blocked attempts alone do not qualify. Section A gives the rule.

Targets. At every position of A and B the target is the solver-optimal action of the *current* state. A detour therefore changes the history the model must interpret without teaching it to choose that non-optimal move. Every intermediate state remains solvable and has a target.

The frozen corpus. We group layouts by wall pattern up to the eight symmetries of the square, then hash groups into training, probe, and confirmatory partitions; development-panel groups are excluded from training. The corpus has 160,000 layouts and eight starts per layout, drawn uniformly from states at optimal distance 8–60. This gives 1,280,000 pairs and 2,560,000 trajectories across 26,571 groups. Optimal and detour trajectories are equally numerous, but detours are longer (48.64 versus 12.88 moves on average). They account for 79.1% of the 78,744,487 supervised positions (Section A).

Coverage. Even the rarest physical context, box against box, appears at least 54,816 times per direction. The audit finds no unsolvable position, invalid target, or replay mismatch. Section A reports all checks, including states reached through different histories.

4 Training conditions and measurements

We compare what the training objectives make recoverable with what the resulting policies can do. The core design preceded the main results; later probe revisions and diagnostic follow-ups are documented in Section I.

4.1 Training objectives

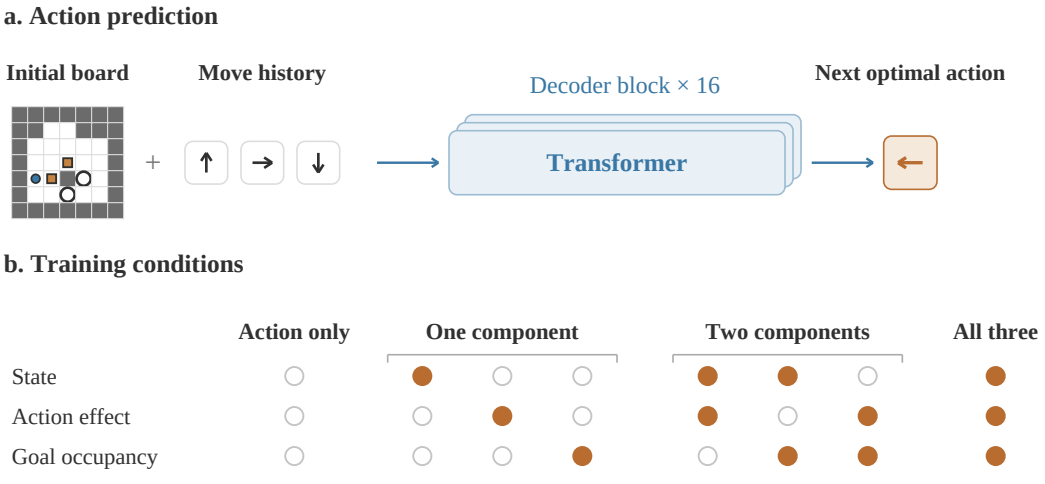

Figure 2: Training setup. (a) The Transformer predicts the highlighted optimal-action target from the initial board and move history. (b) Filled circles mark additional targets in eight component-supervision combinations. Both panels are schematic.

Every condition predicts the next solver-optimal action (Figure 2). *Action only* trains on this target alone. *State supervision*, *Action-effect supervision*, and *Goal-occupancy supervision* add the named ground-truth component. Three pairwise conditions and *All-component supervision* add two and all three components, respectively.

Conditions share the model, corpus, batches within each seed, optimizer, and 51,360 updates of 256 trajectories. Heads attach after block 8 with unit loss coefficients; gradients and compute are not matched (Section B). All eight main conditions and the additional objective controls have three seeds. Action only and all-component supervision have 28 checkpoints for seed 1 and 12 for each other seed; every condition has a final evaluation. Section B.3 describes an additional objective comparison.

4.2 World knowledge

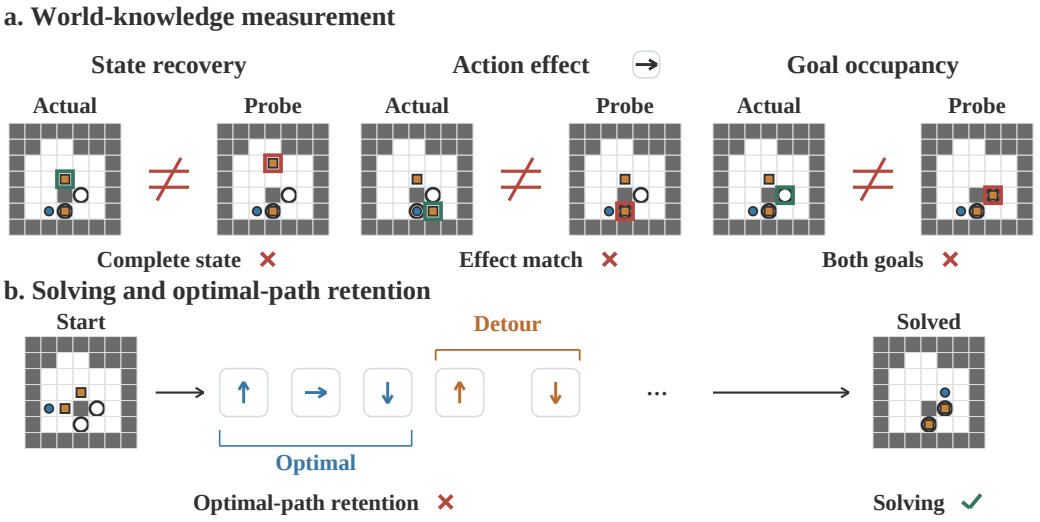

Figure 3: What the measurements count. (a) Actual targets and schematic probe predictions for all three world-knowledge components. Each comparison fails its criterion: one box is misplaced, a rightward push is decoded as stay, or one goal bit is wrong. Effect and goal boards visualize class/bit predictions; other board contents supply context. (b) Three blue optimal moves precede an orange detour: retention fails ($k = 4$), but solving succeeds. These are engine-verified schematics, not model outputs; reported rates aggregate over evaluation cases. Probes read frozen representations; predictions and updated boards do not feed the policy.

Probe data and targets. Separately fitted probes (Figure 3a) use 2,000 layouts and 32,000 trajectories split by structural group. We select primary linear-probe layers on validation data. Complete-state recovery requires the current player and both boxes to be correct: decoding selects the highest-scoring player cell and two box cells, excluding walls. An action-effect probe predicts stay, walk, or push for each candidate action, scored overall and by physical context. A goal-occupancy probe predicts one bit per goal; both must match. Section C details fitting and additional next-state and goal-entry tests.

Beyond linear readouts. To check what linear decoding misses, we fit a one-hidden-layer MLP to the same representation. An *attention-pooling probe* (CH-Wang et al., 2024) also accesses the board and move-history representations, adding their pooled scores to a current-token linear prediction (Figure 8 and Section C.1). Random-network and raw-embedding controls test what the probe can reconstruct itself. For these comparisons, maxima over test layers are descriptive.

4.3 Behavior and uncertainty

Evaluation. The frozen panel has 1,500 episodes: 500 each at initial optimal distances 8–15, 16–31, and 32–60, from previously unevaluated structural groups, one episode per group per bin; the earlier hard panel is development-only. Greedy rollouts (Figure 3b) use no legality masking and retain blocked moves. Optimal-path retention requires each of the first $k \in \{1, 2, 4, 8, 16, 32\}$ moves to reduce exact distance by one, with k at most the bin's lower edge. Solving counts reaching the goal configuration, including after detours; initial distance specifies difficulty, not required path length. Section D.2 gives stopping rules and longer rollouts. Probe and behavior populations differ; follow-ups also probe model rollouts.

Uncertainty. Three-seed ranges describe training variation. Bootstrap intervals resample structural groups while holding the trained models fixed (Section E.1); they do not quantify variation over new training seeds. Historical single-seed diagnostics, unadjusted tests, and descriptive control ranges are identified with their results. None is interpreted as an equivalence test.

5 WHAT FORMS UNDER ACTION PREDICTION ALONE

We begin with the policy trained only to predict optimal actions. It solves many held-out puzzles, but how much of the environment can we recover from its representations? Table 1 compares three probe families with behavior across all eight component-supervision conditions and three seeds.

Table 1: Eight component-supervision conditions: three-seed means (%). Recovery is each seed's best test-layer accuracy over 17 layers (descriptive). Behavior measures solving and first-32 retention on 32–60-step puzzles. Probe and rollout populations differ; seed ranges and the latent-objective comparison are in Table 6.

Training condition	Complete-state recovery			Behavioral competence	
	Linear	Nonlinear	Attention pooling	Solve rate	First-32 retention
Action only	20.8	42.7	38.2	74.5	64.5
State supervision	95.8	95.9	95.8	71.1	59.9
Action-effect supervision	26.4	57.5	44.1	76.4	67.5
Goal-occupancy supervision	25.2	46.9	41.4	70.6	59.5
State + action-effect supervision	95.9	96.0	95.9	72.7	63.0
State + goal-occupancy supervision	95.3	95.5	95.3	73.7	62.4
Action-effect + goal-occupancy supervision	26.8	54.9	41.7	76.0	67.0
All-component supervision	95.5	95.7	95.5	73.5	63.0

Incomplete recovery accompanies successful behavior. Action-only training solves 0.745 of held-out 32–60-step puzzles despite incomplete state recovery under every probe. All-component supervision raises recovery above 0.95 for all three probes, yet its solve rate is 0.735. These measurements use different populations; their difference is not a numerical knowledge deficit.

Part of the low recovery is a measurement limitation. Nonlinear decoding recovers information missed by the linear probe, and the MLP exceeds attention pooling on average (Table 6). Pooling controls reach at most 0.078 on a random network and 0.026 on raw embeddings (Table 5). Thus stronger readouts help, but the results do not establish that reading multiple positions is necessary. These test-layer maxima are distinct from the validation-selected primary estimates (Table 15).

Recovery differs across components. Player-cell accuracy is 0.949, whereas both-box accuracy is 0.213, with box-cell AUROC 0.894 (Figure 9). Overall action-effect accuracy is 0.981; the weakest physical context scores 0.843. Joint goal-occupancy accuracy is 0.622 (Figure 5 and Table 4). The complete-state criterion requires the player and both boxes to be correct at once. Failing it therefore leaves open whether enough information remains for the decisions the policy actually makes.

The policy distinguishes states beyond the matched cues. We test that possibility with state pairs that share local information but require different optimal actions. The matched-neighborhood test excludes a deterministic policy based only on the current neighborhood (Figure 10). A stronger test also fixes the initial board, move count, immediate action effects, last two actions, and push count. Across 20,000 pairs in 127 structural groups, action-only training chooses an optimal action for both members in 89.2% of pairs; all-component supervision reaches 93.1%. The baseline therefore distinguishes remote states beyond these cues. This does not identify its internal encoding or exclude other history heuristics, and some errors remain cue-aligned (Section E.1).

A complementary intervention on the measured code. Repeated correction makes the linear box readout exact without establishing better solving. This concerns the tested code and edit, not all state

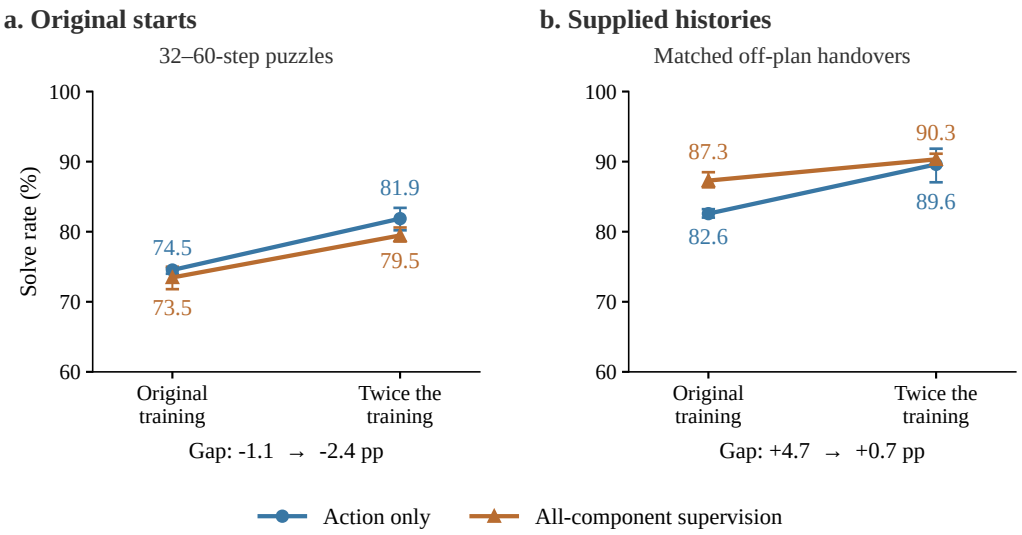

Figure 4: When supervision helps solving. (a) Original starts on 500 held-out puzzles with optimal distances of 32–60. (b) Continued solving on 417 matched off-plan handovers in 141 structural groups. Both conditions are trained with the original or twice-longer schedule. Points are three-seed means; whiskers span seeds, not confidence intervals. Gap denotes all-component supervision minus action only, in percentage points. The panels use different evaluation populations. The history-solving gap shrinks with more training, without establishing equivalence or a causal mediator (Section H.4).

information; rollout associations and intervention controls are in Section F and Figure 11. We next examine supervision during training.

6 How supervision changes knowledge and behavior

Supervision changes what can be recovered. The full 2^3 comparison has three seeds per condition (Table 16). With state supervision, complete-state accuracy ranges from 0.953 to 0.959 across the four component combinations. Without it, even joint action-effect and goal-occupancy supervision reaches only 0.268. Supervision also transfers across components: action-effect and state targets raise joint goal-occupancy accuracy from 0.622 to 0.697 and 0.774, respectively.

Better recovery does not consistently improve original-start solving. All-component supervision raises complete-state recovery from 0.208 to 0.955, while mean 32–60-step solving changes from 0.745 to 0.735 (Figure 5 and Table 15). Averaged over the other objectives, adding state supervision lowers 16–31-step solving in all three seeds; its far-bin effect varies by seed (Section G).

Allowing longer rollouts leaves this contrast largely unchanged. Longer rollouts change far-bin solving by at most 0.2 percentage points per condition, and 95.3% of original failures are already irreversible. Section D.2 distinguishes these failures from unresolved cutoff cases.

Supervision helps after supplied detour histories. Original-start rollouts let a policy choose which states it visits. To test continuation from other histories, we supply detours that move boxes off every shortest path from the start. Here all-component supervision improves solving from 81.4% to 87.6% across three seeds; the development panel gives 83.3% versus 90.0% (Section H). Providing the true current board reduces the gap (Table 21). However, comparison with matched player-only detours does not isolate a benefit specific to off-plan boxes.

Training amount also matters. On enlarged matched handovers, doubling training in both conditions shrinks the supervision advantage from 4.7 to 0.7 points (Figure 4 and Section H.4). Both conditions improve with more training, but their relative advantage depends on the evaluation setting. This leaves training efficiency as a plausible contributor without establishing equivalence or identifying

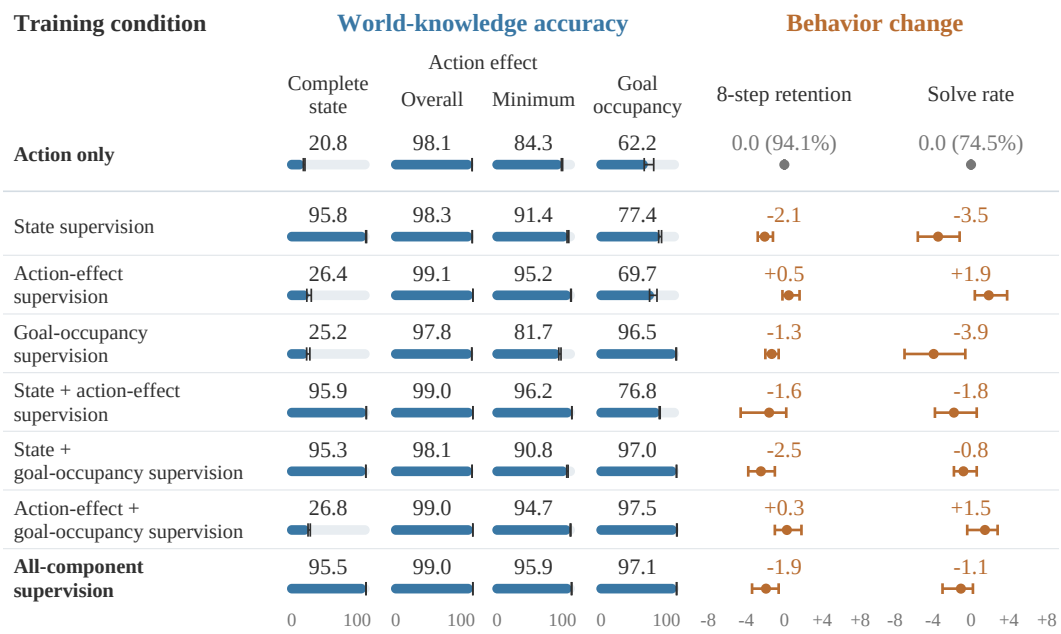

Figure 5: Eight component-supervision conditions, three seeds each. Left: complete-state, action-effect, and joint goal-occupancy accuracies (0–100%). Action effect shows overall accuracy and the minimum across five physical contexts, computed within each seed; per-context results are in Table 4. Right: same-seed changes from action only in 8-step retention (16–31-step puzzles) and solving (32–60-step puzzles), in percentage points; parentheses give absolute baseline performance. Marks show means and min–max seed ranges, not confidence intervals.

the mechanism. The benefit of supervision is therefore real in the tested history setting, but cannot be inferred from recovery gains alone.

Recovered rules can support an external solver. A seed-1 follow-up combines decoded transition and goal rules with exact search. Conditions meeting the specified recovery criterion solve every confirmatory episode with this machinery (Table 20). The experimenter supplies the search and symbolic update operations, so this tests the extracted rules' use in that system, not planning inside the policy. The appendix gives the criterion and its post-result application.

6.1 Relative timing over training

Finally, we ask whether knowledge and behavior improve at the same stage of training. We use t_{50}, the first checkpoint at half a metric's own rise from initialization to its final value (Table 19, seed 1). All-component supervision reaches the readout halfway points earlier, but shares the action-only crossing for 8–15-step solving. On 32–60-step puzzles, solving reaches t_{50} at 22,298 updates for action only and 14,692 for all-component supervision. These crossings describe relative progress on a coarse checkpoint grid; they do not locate when information first exists or define a universal order in which knowledge and competence form.

7 Related work

World models for control. World Models and PlaNet learn predictive representations for control (Ha & Schmidhuber, 2018; Hafner et al., 2019), while MuZero trains latent dynamics around reward, value, and policy predictions used by search (Schrittwieser et al., 2020). Predictive-state representations likewise describe state through future observations rather than privileged coordinates (Littman et al., 2001). Our question concerns the knowledge inside an action-prediction policy: does supervis-

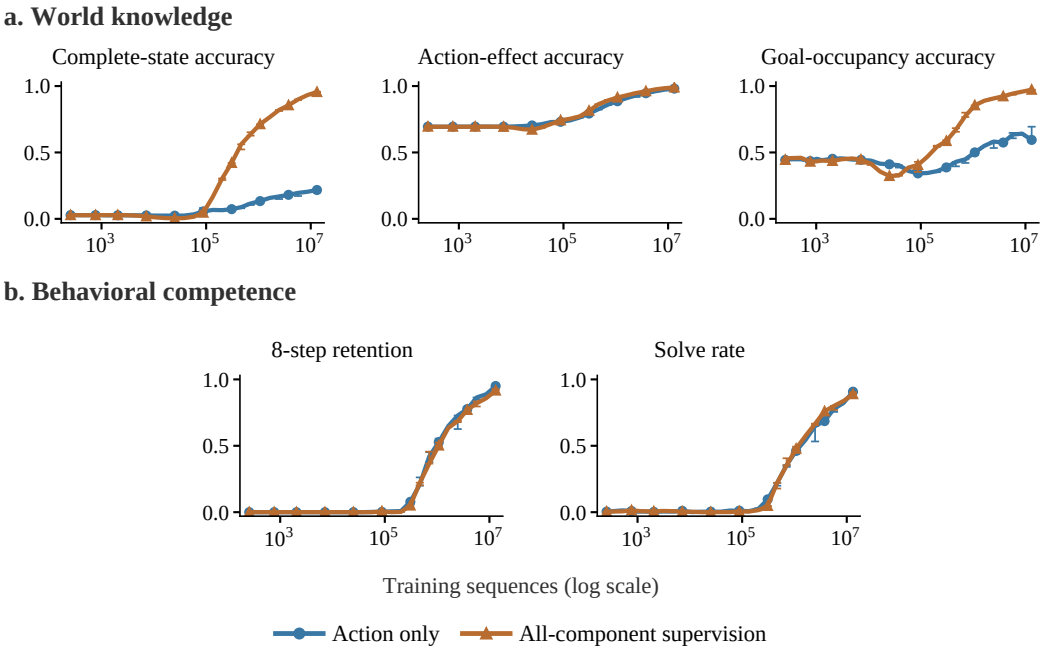

Figure 6: Knowledge and behavior over training. (a) Complete-state, overall action-effect, and joint goal-occupancy accuracy. (b) 8-step retention on 16–31-step puzzles and solve rate across all bins, allowing detours. Colors and markers identify conditions. Curves retain all 28 seed-1 checkpoints, with sparse markers; error bars show min–max seed ranges at shared checkpoints (the other seeds have 12), not confidence intervals. Initialization is plotted at one update. Relative halfway times appear in Table 19.

ing an exact description of its environment improve solving? Poor symbolic recovery alone cannot answer whether it has a useful world model.

World representations in sequence models. Othello and chess studies establish board-state tracking and, in Othello, behavioral effects of representation edits (Li et al., 2023; Nanda et al., 2023b; Toshniwal et al., 2022; Karvonen, 2024). Training tasks and predictive statistics also shape these representations (Li et al., 2025; Brenner et al., 2026). We extend this line by varying supervision of three environment components on the same corpus and testing both original-start solving and continuation after supplied histories. This comparison separates improved state recovery from improvements that depend on the behavioral setting.

What counts as evidence of useful knowledge? Vafa et al. (2024) show that predictive success can coexist with incoherent world models. Probing work similarly distinguishes decodability from task relevance (Hewitt & Liang, 2019; Belinkov, 2022; Ravichander et al., 2021). Our results add a controlled supervision comparison to these evaluation concerns. State-sensitive decisions test specific behavioral alternatives; activation edits test a particular code, with controls for their scope (Geiger et al., 2021; Zhang & Nanda, 2024; Spies et al., 2025). Sokoban planning studies identify useful internal computation and future-plan representations (Guez et al., 2019; Taufeeque et al., 2025); our null box-code repairs do not rule out such mechanisms. Section J develops these connections.

8 Limitations

Scope of the evidence. We study one 50.5M-parameter architecture and supervision site in 7×7 Sokoban with two boxes. The eight main conditions and the training-amount and alternative-target controls have three seeds; some historical diagnostics have one. Longer rollouts exceed training-history lengths and leave a few solvable cutoff states unresolved, so they do not measure unrestricted

eventual success (Section D.2). The goal tests measure occupancy and event response, not general goal understanding. Action and component labels come from an exact engine; this is not an unsupervised training recipe.

What the measurements establish. Probe accuracy depends on the readout family, and attention pooling can perform computation itself. Rollout associations can reflect difficulty or trajectory confounds. The cue-matched tests exclude only the named deterministic policies; other history heuristics remain possible. Activation edits target one linear code, not every state representation. Exact probe output after repair therefore need not mean a corrected internal state estimate.

What remains unresolved. Matching updates does not match auxiliary computation or gradients. Longer-training controls reduce the supervision gap without establishing equivalence or its mediator; alternative targets also differ in difficulty and gradient geometry. Supplying a current board resets history and positions, so it is not a pure intervention on state knowledge. External search uses experimenter-supplied machinery. Several diagnostic criteria and follow-ups were specified after the main results; their provenance is documented in Section I. Group intervals condition on trained seeds, and event sign tests are not cluster-adjusted. These limits prevent a general conclusion about an interpretability–performance tradeoff.

9 Conclusion

In these Sokoban policies, world knowledge and behavioral competence develop together without moving in lockstep. Component supervision makes the full state much easier to recover, but its solving benefit depends on the histories the policy must handle and the training budget. Action-only policies still use information beyond the matched local and history cues. The practical lesson is to evaluate knowledge supervision against the behavior it is intended to improve, using readouts, state-sensitive tests, and controlled task continuations together.

Reproducibility statement

Sections 2 to 4 and the appendix give methods and protocol revisions. The repository includes code, summaries, and raw budget/history records with hashes; other GPU analyses require external data and weights.

Ethics statement

No human subjects or personal data are used; these findings do not assure deployment safety.

AI use statement

In this work, the authors used generative AI tools to aid in polishing the writing and improving the readability of the manuscript, and to assist with writing and debugging the experimental code. The authors carefully reviewed, verified, and edited all AI-generated content and code, and take full responsibility for the content and correctness of this final work.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Sasha Brenner, Thomas R. Knösche, and Nico Scherf. Predictive statistics shape emergent world representations of grid walkers. *arXiv preprint arXiv:2603.16689*, 2026. URL `https://arxiv.org/abs/2603.16689`. Version 2.

Sky CH-Wang, Benjamin Van Durme, Jason Eisner, and Chris Kedzie. Do androids know they're only dreaming of electric sheep? In *Findings of the Association for Computational Linguistics: ACL*

2024, pp. 4401–4420. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-acl.260. URL https://aclanthology.org/2024.findings-acl.260/.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, 2021.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. DeepMDP: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning*, pp. 2170–2179, 2019.

Arthur Guez, Mehdi Mirza, Karol Gregor, Rishabh Kabra, Sébastien Racanière, Théophane Weber, David Raposo, Adam Santoro, Laurent Orseau, Tom Eccles, Greg Wayne, David Silver, and Timothy Lillicrap. An investigation of model-free planning. In *International Conference on Machine Learning*, 2019.

Wes Gurnee and Max Tegmark. Language models represent space and time. In *International Conference on Learning Representations*, 2024.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pp. 2555–2565, 2019.

Jessica B. Hamrick, Abram L. Friesen, Feryal Behbahani, Arthur Guez, Fabio Viola, Sims Witherspoon, Thomas Anthony, Lars Buesing, Petar Veličković, and Théophane Weber. On the role of planning in model-based deep reinforcement learning. In *International Conference on Learning Representations*, 2021.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Conference on Empirical Methods in Natural Language Processing*, 2019.

Michael Ivanitskiy, Alexander F. Spies, Tilman Räuker, Guillaume Corlouer, Christopher Mathwin, Lucia Quirke, Can Rager, Rusheb Shah, Dan Valentine, Cecilia Diniz Behn, Katsumi Inoue, and Samy Wu Fung. Linearly structured world representations in maze-solving transformers. In *Proceedings of UniReps: the First Workshop on Unifying Representations in Neural Models*, volume 243 of *Proceedings of Machine Learning Research*, pp. 133–143, 2024. URL https://proceedings.mlr.press/v243/ivanitskiy24a.html.

Erik Jenner, Shreyas Kapur, Vasil Georgiev, Cameron Allen, Scott Emmons, and Stuart Russell. Evidence of learned look-ahead in a chess-playing neural network. In *Advances in Neural Information Processing Systems*, 2024.

Adam Karvonen. Emergent world models and latent variable estimation in chess-playing language models. In *Conference on Language Modeling*, 2024.

Thomas Kipf, Elise van der Pol, and Max Welling. Contrastive learning of structured world models. In *International Conference on Learning Representations*, 2020.

Belinda Z. Li, Zifan Carl Guo, and Jacob Andreas. (How) do language models track state? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 34429–34452, 2025. URL https://proceedings.mlr.press/v267/li25r.html.

Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. In *International Conference on Learning Representations*, 2023.

Michael L. Littman, Richard S. Sutton, and Satinder Singh. Predictive representations of state. In *Advances in Neural Information Processing Systems*, volume 14, pp. 1555–1561, 2001.

Ziming Liu, Ouail Kitouni, Niklas Nolte, Eric J. Michaud, Max Tegmark, and Mike Williams. Towards understanding grokking: An effective theory of representation learning. In *Advances in Neural Information Processing Systems*, 2022.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *International Conference on Learning Representations*, 2023a.

Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. In *Proceedings of the 6th BlackboxNLP Workshop*, 2023b.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Sébastien Racanière, Théophane Weber, David P. Reichert, Lars Buesing, Arthur Guez, Danilo Rezende, Adrià Puigdomènech Badia, Oriol Vinyals, Nicolas Heess, Yujia Li, Razvan Pascanu, Peter Battaglia, Demis Hassabis, David Silver, and Daan Wierstra. Imagination-augmented agents for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. Probing the probing paradigm: Does probing accuracy entail task relevance? In *Conference of the European Chapter of the Association for Computational Linguistics*, 2021.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588 (7839):604–609, 2020.

Max Schwarzer, Ankesh Anand, Rishab Goel, R. Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *International Conference on Learning Representations*, 2021.

Alex F. Spies, William Edwards, Michael I. Ivanitskiy, Adrians Skapars, Tilman Räuker, Katsumi Inoue, Alessandra Russo, and Murray Shanahan. Transformers use causal world models in maze-solving tasks. *arXiv preprint arXiv:2412.11867*, 2025. URL `https://arxiv.org/abs/2412.11867`. Version 2; also presented at the ICLR 2025 World Models Workshop.

Mohammad Taufeeque, Philip Quirke, Maximilian Li, Chris Cundy, Aaron David Tucker, Adam Gleave, and Adrià Garriga-Alonso. Planning in a recurrent neural network that plays sokoban. *arXiv preprint arXiv:2407.15421*, 2025. URL `https://arxiv.org/abs/2407.15421`. Version 3; earlier version at the ICML 2024 Mechanistic Interpretability Workshop.

Curt Tigges, Michael Hanna, Qinan Yu, and Stella Biderman. LLM circuit analyses are consistent across training and scale. In *Advances in Neural Information Processing Systems*, 2024.

Shubham Toshniwal, Sam Wiseman, Karen Livescu, and Kevin Gimpel. Chess as a testbed for language model state tracking. In *AAAI Conference on Artificial Intelligence*, 2022.

Keyon Vafa, Justin Y. Chen, Ashesh Rambachan, Jon Kleinberg, and Sendhil Mullainathan. Evaluating the world model implicit in a generative model. In *Advances in Neural Information Processing Systems*, 2024.

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *International Conference on Learning Representations*, 2024.

APPENDIX GUIDE

The appendix follows the main analysis from data construction to measurement, behavioral tests, and controls. The guide below locates the details needed to interpret each result; seed coverage and analysis qualifications accompany the relevant experiment.

Location	Contents
Section A	Corpus construction, splits, and engine audits
Section B	Model, training configuration, and latent prediction + state matching
Section C	Probe fitting, controls, and reconstruction results
Section D	Solving, path retention, and longer-rollout checks
Section E	Matched-state decisions and history-cue diagnostics
Section F	Rollout association, direction removal, and state-code repair
Section G	Training-condition comparisons, controls, and development
Section H	Supplied-history continuation, training amount, and alternative targets
Section I	Protocol changes, corrections, and software provenance
Section J	Extended related work

A CORPUS CONSTRUCTION AND EVALUATION-SET AUDITS

Physics coverage. The corpus contains the following executed transitions: WALK 54,719,535, PUSH 11,228,936, wall ahead 11,484,977, box against wall 1,090,435, box against box 220,604. The preregistered minimum per outcome and direction (25,000 transitions in 1,000 puzzles and 250 groups) is met by every cell; the rarest, box against box, has at least 54,816 transitions in 37,872 puzzles and 9,384 groups per direction. All invariant checks pass: every trajectory solves, every position is solvable, every target is optimal, and replay agrees with both engines.

Same state, different history. State matching needs examples in which different histories reach the same state. A position of B is a *natural rejoin* when its state lies on A's path and B's history before it contains a state-changing detour. 95.9% of pairs contain one, 27,504,115 positions in total (preregistered minimum: 20% of pairs and one million positions), with a median history-length difference of 19 actions and a median edit distance of 19. Another 2,335,691 positions differ from A only by blocked attempts; these are excluded from the positive state-matching pairs.

Composition. Detour trajectories account for 79.1% (62,254,924 of 78,744,487 positions). Their mean length is 48.64, compared with 12.88 for optimal trajectories. Median excess length is 39 actions and median inefficiency L/L^* is 3.76; 39.1% of detours use the whole budget. A detour trajectory contains on average 10.00 blocked attempts, 0.18 sideways and 12.31 backward moves. Starts fall 996,353 / 277,308 / 6,339 in the distance bins 8–15 / 16–31 / 32–60.

Generation. Each 7×7 board has a walled border and a connected floor region of 16 to 25 of the 25 interior cells, grown by random frontier growth. The two goals occupy distinct floor cells. Layouts are drawn on the GPU in blocks of 65,536 (327,680 draws in total); a draw is kept if its group belongs to the training partition and not to the development panel, and if its (walls, goals) pair is new up to the eight symmetries. Kept layouts are generated in chunks of equal floor count, and the corpus is cut at exactly 160,000 layouts in draw order, so the floor mix is not biased by the chunking. For each layout, 12 candidate starts are drawn uniformly without replacement from its solvable, unsolved states at distance 8–60, and the first 8 whose detour is accepted within four attempts are kept; a layout with fewer than 8 is replaced by a fresh one.

Detour rule. For a start s_0, trajectory A follows a fixed tie-broken optimal action table and solves in exactly $d(s_0)$ actions. Trajectory B starts from the same s_0. After u actions at state s, a non-optimal candidate with successor s' must satisfy

$$d(s') < \infty, \qquad d(s') \neq d(s) - 1, \qquad u + 1 + d(s') \leq 64.$$

Blocked attempts ($s' = s$) qualify. With probability $q = 0.5$, a candidate is chosen uniformly when available; otherwise B takes the optimal action. The remaining budget forces an optimal suffix. B is accepted only if it solves, is longer than A, and contains a state-changing non-optimal action. Up to

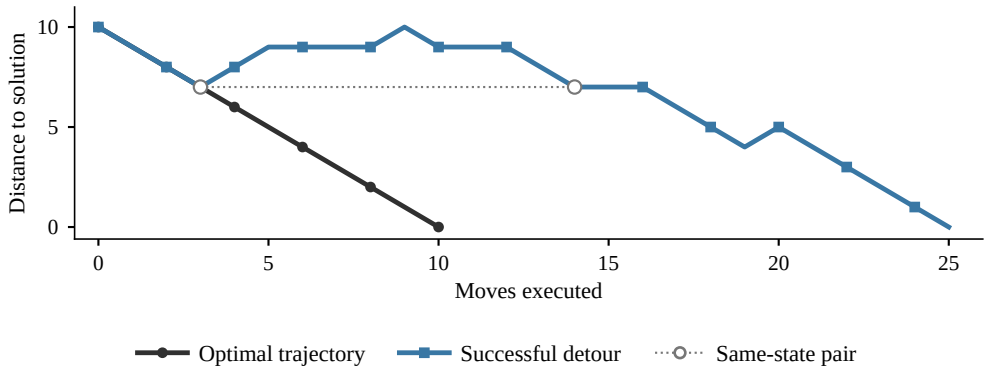

Figure 7: One recorded start, two successful trajectories. After three moves, the detour executes right while its training target is up; the executed move stays in the history. Open circles connected by a dotted line mark the same state reached after 3 and 14 moves, providing a positive pair for state matching. Both trajectories remain solvable and end at a solution.

four attempts are made per start. Both trajectories train the policy; at each position, the target is the current state's solver-optimal action, including after an executed detour.

Invariants and replay. The audit recomputes every count from the stored arrays. It finds zero unsolved trajectories, unsolvable positions, positions without a target, non-optimal targets, optimal trajectories that are not shortest or contain a detour, detours not longer than their optimal twin, detours without a state-changing non-optimal action, start mismatches within a pair, trajectories over the budget and positions whose distance exceeds the remaining budget. Every trajectory of the corpus and of the probe set was then replayed from its start through the GPU successor table (states, distances, branches, goal bits, targets and optimal masks all identical) and 4,000 random transitions of each were stepped with the scalar engine, which shares no code with the GPU tables. Both replay checks find zero mismatches.

Same-state pairs. Natural rejoins: 27,504,115 positions, 95.9% of pairs. Blocked-only same-state positions: 2,335,691. Within-detour revisits of a state after a state change: 28,023,583; natural rejoins are 49.5% of all same-state / different-history position pairs, the rest being revisits within one trajectory.

Probe set. The probe-training, validation, and test splits contain 460,938 / 278,871 / 251,828 action positions in 327 / 130 / 180 groups; the probe partition is disjoint from the training partition by construction.

Confirmatory panel. The panel contains 1,500 episodes over 1,381 layouts, drawn from 7,798,784 fresh layout draws restricted to confirmatory-partition groups that no panel of any earlier evaluation used; 500 distinct groups in each distance bin (no bin needed the preregistered fallback of two episodes per group). The panel file, the partition file and the builder are hashed in its freeze record.

B Model and Training Objectives

B.1 Architecture and Input Alignment

Input and alignment. The pre-LayerNorm decoder-only Transformer has 16 blocks, $d = 512$, 8 attention heads, $d_{\text{ff}} = 2048$, learned positions, and a 17-token vocabulary. The input is

$$\texttt{BOS BOARD}\ t_0 \dots t_{48}\ \texttt{HISTORY}\ a_0\ a_1 \dots a_{L-1}.$$

The 49 tiles describe only the initial board. Token 51 (`HISTORY`) is aligned with s_0; the token of a_{k-1}, at index $51 + k$, is aligned with s_k after k moves and predicts action k. Move counts therefore exclude the board and special tokens, and $k = 0$ denotes the initial state. Causal attention sees only the initial board and actions $a_0, \dots, a_{k-1}$. State readouts target the state at that position.

B.2 Training configuration and component supervision

Training losses. All conditions minimize action cross-entropy. Auxiliary linear heads attach after block 8 at each action-target position. State loss is player cross-entropy plus 25 summed box binary cross-entropies; the action-effect loss sums four three-way losses; the goal-occupancy loss sums two binary cross-entropies. Each loss is averaged over positions and has coefficient 1. Matching the update count therefore does not match loss magnitudes, gradients, or computation. Latent prediction and state matching use contrastive InfoNCE losses and an exponential-moving-average target, as specified in Section B.3.

Table 2: Model and training configuration, shared by the original conditions except the auxiliary objective; extended schedules are in Section H.4.

layers / d_{model} / d_{ff} / heads	16 / 512 / 2048 / 8
parameters	50,548,224 (the auxiliary heads add at most 0.4M, discarded at measurement)
corpus	2,560,000 trajectories, 78,744,487 supervised positions
batch	128 starts with both trajectories (256 trajectories) per update
horizon	51,360 updates (about 5.1 passes)
optimizer	AdamW, $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.1 on linear weights
learning rate	3×10^{-4}, linear warm-up over 3%, cosine to $0.1\times$
gradient clipping	global norm 1.0; an update with a non-finite norm is skipped
precision	bf16 autocast, fp32 master weights
checkpoints	28 log-spaced update counts from 0 to 51,360
seed	1, 2, and 3 for all nine conditions (initialization and batch order); historical appendix controls use seed 1; objective extensions use all three

Efficiency. Each update is split into two length-sorted microbatches, padded to the next multiple of 16 actions; the compiled forward pass uses causal flash attention, which is exact for right-padded rows. The corpus is tokenized once and kept on the GPU. Batch order is a pure function of the update index, so resumed runs replay the same batches. Telemetry is buffered on the device, and a background thread writes recovery snapshots from pinned memory. One condition takes about 2–3.5 hours on one RTX A6000. Measurement reads all 17 readout locations at every position of every probe-set trajectory in one forward per trajectory.

B.3 Latent prediction and state matching

Additional comparison. Alongside the eight component-supervision conditions in the main text, we train *Latent prediction + state matching* with the same paired corpus, architecture, update count, and three seeds. It adds action-conditioned future-representation prediction and aligns representations reached through different histories of the same state. State matching uses engine-supplied equality labels; neither term supplies coordinate targets. This ninth condition has 28 seed-1 and 12 replicate-seed checkpoints. Its final readout and behavioral results appear in Tables 6 and 15; its relative response times appear in Table 19.

Recovery and behavioral scope. The three-seed mean complete-state accuracy is 0.317, compared with 0.208 for action-only training. It remains below explicit state supervision alone (0.958) or with all components (0.955). The MLP exceeds attention pooling on average for this condition (Table 6). Its state-direction ablation reduces solving below same-rank random controls in the seed-1 diagnostic (Table 12), unlike the action-only and all-component conditions. These objective-specific results do not establish that useful latent world models require symbolic-coordinate targets.

Definition. At each time-aligned position, we project the block-8 representation to $u = Ph^{(8)} + b_P$, with $P \in \mathbb{R}^{256\times512}$ and $b_P \in \mathbb{R}^{256}$, then normalize it as $z = u/\|u\|$. This applies to both trajectories of each start. A target branch, an exponential moving average (momentum 0.996) of the transformer and P, gives z^* without gradient. *Multi-step prediction:* from position t, a predictor applies $p \leftarrow p + \mathrm{MLP}([p, \mathrm{onehot}(a)])$ along the executed actions $a_t, \ldots, a_{t+k-1}$, starting at u_t (MLP $260 \to 512 \to 256$, GELU); for $k \in \{1, 2, 4, 8\}$, $p/\|p\|$ is scored by InfoNCE (temperature

0.1) against z^*_{t+k}, with the z^* of every position of the same start whose state differs from s_{t+k} as negatives (positions of the same state are neither positives nor negatives). Only terms with $t + k \leq L$ contribute. Future actions condition the prediction; they are never targets. *State matching:* at every natural rejoin (A_j, B_k), symmetric InfoNCE treats the partner as positive and different-state positions from the same start as negatives, using the online z. $\mathcal{L} = \mathcal{L}_{\text{action}} + \mathcal{L}_{\text{pred}} + \mathcal{L}_{\text{cons}}$; each loss is averaged over its contributing terms in the update.

Anti-collapse and diagnostics. A constant representation cannot satisfy either term, because both contrast the true partner with different-state positions, and the prediction target is a slowly moving network. To check learning and collapse, we log predictive top-1 accuracy at each k, same-state and different-state cosine similarity, rejoin retrieval accuracy, per-dimension standard deviation of z, and the effective rank of its covariance. A failure of latent prediction and state matching to yield recoverable symbolic coordinates is interpreted only after these diagnostics verify optimization of its own objective. At the final checkpoints, three-seed means on test pairs of the probe set are: rejoin cosine 0.857, different-state cosine -0.019, rejoin retrieval 0.959, effective rank 33.667 of the 256-dimensional code, and 8-step predictive top-1 accuracy 0.997 (scored against the online branch, since grid checkpoints do not keep the EMA target). Retrieval ranges from 0.955 to 0.962, and effective rank from 32.983 to 34.036 across seeds. All three seeds learn the auxiliary task without collapsing the representation.

Objective ablations. We train prediction-only and matching-only variants, and prediction horizons $\{1\}$ and $\{1, 2, 4\}$, with the same seed and number of updates (Table 18). Most of the complete-state gain comes from the state matching term: state matching alone reaches 0.335 (rejoin retrieval 0.96), prediction alone 0.209, no better than action-only training (0.217), with a retrieval of only 0.02, so predicting the future latent does not by itself align the histories that reach one state; both together give 0.388. Shorter horizons give 0.304 ($\{1\}$) and 0.341 ($\{1, 2, 4\}$). No variant reaches the state readout accuracy of explicit state supervision. Solve-rate differences are descriptive and single-seed (32–60-step bin 0.710 to 0.782, against 0.748 for action-only training).

C Probe methods and reconstruction results

C.1 Readout definitions and fitting

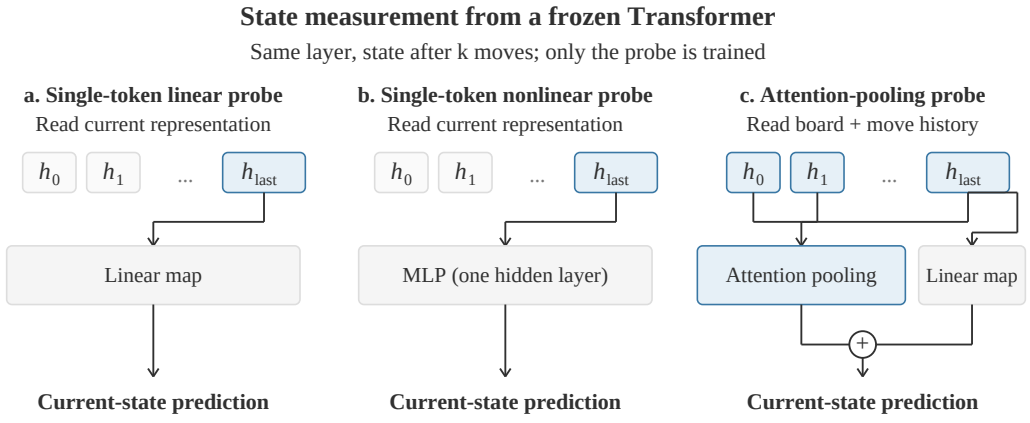

Figure 8: Three state probes after k moves: (a) linear, (b) nonlinear, and (c) attention pooling. Blue boxes mark their inputs. The current-token representation h_{last} at position $51 + k$ already contains preceding context. Attention pooling also reads the initial board and move history, including special tokens, masks future moves, and adds a current-token linear prediction.

Data and selection. The separate probe set contains 2,000 layouts and 32,000 trajectories. Probe-training, validation, and test splits contain 327, 130, and 180 structural groups; trajectories stay within a split. Training is capped at 400,000 positions. Each time-aligned readout targets s_k for

$k = 0, \ldots, L$. Primary linear-probe layers are selected on validation and scored on test; nonlinear and attention-pooling maxima across test layers are descriptive. The Transformer remains frozen while each probe is fitted.

Probe fitting. A position-shared linear probe at each layer predicts 25 player indicators, 25 box indicators, and two goal-occupancy indicators. Decoding selects the highest player score and two highest box scores, excluding walls. Features are standardized per layer on the probe-training split; the state probe minimizes the masked binary cross-entropy of its 52 outputs and the action-effect probe the cross-entropy of its four three-way outputs, each plus $10^{-4}\|W\|^2$, by full-batch L-BFGS (200 iterations, history 20, strong Wolfe line search). Goal-occupancy thresholds maximize balanced accuracy on the probe-validation split. We select the shared state/goal-occupancy layer by complete-state accuracy; the action-effect layer is selected by action-effect accuracy. Both selections use probe-validation, and results are reported on the probe-test split.

Attention-pooling probe. For target c at time k, the output is the fitted local probe at h_{51+k} plus attention pooling of $(Uh_j)_c$ over $j \leq 51 + k$. Keys are Kh_j, the query is $Qh_{51+k} + q_c$, and the key dimension is 64. U starts at zero. Adam uses learning rates 10^{-3} for U, Q, K, q and 10^{-4} for the local probe, eight epochs, and trajectory batches of 128. Fitting uses 4,000 probe-training trajectories; the lowest-loss epoch on 2,000 probe-validation trajectories is retained; all test positions are scored. Trained, random-network, and raw-input conditions use the same protocol. Layer maxima are descriptive test-set maxima, not validation-selected primary estimates.

C.2 Component accuracy and controls

Three-seed component summary. The player's cell is recovered with accuracy 0.949 and AUROC 0.996. Box-cell AUROC is 0.894, while both-box exact accuracy is 0.213 (Figure 9). Action effects are recovered with accuracy 0.981, with box against box the weakest context (0.843, Table 4); combining decoded state and effect recovers the exact next state in 0.203 of cases. Joint goal-occupancy accuracy is 0.622. At goal entry, the relevant score increases more than on matched non-goal pushes in 0.874 of pairs (each seed passes the nominal one-sided sign test, not cluster-adjusted). These measurements support component and event-aligned information, not general goal reasoning. Task-relevant or compressed state representations are compatible with these results, but their form and sufficiency for the policy are not identified.

Additional measurements and controls. State measurements include per-cell AUROC, component accuracy, and layer/history breakdowns. A secondary test combines decoded positions and action effects to construct the next state; distinct targets do not imply independent knowledge components. For goal occupancy, a recorded one-sided sign test compares score changes at entry events with matched non-goal pushes; its nominal p is not adjusted for dependent events within structural groups. Identical attention-pooling probes on trained representations, a random network, and raw embeddings control for reconstruction by the probe itself. An advantage over these controls does not distinguish nonlinear single-token information from information distributed across tokens. The causal prefix mask excludes future tokens; it is not an intervention.

The action-only summary and fixed-layer controls below use seed 1. The physical-context and all-condition probe comparisons (Tables 4 and 6) use three seeds.

Table 3: Seed-1 action-only training at the end of training. State: complete-state accuracy at the validation-chosen site; Action effect: accuracy over all (position, action) cases and in the worst physical context; Goal occupancy: joint goal-occupancy accuracy and whether the representation responds at the entry position (matched sign test). Behavior on the confirmatory panel.

State	complete-state accuracy (site 11)	0.217
Action effect	overall / worst-context accuracy	0.981 / 0.844
Goal occupancy	joint accuracy	0.594
Retention	8-step retention in the 8–15 / 16–31 / 32–60 bins	1.000 / 0.950 / 0.814
Solving	solve rate 8–15 / 16–31 / 32–60	1.000 / 0.974 / 0.748

Table 4: Action-effect accuracy (%) in each of the five physical contexts: three-seed mean [min, max], at the final checkpoint and each seed's validation-selected action-effect layer. The minimum in Figure 5 is taken across contexts within each seed before averaging. Next state is the separate usability test: decoded state plus decoded effect of the executed action gives the exact next state.

	Walk	Push	Wall ahead	Box against wall	Box against box	Next state
Action only	99.0 [99.0, 99.1]	94.6 [94.5, 94.6]	98.2 [98.1, 98.4]	93.2 [93.1, 93.3]	84.3 [83.5, 85.0]	20.3 [19.3, 21.4]
Latent prediction + state matching	99.3 [99.2, 99.3]	96.5 [96.4, 96.5]	98.7 [98.6, 98.7]	95.9 [95.5, 96.1]	91.3 [91.1, 91.4]	27.8 [23.8, 33.1]
State supervision	99.1 [99.0, 99.1]	95.7 [95.5, 95.9]	98.2 [98.1, 98.3]	91.4 [90.4, 92.5]	92.7 [91.9, 93.7]	96.0 [95.5, 96.4]
Action-effect supervision	99.5 [99.4, 99.5]	97.5 [97.5, 97.6]	99.0 [98.9, 99.1]	96.6 [96.4, 96.9]	95.2 [94.8, 95.5]	24.2 [22.4, 25.9]
Goal-occupancy supervision	98.8 [98.7, 98.9]	93.7 [93.3, 94.0]	97.8 [97.7, 98.0]	91.6 [91.1, 92.2]	81.7 [80.4, 83.1]	24.0 [22.8, 26.2]
State + action-effect supervision	99.4 [99.4, 99.4]	97.6 [97.6, 97.7]	98.9 [98.9, 98.9]	96.4 [96.2, 96.5]	96.2 [96.1, 96.3]	96.2 [96.0, 96.4]
State + goal-occupancy supervision	99.0 [98.9, 99.1]	95.3 [95.2, 95.6]	97.9 [97.8, 98.1]	91.2 [90.0, 93.0]	91.5 [91.1, 91.7]	95.6 [95.5, 95.7]
Action-effect + goal-occupancy supervision	99.4 [99.4, 99.5]	97.4 [97.4, 97.4]	98.9 [98.8, 99.0]	96.4 [96.4, 96.5]	94.7 [94.1, 95.0]	24.7 [23.7, 25.8]
All-component supervision	99.4 [99.4, 99.4]	97.6 [97.5, 97.7]	98.9 [98.8, 99.0]	96.2 [96.0, 96.5]	95.9 [95.6, 96.2]	95.8 [95.7, 96.0]

Table 5: Seed-1 complete-state accuracy of the single-token probe (Token) and of the attention-pooling probe (Pool) at five layers, for three trained conditions and the two controls (a randomly initialized network, and the raw token and position embeddings, which are that network's embedding output).

	embedding output		layer 4		layer 8		layer 12		layer 16	
	Token	Pool	Token	Pool	Token	Pool	Token	Pool	Token	Pool
Action only	0.001	0.028	0.103	0.243	0.156	0.316	0.214	0.406	0.197	0.223
Latent prediction + state matching	0.001	0.031	0.101	0.212	0.180	0.448	0.378	0.551	0.305	0.324
All-component supervision	0.001	0.030	0.118	0.126	0.949	0.949	0.955	0.954	0.950	0.950
random init	–	0.026	–	0.053	–	0.053	–	0.078	–	0.064
raw input	–	0.026	–	–	–	–	–	–	–	–

C.3 Layer, history, and probe-family comparisons

History stratification. Every metric is reported by the number of moves ($k = 0$, 1–3, 4–7, 8–15, 16–31, 32–64) and per trajectory kind (optimal or detour) in the measurement files; the heatmaps in Figure 9 are the per-bin complete-state and action-effect accuracies over the 17 readout locations. A one-hidden-layer MLP probe (512 units, AdamW, best probe-validation epoch) at every layer recovers additional state information in conditions without explicit state supervision (Table 6): the three-seed mean of per-seed best-layer accuracy rises from 0.208 (linear) to 0.427 (MLP) in action-only training, compared with 0.382 for attention pooling. Latent prediction and state matching gives 0.317, 0.619, and 0.494, respectively. The nonlinear single-token probe therefore recovers additional information, but its accuracy remains well below the state-supervised conditions (all-component MLP mean 0.957), where all three readouts have similar accuracy.

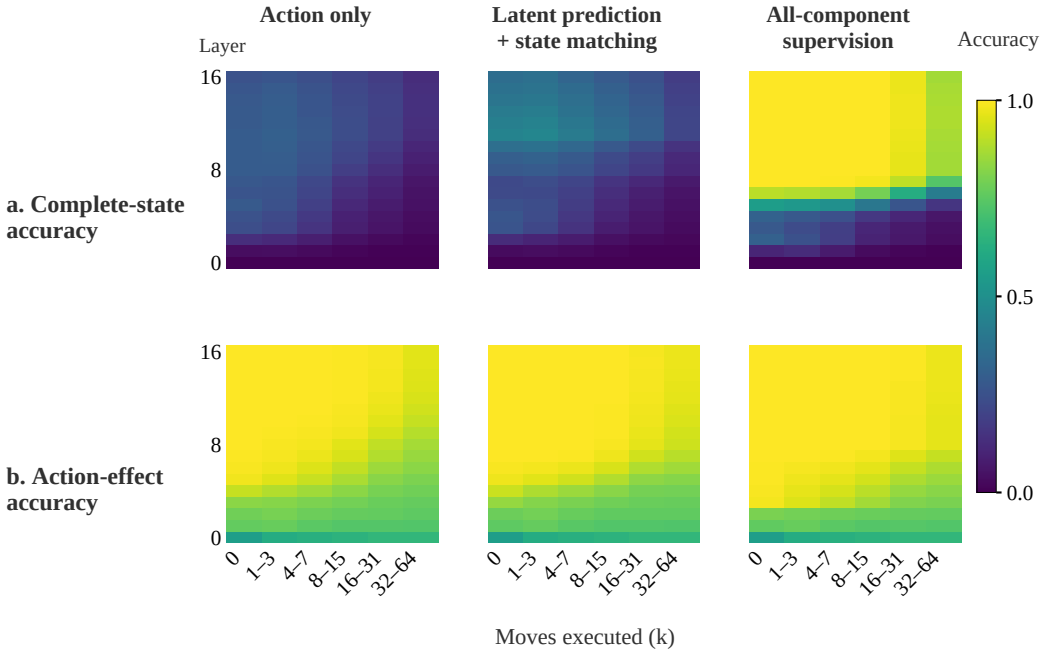

Figure 9: Three-seed mean probe accuracy by layer and number of moves at the final checkpoint, with a common 0–1 scale. (a) Complete-state accuracy. (b) Overall action-effect accuracy. Columns name the training objectives. Layer 0 denotes embeddings; layers 1–16 denote block outputs. These are descriptive readouts, not causal-use tests.

Table 6: All nine conditions: complete-state recovery by probe and behavioral competence, three-seed mean [min, max] (%). Probe scores are per-seed test-layer maxima over 17 locations (descriptive); linear and nonlinear probes read one token, attention pooling reads the prefix. Behavior uses the same policies on 32–60-step puzzles. Probe and rollout populations differ.

	Complete-state recovery			Behavioral competence	
Training condition	Linear	Nonlinear	Attention pooling	Solve rate	First-32 retention
Action only	20.8 [19.8, 21.8]	42.7 [42.3, 43.1]	38.2 [34.7, 43.0]	74.5 [74.0, 74.8]	64.5 [62.2, 67.0]
Latent prediction + state matching	31.7 [27.2, 38.8]	61.9 [56.9, 69.4]	49.4 [43.7, 58.4]	75.9 [75.4, 76.4]	66.0 [64.8, 67.0]
State supervision	95.8 [95.2, 96.2]	95.9 [95.4, 96.3]	95.8 [95.3, 96.2]	71.1 [69.2, 73.6]	59.9 [58.4, 62.8]
Action-effect supervision	26.4 [24.0, 29.4]	57.5 [54.5, 61.4]	44.1 [42.5, 47.1]	76.4 [75.2, 78.6]	67.5 [65.0, 70.8]
Goal-occupancy supervision	25.2 [23.7, 27.7]	46.9 [44.3, 50.3]	41.4 [40.9, 42.2]	70.6 [67.8, 73.4]	59.5 [56.6, 62.8]
State + action-effect supervision	95.9 [95.8, 96.1]	96.0 [95.9, 96.2]	95.9 [95.8, 96.0]	72.7 [71.0, 74.6]	63.0 [61.0, 64.2]
State + goal-occupancy supervision	95.3 [95.3, 95.4]	95.5 [95.5, 95.6]	95.3 [95.3, 95.3]	73.7 [73.0, 74.6]	62.4 [61.8, 63.2]
Action-effect + goal-occupancy supervision	26.8 [25.4, 28.2]	54.9 [51.9, 57.4]	41.7 [30.8, 48.3]	76.0 [74.4, 76.8]	67.0 [65.6, 68.2]
All-component supervision	95.5 [95.4, 95.7]	95.7 [95.5, 95.9]	95.5 [95.3, 95.7]	73.5 [71.8, 75.0]	63.0 [61.8, 65.4]

D PUZZLE SOLVING AND ROLLOUT ROBUSTNESS

D.1 OPTIMAL-PATH RETENTION AND SUCCESSFUL DETOURS

A policy can leave a shortest path and still solve the puzzle. We therefore report both solving and retention: the former counts completed puzzles, including detours, while the latter requires every move in the evaluated prefix to reduce exact distance.

Table 7: Seed-1 optimal-path retention over k actions by distance bin (only k up to the bin's lower edge, so every counted episode has at least k optimal steps to take).

First k actions	Action only	Latent prediction + state matching	State supervision	Action-effect supervision	Goal-occupancy supervision	All-component supervision
Initial optimal distance: 8–15 actions						
1	1.000	0.996	0.994	0.998	0.998	0.998
2	1.000	0.994	0.990	0.998	0.998	0.996
4	1.000	0.994	0.988	0.998	0.998	0.996
8	1.000	0.994	0.986	0.998	0.998	0.996
Initial optimal distance: 16–31 actions						
1	0.984	0.980	0.970	0.970	0.970	0.966
2	0.972	0.970	0.952	0.960	0.966	0.950
4	0.954	0.956	0.934	0.950	0.950	0.926
8	0.950	0.944	0.922	0.950	0.936	0.916
16	0.950	0.938	0.916	0.948	0.930	0.914
Initial optimal distance: 32–60 actions						
1	0.926	0.936	0.920	0.936	0.924	0.926
2	0.890	0.900	0.878	0.904	0.886	0.888
4	0.864	0.862	0.846	0.884	0.852	0.840
8	0.814	0.800	0.778	0.834	0.794	0.784
16	0.726	0.708	0.640	0.756	0.670	0.656
32	0.670	0.648	0.584	0.708	0.592	0.618

Self-generated detours. An exploratory audit of original rollout records separates any departure from a shortest path from actually solving after it. For action-only training in the far-distance bin, 28.6% of episodes that diverge subsequently solve; restricting to first departures that remain solvable gives 37.6%. Successful recoveries contribute 10.2 percentage points of the overall solve rate. These conditional rates pool the three seeds' episode records; the same puzzles recur across seeds. Model-chosen departures depend on puzzle difficulty and policy behavior, so they are descriptive, not a causal detour intervention.

D.2 LONGER ROLLOUTS AND IRREVERSIBLE FAILURES

Question and protocol. Initial optimal distance stratifies puzzle difficulty; a solution can take more actions. We therefore distinguish shortest-path retention from whether the policy solves, allowing detours. To check whether the stopping budget changes the conclusion, this follow-up reevaluates all nine conditions at their final checkpoints, with three seeds and the same 1,500 frozen episodes. The protocol was specified after the original results. Each greedy trajectory continues until solving or 128 actions, with endpoints at 64, 96, and 128. There is no updated-board input, history reset, action masking, position remapping, or weight change. All 27 runs reproduce every original action and exact state distance through action 64, as well as the recorded solve rates.

Solving, irreversible failure, and cutoff censoring. We recompute exact distances on each layout's complete finite state graph. Distance relaxation runs to a fixed point or beyond the simple-path length bound before a state is labeled unsolvable. We classify each endpoint as solved, irreversibly unsolvable, or still solvable but unresolved at the cutoff. The last category means a solution exists, not that this policy will find it. Historical int16 records encoded both unreachable distance and padding as -1; the executed-action mask distinguishes them. New records preserve int32 unreachable values.

Table 8: Action-budget follow-up on puzzles with initial optimal distance 32–60. Entries are percentages averaged over the three training seeds, each evaluated on the same 500 puzzles. The final three columns partition outcomes at 128 actions (up to rounding). "Still solvable" denotes cutoff censoring, not observed future success. Per-seed counts and all distance bins are retained in the accompanying episode analysis.

Training condition	Solved 64 actions	Solved 96 actions	Solved 128 actions	Unsolvable at 128	Still solvable at 128
Action only	74.53	74.60	74.60	24.73	0.67
Latent prediction + state matching	75.87	76.00	76.07	23.53	0.40
State supervision	71.07	71.13	71.13	27.67	1.20
Action-effect supervision	76.40	76.53	76.53	23.13	0.33
Goal-occupancy supervision	70.60	70.60	70.60	28.27	1.13
State + action-effect supervision	72.73	72.73	72.73	26.67	0.60
State + goal-occupancy supervision	73.73	73.73	73.73	25.47	0.80
Action-effect + goal-occupancy supervision	76.00	76.00	76.00	23.47	0.53
All-component supervision	73.47	73.47	73.53	25.67	0.80

Repeated states alone are not treated as policy loops, because the history-conditioned policy can change its action on a revisit.

Results. In the 32–60 bin, the 27 runs yield 13,500 seed–episode evaluations on 500 distinct puzzles. Of 3,534 non-solves at 64 actions, 3,369 (95.3%) are already irreversibly unsolvable. Only 8 additional solves occur by 128 actions; 97 remain solvable at that cutoff. Across all distance bins, 9 additional solves occur in 40,500 evaluations. Mean far-bin gains range from zero to 0.2 percentage points per condition (Table 8). Action only reaches 0.746 and all-component supervision reaches 0.735 at 128 actions. The original endpoint already includes detours: 153 of action only's 1,118 far-bin solves across seeds take more actions than the initial optimal distance. The nine added pairwise-supervision checkpoints produce no additional far-bin solves, preserving their original solving comparison.

What can further continuation change? If S_B, U_B, and N denote solved, irreversibly unsolvable, and total episodes after budget B, any continuation preserving those trajectories has final solved fraction between S_B/N and $1 - U_B/N$. These are sample-specific logical bounds, not confidence intervals or unlimited-policy measurements. For the three-seed far-bin mean at 128, even solving every remaining solvable case would raise all-component supervision only to 74.33%, below action only's already observed 74.60%. Thus additional time alone cannot reverse that aggregate comparison on these realized trajectories; resets or undoing earlier pushes would change the protocol.

Length and generality limits. Training histories contain at most 64 actions, so extension also tests length extrapolation. The unchanged model has 192 learned positions and a 52-token initial prefix; it can predict at most 141 actions under this input convention. The 128-action evaluation fits that capacity but does not establish unrestricted eventual solving. The bounds constrain only continuation of this panel's observed trajectories, not unseen puzzles, other training seeds, or policies equipped with recovery mechanisms. This follow-up supports robustness of the reported behavioral contrast to the tested budget increase.

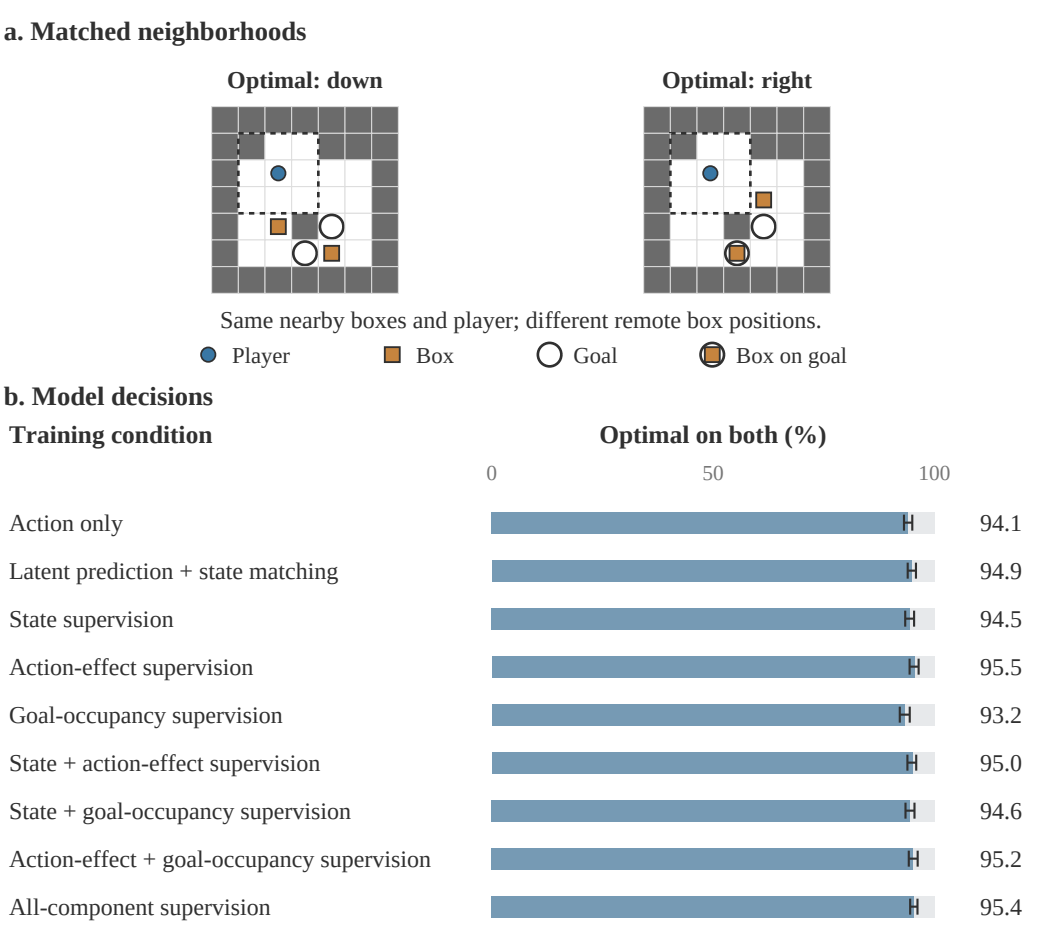

Figure 10: Matched-neighborhood test, all nine conditions at three seeds, including latent prediction + state matching. (a) States share walls, goals, player position and nearby boxes (dashed regions), but require different optimal actions. (b) Blue bars show mean both-optimal accuracy; gray remainders denote pairs with at least one non-optimal action. Whiskers are 95% structural-group bootstrap intervals conditional on the trained seeds. A deterministic current-local-only policy scores zero. Initial states and history lengths are unmatched.

E State-sensitive decisions and shortcut controls

E.1 Matched-neighborhood and matched-cue tests

Decisions requiring state distinctions. The original matched-neighborhood test fixes current radius-1 neighborhoods but requires disjoint optimal-action sets. Action-only training is optimal on both states in 0.941 of 4,540 pairs across three seeds (Figure 10); the radius-2 test gives 0.891. This excludes deterministic current-local-only policies, which must choose the same action in both states.

Question and construction. The original matched-neighborhood test leaves initial boards and history lengths unmatched. We therefore add a follow-up that rules out policies based on a larger set of specified cues. It was designed after the original results. From each of 4,024 held-out probe-set starts in 503 layouts, we sample 1,024 histories using the training detour law ($q = 0.5$, budget 64). Pair members have the same initial board and move count $k \geq 2$, different current states, and disjoint sets of all solver-optimal actions. Both remain solvable within the original budget. The matching tests hold successively more cues fixed. The *local-cue* test fixes the current player cell, radius-1 box occupancy and outcomes of all four candidate actions. The *last-two-moves* test also fixes the two most recent moves. The *primary test* additionally fixes the number of executed pushes. A stricter

variant matches the last four moves instead of two, while retaining the push-count constraint. A wider-neighborhood variant matches radius-2 occupancy and the last two moves, without fixing push count. Static walls and goals are fixed by the shared initial board. The *same-optimal-actions control* matches the primary test's cues and has different remote boxes but identical optimal-action sets. The two *neighborhood* tests reproduce the original radius-1 and radius-2 comparisons, which leave initial boards and history lengths unmatched.

An engine-only pilot determined construction size before model evaluation. We retain at most ten pairs per start and 20,000 per tier, with fixed sampling seeds, and freeze the set by SHA-256. Independent token decoding verifies every pair's matched cues and different required actions. Thirty synthetic tests cover alignment, oracle and cue-only policies, replay, freezing and uncertainty; separate scalar replay and exhaustive BFS verify 64 sampled primary members with zero discrepancies. No checkpoint is selected using these new outcomes.

Evaluation and uncertainty. We evaluate all nine final training conditions at three seeds, with greedy actions at token $51 + k$ on the fixed histories. These are history-conditioned next-action decisions, not new model-generated solving trajectories. A deterministic policy only of the matched cues has zero both-optimal accuracy on distinction pairs, since it must choose the same action for disjoint optimal-action sets. This zero bound does not apply to the same-optimal-actions control. The table reports three-seed means and ranges. Primary intervals use 2,000 shared structural-group bootstrap draws, averaging the three seeds on each pair before resampling groups; they quantify group uncertainty conditional on these trained seeds. Secondary tiers and strata are descriptive. Tiers select different pairs and history distributions, so their accuracy differences are not causal effects of adding a cue constraint.

Matched information	Pairs	Groups	Action only	All-component supervision
Current-neighborhood matching				
Neighborhood (radius 1)	4,540	153	94.1 [93.8, 94.3]	95.4 [94.9, 95.7]
Neighborhood (radius 2)	467	74	89.1 [88.9, 89.3]	90.1 [88.0, 91.6]
Same initial board and move count				
Local cues	20,000	157	88.0 [87.7, 88.3]	91.6 [91.4, 91.7]
Local cues + last two moves	20,000	157	89.9 [89.6, 90.1]	92.9 [92.7, 93.3]
Local cues + last two moves + push count (primary)	20,000	127	89.2 [88.9, 89.4]	93.1 [92.9, 93.2]
Local cues + last four moves + push count	17,158	126	89.1 [88.8, 89.4]	92.9 [92.7, 93.1]
Radius-2 cues + last two moves	13,166	106	73.5 [73.1, 74.1]	80.0 [79.8, 80.3]
Same optimal actions (control)	20,000	161	96.2 [96.1, 96.3]	97.4 [97.3, 97.5]

Table 9: Cue-matched decisions: both-optimal accuracy (%), mean [min, max] over three training seeds. The primary test matches local cues, the last two moves and push count. The same-optimal-actions control uses the same matched cues but identical optimal-action sets; all other rows require disjoint sets. The two neighborhood-only tests leave initial boards and history lengths unmatched.

Results and interpretation. Primary accuracy is 89.2% (95% interval [88.0, 90.3]%) for action-only training and 93.1% ([92.1, 94.0]%) for all-component supervision. The paired difference is +3.9 percentage points ([+3.4, +4.3]). The baseline therefore responds successfully to remote state differences even when these initial-board, local, recency and push-count cues are identical. Seed-1 action-only controls score 0.3% at initialization, 22.5% at update 1,825 and 33.7% at update 4,203, supporting a learned ability. Descriptively, baseline accuracy remains 89.2% when a member's box configuration lies on no shortest path from the start, and 88.1% when the differing-box information is at least eight actions old. Corresponding supervised accuracies are 93.1% and 92.6%.

These results go beyond excluding a local-only policy. They also show why the recovery–behavior comparison depends on the evaluation: supervision improves remote-state-dependent decisions even though it does not consistently improve unassisted solving. They do not identify a complete internal state model, exclude arbitrary functions of earlier actions, or establish that higher probe accuracy mediates the improvement. The auxiliary objectives change together in this comparison. Correctness

dependence and partner-error statistics saved with the raw results do not by themselves establish a causal shortcut mechanism. All nine conditions' primary results appear in Table 22; appendix training controls and developmental records are also archived.

Residual dependence and specificity. Exceeding zero excludes policies solely of the named cues, but initialization also exceeds zero: the large learned accuracy is the substantive evidence. The residual-dependence criterion in the protocol cannot resolve that remaining question at these accuracies. Its shortfall, $p_u p_v - P(\text{both correct})$, cannot exceed $(1-p_u)(1-p_v)$, whose largest value across final main runs is only 0.41 percentage points, below the 3-point margin. Same-key decoys cover only a small subset of one-wrong pairs. Failures retain same-action and partner-directed error patterns, including excess over the optimal-action-rank null; these are cue-consistent associations, not identified causal shortcut use.

The supervision gain also occurs on decision-irrelevant control pairs with identical optimal-action sets. Applying their proportional error reduction to the distinction pairs predicts 87.2% of the all-component gain. Some excess remains under group inference, but it is not stable under three-seed inference. Action-only training for twice the updates also reaches 92.5% both-optimal accuracy in a single-seed control (Table 23). Thus neither a remote-state-specific learning mechanism nor elimination of residual cue dependence follows from the high distinction accuracy.

A decoy-informative variant requires a third state sharing the primary cue key and a discriminating alternative-state error null for both pair members. Engine-only construction yields 1,394 pairs from 15 structural groups, below the fixed minimum of 30 groups. No model is evaluated on this enriched set; its limited coverage supplies no evidence for absent residual cue dependence.

E.2 Descriptive history-cue diagnostics

Conflict with history cues. The post-result shortcut audit evaluates next-action accuracy on fixed held-out detour positions. A plan-clock cue uses optimal-plan actions at the observed move count; a push-stage cue uses optimal-plan boxes after the observed push count with the current player. A conflict means that cue and true state require disjoint optimal-action sets. Conflict and agreement strata differ in difficulty and number of valid optimal actions, so raw differences are not causal effects. Table 10 reports the corrected standardisation for all conditions covered by that analysis, retaining the longer-training and optimal-only controls. The original broader strata remain archived.

	Seeds	Plan-clock conflict minus agree	Push-stage conflict minus agree	Off-plan minus on-plan
Action only	3	+0.02 [-0.28, +0.33]	-1.44 [-2.18, -0.67]	-1.54 [-1.99, -1.06]
Latent prediction + state matching	3	-0.03 [-0.28, +0.22]	-0.89 [-1.49, -0.30]	-0.81 [-1.15, -0.39]
Goal-occupancy supervision	3	+0.13 [-0.18, +0.47]	-1.71 [-2.50, -0.82]	-1.64 [-2.08, -1.13]
All-component supervision	3	+0.08 [-0.23, +0.42]	-0.88 [-1.51, -0.26]	-0.46 [-0.81, -0.04]
Action only twice the updates	1	+0.06 [-0.22, +0.32]	-1.00 [-1.57, -0.37]	-0.96 [-1.39, -0.46]
All-component supervision optimal trajectories only	1	-3.35 [-4.44, -2.27]	-2.71 [-4.91, -0.54]	-5.79 [-6.96, -4.47]

Table 10: Descriptive accuracy differences in percentage points, with 95% structural-group bootstrap intervals conditional on the trained seeds. Plan and stage contrasts standardise optimal-action-set size, current distance and on/off-plan status; the last contrast standardises set size and distance. This post-hoc adjustment neither randomises histories nor excludes unmeasured cues.

The three-seed conditions in this corrected comparison show no consistent final-checkpoint penalty specific to the plan clock; the optimal-only control does show a deficit. Stage conflict and off-plan states retain accuracy deficits. Such deficits locate harder cases without proving that a policy follows the conflicting cue.

Different histories reaching the same state. Natural optimal/detour rejoin pairs reach identical states at different move counts. The action-only model is optimal on both in 98.3% of final pairs across three seeds. The extra detour-minus-optimal accuracy change on plan-clock-conflict versus agreement pairs is +0.06 points (group interval [-0.11, +0.25]), with no established plan-clock penalty. This changes history and position together; it is not selective state erasure. Latent prediction and state matching was explicitly trained on rejoin positives, so its performance on held-out rejoin pairs is an objective-aligned diagnostic.

Development over training. The cue-matched development analysis follows action-only training, latent prediction and state matching, and all-component supervision across the saved checkpoints at three seeds. Early same-action behavior is already strong at initialization, so it cannot by itself establish that a shortcut was learned. Distinction accuracy improves later than accuracy on decisions settled by the matched cues. The gap persists after standardising history length and optimal-action-set size, while within-start analysis reveals dependence masked by aggregate accuracy. Both auxiliary-objective conditions acquire cue-matched competence earlier relative to their own endpoints; their timings do not separate a state-supervision-specific advantage. These post-hoc developmental comparisons are limited by checkpoint spacing and differences in task difficulty; they do not establish a causal progression from shortcuts to state tracking.

These audits are exploratory. The neighborhood comparison includes all three seeds with structural-group intervals recomputed from frozen pair records. The stricter matched-cue tests support the named policy exclusions; the descriptive conflict and rejoin audits do not exclude further heuristics.

F State readouts and interventions during behavior

F.1 Readout accuracy along model-generated rollouts

We apply the fitted state probe to states each policy visits, then compare episodes that solve with those that fail. This puts recovery and behavior on the same rollouts, although the comparison remains observational. Complete-state accuracy requires the player and both boxes to be correct at the same position. The early window includes the initial state and states after the first 16 moves. The reported accuracy averages over these positions; it is not the fraction of episodes whose entire state sequence was decoded correctly. Successful episodes can therefore have many decoding errors. This establishes that exact recovery by this linear probe at every early position is not required for observed success; it does not establish that the policy lacks state information. The episode-level regression adjusts for distance bin, leaving finer difficulty and trajectory confounds possible.

F.2 Removing state-probe directions

We next remove the measured state directions and test whether behavior changes, using same-rank random subspaces as controls. Removing a direction tests dependence on it; the subsequent repair experiments instead change its content.

F.3 Selective box-code replacement and one-decision repair

Question and design. The removal test asks whether behavior depends on the probe directions. Here we ask whether replacing their box-position content changes decisions, and whether supplying correct content improves solving. We change the hidden activation and pass it through the remaining model to choose an action. The fixed probe only verifies what the edited representation decodes to; changing that readout does not guarantee correction of every state representation used by the policy. This follow-up uses 1,424 cue-matched pairs from 103 structural groups, evaluated in both directions (2,848 directed cases). It reuses the histories in Section E.1, whose aggregate clean behavior was already known. In addition to the primary cue matches, both members have the same goal occupancy. Up to 16 pairs per group are selected by engine metadata and a fixed random seed, without model outputs. Action-only and all-component models each use three final training seeds. The layer is the previously validation-selected state-probe layer; the original linear probe is fitted on probe-training data only. Neither donor selection nor intervention fitting uses action gradients or intervention outcomes.

Table 11: Seed-1 complete-state decoding along each model's own confirmatory rollouts: all positions; 32–60-step episodes over their first 16 actions (solved / failed) and over the whole episode (solved / failed); p of the episode-level logistic regression of solving on first-16 decoding with the distance bin as covariate. These exploratory nominal p-values are not cluster-adjusted or evidence that readout errors cause failure.

	all	far, first 16		far, whole episode		p
		solved	failed	solved	failed	
Action only	0.228	0.409	0.294	0.288	0.136	1e-06
Latent prediction + state matching	0.379	0.560	0.472	0.430	0.263	0.003
State supervision	0.903	1.000	0.977	0.997	0.679	0.0006
Action-effect supervision	0.294	0.441	0.361	0.331	0.231	0.002
Goal-occupancy supervision	0.258	0.429	0.329	0.297	0.161	2e-05
All-component supervision	0.920	0.999	0.981	0.996	0.735	0.0005
State + action-effect supervision	0.922	1.000	0.988	0.999	0.740	0.008
State + goal-occupancy supervision	0.917	1.000	0.973	0.997	0.692	3e-05
Action-effect + goal-occupancy supervision	0.287	0.436	0.358	0.313	0.241	0.001

Table 12: Seed-1 mean-ablation of the 50 linear state directions at the selected layer (state readout before / after), 32–60-step solve rate and matched-neighborhood accuracy (both members optimal) without ablation, with the state ablation, and the range over five random subspaces of the same rank. These extrema are descriptive controls, not confidence intervals. All-component supervision has a pair-specific reduction below its control range despite no selective far-solve reduction.

	Layer	State	32–60-step solve rate			pairs both optimal		
		before / after	none	State	random	none	State	random
Action only	11	0.218 / 0.001	0.748	0.742	0.706–0.726	0.942	0.935	0.931–0.938
Latent prediction + state matching	11	0.388 / 0.000	0.754	0.710	0.718–0.776	0.949	0.919	0.940–0.944
All-component supervision	13	0.955 / 0.000	0.718	0.726	0.684–0.718	0.949	0.937	0.942–0.949

Protected box-logit replacement. Write the fitted probe as $z = hA + c$ in raw residual coordinates. Let N contain the player and goal columns of A, and B its box columns. With Q_N an orthonormal basis for N, define $B_\perp = (I - Q_N Q_N^\top)B$. For desired box-logit change d, the edit is $h' = h + dB_\perp^+$, using a pseudoinverse with relative tolerance 10^{-6}. The box map has rank 25 in all checkpoints. This changes the specified box logits while preserving the player and goal logits. Inputs, physical states and model weights are unchanged. We check achieved box logits, protected-logit drift and intervention norms on the actual edited activations.

The counterfactual arm uses the matched remote-state donor's natural box logits. Correct repair sets the true two box cells to $+2$ and other cells to -2; wrong repair uses the donor's valid box configuration with the same values. Five random controls match each example's intervention norm in the full probe's null space, preserving every probe logit. A distinct-history, same-state donor exists for 658 cases; its contrast uses that subset with a matching clean arm. Self replacement and restoration of original logits check algebraic identity, not scientific rescue. Full residual transfer is a sensitivity control that does not isolate box information from other variables.

Scope extension and numerical verification. The first assay edits only the current state position. After observing its small action effects, an explicitly exploratory extension applies the same edits to every observed state position at the same layer. Earlier states are obtained by replaying the observed histories; no future tokens are used. Player and goal logits remain protected at every position. The full-prefix residual control reproduces donor actions in every case. Historical donor boxes can conflict with the recipient's protected historical cues, so the partial intervention need not correspond to a coherent physical trajectory.

Initial runs used bfloat16 inference. Identical clean inputs under different batch arrangements had unchanged first actions but logit differences up to 0.25, motivating an FP32 repeat with TF32 disabled. We retain both versions; Table 13 reports FP32 verification with the same probes and examples. This precision audit was chosen after seeing the original outcomes; sensitivity to patching metrics and corruption methods motivates retaining the control results alongside each intervention (Zhang & Nanda, 2024).

Table 13: Selective box-information edits, three-seed FP32 means. Box decoding and optimal actions are percentages on the intervention set, shown as clean / counterfactual / correct repair, scored against the recipient state. The last column is the change in continued solve rate after the repaired first decision, in percentage points with a structural-group bootstrap 95% interval. These are different populations from the main probe and original-start rollout panels.

Training	Edited positions	Exact boxes clean / swap / repair	Optimal action clean / swap / repair	Solve change repair − clean
Action only	Current position	18.7 / 3.0 / 100.0	93.7 / 93.6 / 93.7	-0.09 [-0.25, +0.08]
All-component supervision	Current position	90.8 / 0.6 / 100.0	95.9 / 95.8 / 95.8	+0.00 [-0.13, +0.13]
Action only	All state positions	18.7 / 3.0 / 100.0	93.7 / 93.6 / 93.5	-0.21 [-0.44, +0.00]
All-component supervision	All state positions	90.8 / 0.6 / 100.0	95.9 / 95.6 / 95.1	-0.34 [-0.65, -0.09]

Decisions and subsequent solving. Correct repair makes box decoding exact by construction, but does not establish a solving benefit in either model or intervention scope. The original current-position assay has small donor-directed log-odds changes above its norm controls: 0.0108 (95% interval [0.0078, 0.0135]) for action-only and 0.1043 ([0.0850, 0.1238]) for all-component supervision. Extending the edit to all state positions gives -0.0001 ([-0.0065, 0.0065]) and 0.0219 ([-0.0052, 0.0477]), respectively. The full per-arm records include wrong repairs, same-state donors and all random controls; being less harmful than a wrong repair is not counted as improvement over clean behavior.

For the solving endpoint, a fixed subset of 784 directed cases receives the patched first decision and then continues under the unchanged policy, retaining the original board and complete move history. Arms choosing the same first action share the identical continuation. Evaluation stops at solution, proven irreversible failure, or total history length 128; still-solvable endpoints are reported separately. This is a one-decision intervention, not continuous oracle assistance. Eight CPU checks cover the edit constraints, null-space controls, token alignment, identity and donor-transfer controls, and continuation logic. Independent scalar replay and exhaustive BFS audit the stored trajectories.

Inference and interpretation. Each scope uses 5,000 shared structural-group bootstrap draws, retaining both orientations and repeated members and averaging the same three seeds. Intervals condition on these trained models. Four planned directional contrasts per model (counterfactual log odds versus random; correct-repair optimal-action probability versus clean, wrong repair and random) use Holm adjustment within each scope's eight-test family. The prefix extension and numerical audit are disclosed post-result diagnostics. Fine-grained probability effects do not establish large decision or solving benefits. Oracle logit targets may be off-manifold, and protected probe logits do not guarantee preservation of all other information. The result concerns the tested linear code, layer and edits: it neither proves that every state representation is unnecessary nor identifies the mechanism mediating the benefit of training-time supervision.

F.4 REPEATED CORRECTION DURING SOLVING

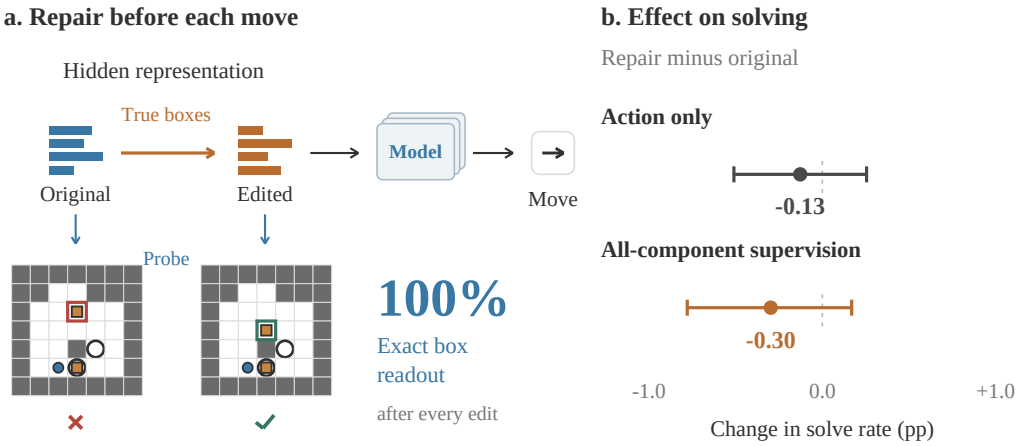

Figure 11: Repeated correction of a linear box code. (a) True box positions guide an activation edit; the fixed probe verifies exact box recovery, and the remaining model chooses the move. (b) Three-seed mean solving changes (percentage points) with 95% structural-group bootstrap intervals over 784 directed continuations. No solving benefit is established for this code and edit (Table 14).

To test whether one-decision edits miss accumulating state errors, we use the same 784 directed cases and all six checkpoints, keeping probes, layers, and target magnitudes fixed. A new forward pass before each action replaces the current position's box logits using that arm's actual current state. Earlier positions and the complete observed history remain unchanged; no future state or optimal-action target enters the edit. We compare clean, first-decision-only, and every-decision true repair, every-decision wrong repair, and five repeated null-space controls. Wrong targets are distinct valid two-box placements with the current player and goal-occupancy bits preserved. Each random control uses a fixed direction per member and matches the true-repair norm at its own visited state. Stopping rules match the one-decision experiment.

	Clean	First only	Repeated	Wrong	Random	Δ repeated − clean
Action only	88.27	88.18	88.14	88.05	88.09	-0.13 [-0.51, +0.26]
All-component supervision	92.60	92.60	92.30	92.35	92.45	-0.30 [-0.78, +0.17]

Table 14: Solving with repeated current-position box-code correction, three-seed means (%). First only and repeated use true box targets; wrong targets and the mean of five random controls are applied at every decision. The final column gives percentage-point changes and 95% structural-group intervals. These are assisted continuations on the fixed intervention sample.

All repaired decisions have exact box readout, but repeated correction does not establish a solving benefit (Table 14). Relative to first-only repair, changes are -0.04 points ([-0.41, +0.33]) for action-only training and -0.30 ([-0.79, +0.17]) for all-component supervision. Against mean random controls, they are +0.05 ([-0.34, +0.45]) and -0.14 ([-0.59, +0.30]), respectively. None of the eight directional solving contrasts passes Holm adjustment. All clean and first-only trajectories exactly reproduce the archived FP32 results. An independent scalar engine verifies all 42,336 arm trajectories; 594 exhaustive-BFS queries verify sampled distances with no discrepancies. Repeating the correction therefore does not rescue the earlier result under this code and edit; insufficient duration is not supported as its explanation. It still does not establish non-use of other encodings or remove the risk that oracle logit targets lie outside the model's natural representations.

G TRAINING-CONDITION COMPARISONS AND CONTROLS

G.1 THREE-SEED RESULTS FOR ALL NINE CONDITIONS

Table 15 collects the final readouts and behavior of all nine conditions, including latent prediction and state matching, with their seed ranges.

Table 15: All nine conditions: mean and [min, max] over three training seeds (n). State: complete-state accuracy; Action effect: worst-context accuracy; Goal occupancy: joint accuracy; 8-step retention and solve rate on the confirmatory panel.

	n	State	Action effect	Goal occupancy	8-step retention 16–31	Solve rate 16–31	Solve rate 32–60
Action only	3	0.208 [0.198, 0.217]	0.843 [0.835, 0.850]	0.622 [0.579, 0.694]	0.941 [0.934, 0.950]	0.969 [0.964, 0.974]	0.745 [0.740, 0.748]
Latent prediction + state matching	3	0.317 [0.272, 0.388]	0.913 [0.911, 0.914]	0.659 [0.647, 0.677]	0.931 [0.920, 0.944]	0.963 [0.956, 0.974]	0.759 [0.754, 0.764]
State supervision	3	0.958 [0.952, 0.962]	0.914 [0.904, 0.925]	0.774 [0.754, 0.789]	0.920 [0.916, 0.922]	0.951 [0.946, 0.958]	0.711 [0.692, 0.736]
Action-effect supervision	3	0.264 [0.240, 0.294]	0.952 [0.948, 0.955]	0.697 [0.644, 0.733]	0.945 [0.936, 0.950]	0.969 [0.964, 0.976]	0.764 [0.752, 0.786]
Goal-occupancy supervision	3	0.252 [0.237, 0.277]	0.817 [0.804, 0.831]	0.965 [0.963, 0.970]	0.927 [0.918, 0.936]	0.956 [0.952, 0.958]	0.706 [0.678, 0.734]
State + action-effect supervision	3	0.959 [0.958, 0.961]	0.962 [0.961, 0.963]	0.768 [0.761, 0.772]	0.925 [0.904, 0.936]	0.950 [0.938, 0.962]	0.727 [0.710, 0.746]
State + goal-occupancy supervision	3	0.953 [0.953, 0.954]	0.908 [0.900, 0.916]	0.970 [0.969, 0.970]	0.916 [0.912, 0.924]	0.948 [0.942, 0.954]	0.737 [0.730, 0.746]
Action-effect + goal-occupancy supervision	3	0.268 [0.254, 0.282]	0.947 [0.941, 0.950]	0.975 [0.972, 0.977]	0.943 [0.934, 0.956]	0.969 [0.964, 0.976]	0.760 [0.744, 0.768]
All-component supervision	3	0.955 [0.954, 0.957]	0.959 [0.956, 0.962]	0.971 [0.969, 0.973]	0.921 [0.916, 0.928]	0.955 [0.950, 0.960]	0.735 [0.718, 0.750]

G.2 COMPONENT INTERACTIONS AND PAIRED COMPARISONS

Exploratory factorial contrasts. Using all 24 explicit-supervision runs, we average each objective's within-seed high-minus-low difference across the four settings of the other two objectives. Adding state supervision changes 16–31-step solving by -1.48 percentage points (seed range [-2.10, -0.95]); all three seeds are negative. Its 32–60-step mean is -1.63 points, but the range [-3.95, +0.65] includes both signs. Adding action-effect supervision changes far-bin solving by +2.17 points (range [+1.85, +2.65]). These post-hoc descriptive contrasts support an objective- and difficulty-dependent tradeoff; three seeds do not establish population-level significance or equivalence.

G.3 LOSS WEIGHTING, EXPERIENCE, AND TRAINING AMOUNT

Loss budget and experience controls. All-component supervision carries three auxiliary terms of weight 1; one control trains it with each weight $1/3$, so the sum of its auxiliary coefficients equals one. Because the heads sum different numbers of loss terms, this matches only their sum, not loss magnitudes or gradient budgets. The experience controls vary what the model sees: the share of detour trajectories (0, 25, 75 and 100%, against 50% in the main corpus), the step-level detour probability of the generator ($q_{\text{detour}} = 0.75$; the 0.25 corpus failed its preregistered box against box minimum and was not trained), and the amount of data for action-only training (half and twice the layouts, same number of updates). An unsolvable-state exposure control was not run: it needs a target rule for states without an optimal action. Table 18 lists every control; the latent prediction and state matching decompositions are discussed in Section B.3.

Table 16: Complete 2^3 explicit-supervision factorial, three seeds per cell: mean and [min, max]. State: complete-state accuracy; Action effect: overall and worst-context accuracy; Goal occupancy: joint accuracy and number of seeds passing the goal-entry event test. Behavioral results are in Table 15. Ranges describe seed variation, not confidence intervals.

	State	Action effect		Goal occupancy	
	complete	overall	worst	joint	event passes
Action only	0.208 [0.198, 0.217]	0.981 [0.981, 0.982]	0.843 [0.835, 0.850]	0.622 [0.579, 0.694]	3/3
State supervision	0.958 [0.952, 0.962]	0.983 [0.982, 0.983]	0.914 [0.904, 0.925]	0.774 [0.754, 0.789]	3/3
Action-effect supervision	0.264 [0.240, 0.294]	0.991 [0.990, 0.991]	0.952 [0.948, 0.955]	0.697 [0.644, 0.733]	3/3
Goal-occupancy supervision	0.252 [0.237, 0.277]	0.978 [0.976, 0.979]	0.817 [0.804, 0.831]	0.965 [0.963, 0.970]	3/3
State + action-effect supervision	0.959 [0.958, 0.961]	0.990 [0.990, 0.990]	0.962 [0.961, 0.963]	0.768 [0.761, 0.772]	3/3
State + goal-occupancy supervision	0.953 [0.953, 0.954]	0.981 [0.981, 0.981]	0.908 [0.900, 0.916]	0.970 [0.969, 0.970]	3/3
Action-effect + goal-occupancy supervision	0.268 [0.254, 0.282]	0.990 [0.990, 0.990]	0.947 [0.941, 0.950]	0.975 [0.972, 0.977]	3/3
All-component supervision	0.955 [0.954, 0.957]	0.990 [0.989, 0.990]	0.959 [0.956, 0.962]	0.971 [0.969, 0.973]	3/3

Table 17: Seed-1 paired comparison of every condition with action-only training on the same confirmatory episodes: change in solve rate (episodes solved only by the condition / only by action-only training) and the two-sided exact McNemar p, uncorrected, per distance bin.

	8–15		16–31		32–60	
	Δ (+/−)	p	Δ (+/−)	p	Δ (+/−)	p
Latent prediction + state matching	+0.000 (0/0)	1	+0.000 (9/9)	1	+0.006 (50/47)	0.839
State supervision	-0.006 (0/3)	0.25	-0.016 (8/16)	0.152	-0.056 (37/65)	0.00721
Action-effect supervision	+0.000 (0/0)	1	+0.002 (6/5)	1	+0.038 (54/35)	0.0558
Goal-occupancy supervision	+0.000 (0/0)	1	-0.016 (8/16)	0.152	-0.042 (35/56)	0.0354
All-component supervision	-0.002 (0/1)	1	-0.024 (4/16)	0.0118	-0.030 (41/56)	0.155
State + action-effect supervision	-0.002 (0/1)	1	-0.036 (7/25)	0.0021	-0.038 (43/62)	0.0785
State + goal-occupancy supervision	+0.000 (0/0)	1	-0.032 (5/21)	0.00249	-0.018 (40/49)	0.397
Action-effect + goal-occupancy supervision	+0.000 (0/0)	1	-0.010 (8/13)	0.383	+0.020 (51/41)	0.348

Rescaling all-component supervision's auxiliary coefficients gives similar recorded readouts and solving in this single-seed control (state 0.958, 32–60-step solving 0.736, against 0.955 and 0.718 for all-component supervision).

The optimal-only control tests whether exposure to detours matters. With only optimal trajectories in training, all-component supervision reads the state in 0.997 of optimal-history positions but only 0.230 of detour-history positions, and acts optimally on only 0.511 of the latter; with 25, 50, 75 or 100% detours the state is recovered in 0.938, 0.955, 0.962 and 0.971 of positions, while 32–60-step

Table 18: Single-seed appendix controls at their final checkpoint. Readout columns follow Table 16, except that the event flag is yes/no. Solve rates cover the three distance bins; the last column is held-out rejoin retrieval.

	State	Action effect		Goal occupancy		Solve rate			Matching
	complete	overall	worst	joint	at event	8–15	16–31	32–60	retrieval
Latent prediction only	0.209	0.981	0.857	0.630	yes	1.000	0.966	0.710	0.02
State matching only	0.335	0.987	0.910	0.622	yes	1.000	0.974	0.782	0.96
Latent prediction + state matching horizon {1}	0.304	0.990	0.932	0.702	yes	1.000	0.968	0.766	0.96
Latent prediction + state matching horizons {1,2,4}	0.341	0.987	0.908	0.593	yes	1.000	0.960	0.728	0.96
All-component supervision coefficients 1/3	0.958	0.990	0.959	0.967	yes	1.000	0.946	0.736	–
All-component supervision optimal only (0%)	0.401	0.806	0.416	0.695	yes	0.996	0.936	0.680	–
All-component supervision 25% detours	0.938	0.985	0.932	0.964	yes	0.996	0.948	0.714	–
All-component supervision 75% detours	0.962	0.991	0.960	0.972	yes	0.998	0.952	0.726	–
All-component supervision detours only (100%)	0.971	0.993	0.969	0.977	yes	0.998	0.952	0.752	–
All-component supervision $q_{\text{detour}} = 0.75$	0.918	0.981	0.938	0.939	yes	1.000	0.956	0.684	–
Action only half the layouts	0.215	0.978	0.829	0.669	yes	1.000	0.952	0.648	–
Action only twice the layouts	0.223	0.982	0.873	0.661	yes	1.000	0.976	0.770	–
Action only twice the updates	0.248	0.985	0.892	0.702	yes	1.000	0.984	0.802	–

solving stays between 0.680 and 0.752. A higher step-level detour probability ($q = 0.75$) behaves like the main corpus (state 0.918).

Data scale affects behavior more than linear state recovery in action-only training. With half, the full, and twice the layouts the state is read in 0.215, 0.217 and 0.223 of positions while 32–60-step solving rises from 0.648 to 0.748 and 0.770; within this tested range, data scale changes solving more than linear state recovery.

A similar pattern appears when training lasts twice as many updates. The longer action-only run reads the state in 0.248 of positions and solves 0.802 of the far episodes, so the longer run improves solving without a comparably large change in linear state recovery. These controls do not exclude effects at other scales.

G.4 Relative development over training

The checkpoint crossings below summarize relative progress on each metric for action-only training, all-component supervision, and the latent comparison. Figure 6 shows the main conditions' development curves; Section E gives the complementary cue-matched analysis.

Table 19: Seed-1 relative response timing: the first checkpoint (in updates of 256 trajectories) at which a metric has covered half of its rise from initialization to the end of training (t_{50}). State: player-cell and complete-state accuracy; Action effect: worst-context accuracy; Goal occupancy: joint accuracy; retention: 8-step retention in the 16–31 bin; solve rate: the 8–15 and 32–60 bins. Each metric uses its own endpoints; these grid crossings are not absolute formation thresholds and have no seed confidence intervals.

	Player-cell accuracy	Complete-state accuracy	Action effect	Goal occupancy	8-step retention	Solve rate 8–15	Solve rate 32–60
Action only	1,202	4,203	4,203	6,379	4,203	1,825	22,298
Latent prediction + state matching	1,202	6,379	4,203	9,681	4,203	1,825	22,298
All-component supervision	1,202	1,825	2,769	2,769	4,203	1,825	14,692

G.5 External symbolic search

External search procedure and scope. This follow-up asks whether recovered rules can support a separate symbolic solver. After the main results, we applied a criterion from an earlier protocol revision: state ≥ 0.50, all five transition branches recovered, joint goal occupancy ≥ 0.70 with the event test, and a recovered goal rule. The qualifying conditions are State supervision, All-component supervision, State + action-effect supervision, State + goal-occupancy supervision. In each, exact search solves every confirmatory episode from both the true and decoded initial state. Walls, coordinates, transition signatures, update operations, and exact search are supplied by the experimenter. Initial-state decoding is a consistency check, not a test of long-history tracking, since the initial board is in the input.

Table 20: Seed-1 external symbolic sufficiency (conditional follow-up; numerical criterion applied after the main results). Criteria: state, $\hat{T}$ (5 of 5 branches), goal occupancy and $\hat{G}$ recovered. Solve rates 8–15 / 16–31 / 32–60 of exact search on the recovered symbolic model from the true start, from the model's decoded start, and the model-generated rollout.

	Criteria	$\hat{T}$	search, true start	search, decoded start	Model
Action only	closed	4/5	–	–	1.000 / 0.974 / 0.748
Latent prediction + state matching	closed	4/5	–	–	1.000 / 0.974 / 0.754
State supervision	open	5/5	1.000 / 1.000 / 1.000	1.000 / 1.000 / 1.000	0.994 / 0.958 / 0.692
Action-effect supervision	closed	4/5	–	–	1.000 / 0.976 / 0.786
Goal-occupancy supervision	closed	4/5	–	–	1.000 / 0.958 / 0.706
All-component supervision	open	5/5	1.000 / 1.000 / 1.000	1.000 / 1.000 / 1.000	0.998 / 0.950 / 0.718
State + action-effect supervision	open	5/5	1.000 / 1.000 / 1.000	1.000 / 1.000 / 1.000	0.998 / 0.938 / 0.710
State + goal-occupancy supervision	open	5/5	1.000 / 1.000 / 1.000	1.000 / 1.000 / 1.000	1.000 / 0.942 / 0.730
Action-effect + goal-occupancy supervision	closed	4/5	–	–	1.000 / 0.964 / 0.768

H CONTINUING FROM A HISTORY VERSUS AN EXPLICIT CURRENT BOARD

H.1 MATCHED HISTORY AND CURRENT-BOARD INPUTS

Design. To separate continuation from history from solving a freshly observed state, we compare two inputs for the same physical board and remaining action budget. This follow-up was designed after the main results. We sample 4,096 training-law detour walkers per start from the 16–31 and 32–60 bins of the confirmatory and development panels. At action counts $j \in \{8, 12, 16\}$, valid handovers have remaining distance $d \geq 8$ and $j + d \leq 60$. *On-plan states* lie on some shortest path from the initial board. *Detours with on-plan boxes* leave those paths but retain a box configuration on some shortest path. For *off-plan boxes*, the box configuration lies on no shortest path, and the current optimal actions are disjoint from three cues: the optimal-plan actions at depth j, the actions implied by initial box positions and the current player, and those implied by every optimal-plan box configuration after the same push count. At most four handovers per start per class are selected before evaluation. The frozen set contains 5,079 handovers; the panel reuse and construction-size choice from an engine-only pilot are disclosed in the execution record.

The *history* arm receives the original board and the sampled prefix and continues greedily. The *current-board* arm receives the exact handover board with empty history. Both start from the identical physical state and have $64 - j$ actions, with no legality masking. Neither arm receives further state observations. Because the histories are sampled, they need not be histories on which the policy would fail. The test measures controlled continuation rather than explaining every original rollout failure. Re-rooting changes both state availability and the history/position context; its effect is operationally a history-conditioned continuation cost, not a pure intervention on internal world knowledge.

Validation and uncertainty. Twelve synthetic tests validate classification, alignment, budgets and the evaluation pipeline. An engine oracle solves every real handover in both arms with identical actions/distances. Independent scalar replay and exhaustive BFS audit 192 learned-policy continuations (534 unique distance queries), with no discrepancies. All reported solve and optimal-action outcomes are recomputed from the archived actions/distances. We evaluate all nine conditions at three seeds and the protocol's two sensitivity controls, an early action-only checkpoint and an optimal-only trained model; both controls have a larger off-plan-box recovery effect than action-only or all-component training in both pools under group-bootstrap intervals.

Intervals use 2,000 structural-group draws, shared across seeds and conditions, conditional on the three trained models. The initial matching allows on-plan-box detour partners from another start with the same action count, distance bin, non-optimal-action count bin and last-two-action optimality flags. Before learned-model evaluation, a dependency audit found that these cross-group matches connect 84.6% of confirmatory pairs into one component. We therefore fixed same-start matches as the primary inferential subset: 220 pairs in 140 groups on the confirmatory pool and 360 in 190 on the development pool. All-pair estimates remain archived with a component-bootstrap sensitivity analysis; their concentrated dependence precludes precise population inference.

			Action only		All-component supervision	
Pool	Handover state	N	History	Board	History	Board
Confirmatory	On-plan state	674	87.9	91.2	90.9	91.8
Confirmatory	Detour, on-plan boxes	648	88.3	93.9	90.1	93.0
Confirmatory	Off-plan boxes	679	81.4	92.3	87.6	90.7
Development	On-plan state	1041	85.9	91.3	90.3	91.6
Development	Detour, on-plan boxes	988	86.2	91.4	89.7	90.9
Development	Off-plan boxes	1049	83.3	90.8	90.0	92.2

Table 21: Continued solving (%), averaged over three seeds, with the same physical handover state and remaining budget. "Board" supplies the exact current board and resets history. Detours with on-plan boxes control for history deviations without off-plan box configurations.

Conditional behavioral benefit. On confirmatory handovers with off-plan boxes, all-component supervision improves history-arm solving from 81.4% to 87.6%: +6.2 percentage points, group

interval [+3.8, +8.6]. The development pool shows the same direction, 83.3% versus 90.0% (difference +6.7, interval [+4.5, +8.9]). Supplying the true current board gives 92.3% and 90.7% on confirmatory handovers, and 90.8% and 92.2% on development handovers. Thus supervision improves actual continued solving in this history-conditioned setting; its benefit cannot be dismissed from the original-start solve rates alone.

What the stronger control resolves. Define continuation cost as current-board minus history solve rate. The primary same-start contrast subtracts the cost for detours with on-plan boxes from the cost for off-plan boxes, then compares supervised minus baseline models. The result is -2.4 percentage points (95% interval [-7.2, +2.2]) on confirmatory starts and -1.2 ([-4.9, +2.2]) on development starts. The confirmatory interval includes zero and extends beyond the specified ± 5-point equivalence margin. First-action intervals also do not lie inside the specified ± 3-point margin in both pools. We therefore establish neither equivalent tracking costs nor an additional cost reduction specific to off-plan boxes after controlling for general detour histories. The solving benefit therefore remains, but the control does not attribute it specifically to tracking boxes that have left every optimal plan. No complete world-model mechanism or elimination of all history shortcuts is inferred from this intervention.

H.2 Enlarged same-start comparison

A subsequent frozen evaluation matches within each start and the original strata before imposing a cap of 12 pairs per start, yielding 417 confirmatory and 612 development pairs. It reuses the start pools with a new engine-generated history sample; this changes the matched population and is not an independent replication. A synthetic sensitivity control acts optimally while boxes remain on any original optimal plan, but projects off-plan boxes onto stale plan-stage locations. Its re-rooted arm solves every case, and its first action is correct on player-only detours but wrong on off-plan handovers with defined cues. This deliberately oracle-gated corruption validates sensitivity, not a proposed mechanism of the learned model.

Across all three seeds, all-component supervision improves forced off-plan solving by +4.72 points (group interval [+2.31, +7.69]) on confirm and +5.56 ([+3.04, +7.93]) on development. Its extra gain over matched player-only detours remains uncertain: +1.84 ([-0.62, +4.51]) and +1.20 ([-1.22, +3.90]), respectively. The original solve tracking-cost contrast is -2.72 ([-6.38, +0.73]) on confirm, still crossing the negative five-point margin and zero. First-action contrasts differ across pools: -2.72 ([-5.41, -0.25]) versus +1.47 ([-0.68, +3.58]). The larger sample and selective sensitivity control preserve the continuation benefit, but still do not establish a mechanism specific to off-plan boxes. Intervals condition on the three trained seeds; all per-seed estimates and the original results remain available.

H.3 All conditions and training-amount controls

Table 22 compares all nine conditions at three seeds on the cue-matched decision test and forced-history continuation test (Sections E.1 and H). Each column answers a different behavioral question on a fixed evaluation population, so its percentage should be interpreted within that test. Training, frozen-set and checkpoint hashes are retained with the raw records. The reporting script recomputes optimal-action membership, solve indicators from stored distance trajectories, and seed means before exporting these tables.

All nine conditions use the same frozen handovers and evaluation code (Table 22). Conditions containing state or action-effect supervision improve history-arm solving with off-plan boxes in both pools, whereas goal-occupancy supervision alone does not. These are secondary endpoints. The matched excess in the all-component history-arm benefit for off-plan boxes over detours with on-plan boxes is only +2.73 points (95% group interval [-1.07, +6.30]) on confirmatory starts and +1.57 ([-0.75, +4.07]) on development starts. Gains also occur with on-plan boxes; the primary difference-in-differences does not establish an off-plan-specific advantage.

The primary confirmatory contrast's bootstrap standard error exceeds the protocol's precision target, so its inconclusive interval is not evidence of equivalence. The positive controls establish general sensitivity to history processing, not sensitivity specifically to off-plan boxes. In exploratory seed-1 controls on the same handovers, longer action-only training reaches 87.0% and 89.3% history-arm

	Matched-cue decisions both optimal Primary tier	Forced-history solving Confirmatory	Forced-history solving Development
Action only	89.2 [88.9, 89.4]	81.4 [78.8, 83.9]	83.3 [82.0, 84.0]
Latent prediction + state matching	91.6 [91.3, 91.7]	85.8 [84.4, 86.6]	86.7 [85.9, 87.8]
State supervision	92.7 [92.4, 93.0]	86.5 [85.1, 87.5]	89.3 [89.0, 89.5]
Action-effect supervision	92.9 [92.7, 93.2]	86.5 [86.0, 87.0]	87.9 [85.3, 89.6]
Goal-occupancy supervision	88.3 [87.5, 88.8]	81.3 [80.3, 82.6]	81.7 [80.3, 83.1]
State + action-effect supervision	93.4 [93.3, 93.5]	86.5 [85.3, 87.3]	88.4 [87.3, 90.1]
State + goal-occupancy supervision	91.9 [91.8, 92.2]	85.6 [84.1, 86.7]	88.8 [88.1, 89.4]
Action-effect + goal-occupancy supervision	92.4 [91.8, 92.6]	85.7 [85.6, 85.9]	87.4 [86.0, 89.3]
All-component supervision	93.1 [92.9, 93.2]	87.6 [87.3, 87.9]	90.0 [88.8, 92.2]

Table 22: All nine conditions: percentages, mean [min, max] over three seeds. Decisions use the 20,000 primary cue-matched pairs; continuation uses handovers with off-plan box configurations. These seed ranges are not confidence intervals. Population-specific uncertainty and controls are described in Sections E.1 and H.

	Matched-cue both optimal Primary tier	Forced-history solving Confirmatory	Forced-history solving Development
Action only	88.9	83.9	82.0
All-component supervision	93.1	87.6	88.8
Action only, twice the updates	92.5	87.0	89.3
Action only, half the layouts	86.8	76.4	78.1
Action only, twice the layouts	89.6	86.0	84.3

Table 23: Training-amount controls on the same frozen tests (%), all seed 1. The first two rows are their seed-matched references, not three-seed averages. The longer run doubles the updates; half/twice-layout runs retain the original update count. These exploratory controls are not a replicated equivalence test.

solving on the confirmatory and development pools. Data-scale controls also change performance (Table 23). These controls preserve the observed benefit at matched updates but leave training amount or efficiency as competing explanations; they do not establish equivalence across training seeds.

H.4 Training amount and alternative prediction targets

A three-seed 2×2 comparison crosses action-only versus all-component supervision with the original versus twice-longer training schedules. The longer runs start from initialization and stretch the same warmup/cosine schedule to 102,720 updates. Architecture, corpus, batch size, and optimizer are shared; auxiliary heads and weights are unchanged. This matches the number of updates and examples, but auxiliary objectives still add FLOPs.

Two further three-seed controls replace current-state targets with a learnable alternative-world state, or replace latent prediction by future-state prediction. We evaluate these controls on the frozen, enlarged same-start history panel. They extend the original study without providing an independent test population.

Common endpoint and uncertainty. The primary endpoint is history-arm solving on 417 confirmatory off-plan handovers in 141 structural groups, paired to player-detour controls from the same starts. Table 24 reports every trained seed. Table 25 contrasts the same handovers across conditions: 2,000 structural-group bootstrap draws condition on the three trained seeds; the separate seed-t intervals use three seed-level differences. With only three seeds, the latter are imprecise and rely on a normality assumption. Neither interval establishes equivalence, an asymptotic limit, or a causal mediator.

Training condition	State	Solve	History	History seeds 1 / 2 / 3
Action only	20.8	74.5	82.6	83.2 / 82.5 / 82.0
All-component supervision	95.5	73.5	87.3	88.5 / 86.3 / 87.1
Action only, twice the updates	23.5	81.9	89.6	87.1 / 91.8 / 89.9
All-component, twice the updates	96.8	79.5	90.3	91.1 / 89.4 / 90.4
State supervision	95.8	71.1	86.4	87.5 / 88.2 / 83.5
Alternative-world state supervision	34.6	52.1	68.2	70.3 / 65.5 / 68.8
Latent prediction + state matching	31.7	75.9	87.3	87.8 / 87.3 / 86.8
Future-state prediction + state matching	96.4	74.8	86.7	88.2 / 86.8 / 85.1

Table 24: Objective and training-amount controls. All entries are percentages. State is the complete-state linear-probe score; Solve is original-start solving on 32–60-step puzzles; History is the matched forced-history endpoint. These three metrics use different evaluation populations. The first three numeric columns are three-seed means.

History-solving contrast	Difference	Group 95% interval	Seed-t 95% interval
Supervision, original updates	+4.72	[+2.31, +7.69]	[+2.80, +6.63]
Supervision, twice the updates	+0.72	[-2.71, +3.73]	[-7.34, +8.78]
More updates, action only	+7.03	[+4.51, +10.07]	[-0.07, +14.14]
More updates, all-component	+3.04	[+0.87, +5.36]	[+2.13, +3.95]
Supervision-by-updates interaction	-4.00	[-7.44, -1.11]	[-10.37, +2.37]
Alternative state minus action only	-14.39	[-18.21, -10.94]	[-20.07, -8.71]
Alternative state minus state supervision	-18.23	[-22.37, -14.61]	[-28.56, -7.89]
Future state minus action only	+4.16	[+1.41, +7.09]	[+1.75, +6.56]
Future state minus latent prediction	-0.56	[-3.83, +2.42]	[-3.25, +2.13]
Future state minus state supervision	+0.32	[-1.23, +2.01]	[-3.65, +4.29]

Table 25: Paired differences in percentage points on the common matched history endpoint. Group and seed intervals answer different uncertainty questions; a group interval excluding zero is not a seed-generalization result.

Training amount. The supervision gap shrinks from 4.7 to 0.7 points when both conditions receive twice the updates. The interaction is negative in all three seeds. Its group interval excludes zero, but its seed interval does not. This supports attenuation for these trained models and makes training efficiency a plausible contributor, without proving that the entire supervision benefit is an optimization effect. Longer action-only training also improves original-start solving while complete-state recovery remains low.

Alternative-world state. This control replays the real move history on the original board with a fixed permutation: up becomes right, down becomes left, left becomes up, and right becomes down. Inputs and real optimal-action targets do not change; only the state-head targets change. Head, attachment site, coefficient, and training schedule match state supervision. Scalar replay, prefix alignment, physical transition branches, and pilot loss descent are checked before full training. Alternative-world states may deadlock and are not filtered for solvability. This target worsens continuation relative to both action only and real-state supervision in every seed. It refutes a benefit from this particular substitute tracking task; unequal target difficulty and gradient alignment prevent a claim that true state information is the unique cause of the real-state benefit.

Future-state prediction. The action-conditioned predictor retains the latent objective's site-8 projection, residual recurrent MLP, horizons $\{1, 2, 4, 8\}$, and state-matching term. A shared player/box

decoder replaces the latent prediction loss with future dynamic-position labels from the engine. It consumes the current representation and intervening actions, with no future-state input. The loss averages valid state/horizon pairs; the extra decoder changes parameter count and loss scale. This is symbolic supervision, not a label-free objective. Complete-state recovery reaches 96.4%, while original-start solving is 74.8% and matched-history solving is 86.7%. The history improvement over action only is positive in all seeds; differences from latent prediction and from current-state supervision have intervals spanning zero. Thus a predictive objective can also yield high symbolic recovery, while the behavioral comparisons do not follow the ranking of recovery scores. No equivalence between the objectives is inferred.

Gradient diagnostic. A separate descriptive audit evaluates five conditions, three seeds, and three stored training stages on the same eight training minibatches. It measures unclipped FP32 auxiliary/action gradient norms and cosines globally and by block. Goal-occupancy supervision supplies nonzero gradients but differs from action-effect supervision in magnitude and alignment. It is consequently not a matched-gradient control. These measurements do not identify optimization as a mediator or isolate information content from optimization geometry.

I Protocol provenance and reproducibility

Protocol history and follow-ups. The protocol records revision 1 at 03:55 EDT and revision 2 at 04:37 EDT on 2026-09-24, with the nine conditions, effect target, structural objective, and pair-audit rules specified before their main measurements. Revision 2b at 04:55 expanded the readout's context to the full prefix. A 05:05 record changed a confirmatory-panel fallback after the original group lists proved unavailable; the final panel did not use the two-episodes-per-group fallback. At 05:34, after a pipeline test on intermediate action-only training checkpoint 2769, the attention-pooling probe was changed to the residual form in Section C.1. The design history was later rewritten for repository housekeeping, with the old-to-new commit mapping retained internally. These internal records document how the design developed; they should not be treated as an independently timestamped public preregistration.

After main results were available, follow-up specifications covered the functional test (09:55), experience controls (11:10), symbolic sufficiency (11:15), model-rollout and probe controls (12:30), and subspace ablation (13:30). The sufficiency criterion used numerical thresholds from revision 1 only after results were known, because revision 2 did not specify them. Functional pairs were frozen before models were evaluated on those pairs. These are conditional or post-result follow-ups, not independent prospective confirmation. The archived protocol preserves amendments and deviations.

Submission-stage corrections. The original functional-use implementation resampled individual pairs, despite shared structural groups. We recompute those intervals by resampling structural groups from the original frozen pairs across all nine conditions and three seeds (Figure 10). Historical records remain archived; interval recomputation does not retrain or select checkpoints.

Environment. Recorded run metadata identifies PyTorch 2.4.1 with CUDA 12.1 and NVIDIA RTX A6000 GPUs. A complete original package lock was not retained in the supplied repository. The separate CPU validation environment is not the training environment. Full reproduction requires the original partition, data arrays, and checkpoints; CPU fixtures validate code paths, not experimental accuracy.

J Extended related work

Predictive world models and useful state abstractions. A world model need not reproduce a human-readable board to support action. Predictive-state representations describe state through predictions of future observations conditioned on actions (Littman et al., 2001). World Models and PlaNet learn latent environment dynamics for control (Ha & Schmidhuber, 2018; Hafner et al., 2019); MuZero predicts reward, policy, and value for planning without reconstructing observations (Schrittwieser et al., 2020). DeepMDP relates latent reward and transition prediction to state abstraction (Gelada et al., 2019). These approaches give a reason to keep symbolic recovery and behavioral

utility separate: the coordinates convenient for an observer need not be the coordinates used by a successful policy.

Our experiments do not compare the control performance of those algorithms. They hold the policy architecture and training experience fixed while varying auxiliary knowledge targets. The question is whether making particular environment information recoverable also helps that policy act. This is an empirical comparison within Sokoban, not a test of whether world models are useful in general.

World representations that emerge from sequences. Li et al. (2023) recover an Othello board representation and show that editing it can change move predictions. Nanda et al. (2023b) find a linear representation when cells are described relative to the mover. This change of target is especially relevant here: a weak readout can reflect the chosen description rather than missing information. Chess models likewise provide evidence of state tracking and latent-variable estimation (Toshniwal et al., 2022; Karvonen, 2024). Related work studies spatial and temporal representations in language models (Gurnee & Tegmark, 2024) and maze representations in Transformers (Ivanitskiy et al., 2024).

We build on this use of environments with known state. Our comparison separates current positions, action consequences, and goal occupancy, then asks whether supervising each component improves complete solutions and continuation from supplied histories. The input restriction—an initial board followed by moves— is shared with earlier sequence studies and is not itself our contribution.

Evaluating knowledge beyond predictive success. Vafa et al. (2024) assess whether a generative model recovers the distinctions and equivalences of an underlying finite-state world. They find incoherence that standard predictive diagnostics miss. Our complementary question concerns the behavioral value of explicitly supervised components. Exact engine labels let us measure recovery, construct states with matched cues but different optimal actions, and compare solving under different objectives. High matched-pair accuracy rules out the specified cue-only policies; it does not certify a complete or uniquely identified internal world model.

Probes and causal relevance. Linear probes measure information accessible to a chosen decoder (Alain & Bengio, 2016). Control tasks and analyses of task relevance show why that information cannot automatically be treated as a mechanism of behavior (Hewitt & Liang, 2019; Belinkov, 2022; Ravichander et al., 2021). We use structural-group splits, several probe families, and random-network and raw-embedding controls for attention pooling. Activation interventions then ask a different question: does changing this measured code change decisions or solving? Causal abstraction and patching work motivates that distinction and the need for intervention controls (Geiger et al., 2021; Zhang & Nanda, 2024). In maze models, Spies et al. (2025) find that activating connectivity features is easier than suppressing them. Our unsuccessful box-code repairs likewise have a narrow interpretation; they do not establish non-use of other encodings.

Learning objectives and the course of training. Li et al. (2025) show that intermediate tasks can favor different state-tracking mechanisms. Brenner et al. (2026) connect grid-walker representations to predictive statistics and architecture. These studies motivate varying learning objectives rather than inspecting only one final model. Our latent comparison uses action-conditioned prediction and state matching; related objectives include self-predictive representations (Schwarzer et al., 2021) and contrastively trained structured world models (Kipf et al., 2020). Engine equality labels supply the matching relation, so this comparison is not entirely free of environment supervision. Future-state and alternative-world targets further test which aspects of the auxiliary task matter, without matching their gradient geometry.

Our checkpoint crossings track relative improvements in knowledge and behavior, following analyses of induction heads, grokking, and circuits (Olsson et al., 2022; Power et al., 2022; Nanda et al., 2023a; Liu et al., 2022; Tigges et al., 2024). They do not establish a universal formation sequence.

Planning in Sokoban and related games. Prior studies examine Sokoban planning (Guez et al., 2019; Racanière et al., 2017), planning beyond model learning (Hamrick et al., 2021), and chess look-ahead (Jenner et al., 2024). Taufeeque et al. (2025) identify a causal future-plan representation and cycles that provide extra computation in a recurrent Sokoban policy. We test current-state supervision in history-conditioned Transformers; identifying planning abstractions beyond our probes requires different interventions.