Do Retrieved Errors Improve Decisions?

A Prospective Audit of Memory-Corrected Visual Planning

Daniel Lobato Garcia*

September 2026

Abstract

A world model predicts what will happen when an agent takes actions. Wrong predictions can make a bad plan look promising. Can remembering past prediction mistakes help the agent choose better plans?

We test this with an agent navigating through a doorway. Across 24 new navigation tasks and two fixed world models, we compare ways of using memory to score the same candidate plans. We find weak evidence of a small improvement from individual corrections over a shared correction to eligible plans. The improvement concerns visual similarity to the goal: actual goal-reaching changes little, and advantages over uncorrected or shuffled-memory scoring remain uncertain.

We also examine why planning fails: sometimes the candidate plans contain no successful route, sometimes useful plans are excluded from correction, and sometimes the scoring method chooses poorly.

Keywords: World models; model-based planning; episodic memory; retrieval; cross-entropy method (CEM); prediction error.

1 Introduction

We study whether memories of a world model's past errors can improve its choice of actions. For each proposed plan, a memory supplies errors observed on similar actions and starting states. These errors adjust the model's predicted cost before selection. The agent's task is to reach a goal on the other side of a wall by passing through a doorway. We compare the plans selected with each correction and measure their outcomes in the simulator.

Visual world models optimize actions against a learned feature-space objective [1, 4]. Prediction accuracy and decision quality can diverge [8, 9]: a common shift in all predicted costs preserves their ordering, while a small change near the minimum can change the selected plan. This motivates our question: *which part of a retrieved error correction improves the choice?*

We report two studies and an oracle analysis in a Wall navigation environment:

1. A matched-memory intervention tests whether errors collected on optimizer-selected actions transfer better than errors collected on transferred recorded actions. The predeclared 24-task comparison is inconclusive.
2. A *separately frozen, fresh 24-task audit* compares full correction with a support-offset control that cannot improve ordering within the supported set. This isolates the incremental decision effect of varying the correction among supported candidates, conditional on the fixed support rule.
3. An explicit support-constrained oracle separates the cost of restricting which plans can be selected from excess error in selecting among the attainable plans. Sealed candidate-level records and a second arithmetic implementation make these diagnostics inspectable without training a model.

The diagnostic identities follow elementary finite-set algebra. Both studies rescore fixed candidate pools, so every scoring rule faces the same choices. The empirical contribution is a controlled case study of how remembered cost errors affect selection, together with diagnostics that can be applied to other fixed-pool scorers. Table 1 summarizes the two designs.

2 Related Work

Visual prediction and planning. DINO-WM [1] predicts frozen DINOv2 patch features [2] conditioned on actions and plans toward visual goals. We train two local models with this architecture and hold them fixed throughout the correction studies. Their limited training budget is described in Section 3. Work on physical planning with joint-embedding world models examines conditions that support effective planning [4]. We examine the subsequent step of using stored experience to adjust a frozen model's scores.

Search and episodic experience. CEM fits a proposal distribution to elite samples [5]. iCEM improves sampling efficiency with temporally correlated action noise and reuse of elite samples [6]. IMWM retrieves inverse experience for initialization and combines learned inverse compatibility with a reliability gate [3]. Neural Episodic Control [7] retrieves learned action values associated with past outcomes. Our memory stores predicted and executed endpoints and transfers their *goal-dependent cost difference*. The shared-offset control ablates variation in that transferred error. Evaluation of this correction inside iCEM or IMWM remains future work.

Prediction errors that matter for control. Value-aware model learning [8] and objective-mismatch studies [9] establish that conventional predictive fit need not track control quality. PETS [10] addresses model uncertainty through probabilistic ensembles and trajectory sampling. Latent-state dynamics residuals can adapt a world model [11]. Here, the model and latent transitions stay fixed; memory acts on the terminal goal score. We measure how that intervention changes the costs of the selected plans.

*The author works at Meta on Instagram AI Search. This research is not affiliated with Meta and does not represent Meta's views.

Recent diagnostic and transfer work. Vakalis [12] directly audits latent intervention fidelity and reports support-dependent error-ranking failures. VERDI [13] validates retrieved optimization strategies on a target world model. Its unit of reuse is an optimization strategy; ours is a candidate plan's cost error. Loss-conditioned state execution [14] distinguishes beneficial feasible updates from the benefit of a particular proposal. Our attainable-set calculation concerns which candidate plans a fixed support mask permits a scorer to select; it uses executed outcomes to measure the limits of a fixed correction mask.

3 Setting and Error Memory

3.1 Fixed Models and Candidate Pools

Two action-conditioned world models use frozen visual patch features and the same 512 aligned training episodes, 64 validation episodes and 4,000-update budget. Validation nine-step feature MSE selects updates 3,500 and 4,000. Both models are kept, independently of planning outcomes. Inputs include starting image, initial position and candidate actions. Nine autoregressive model steps each group five two-dimensional raw actions, yielding a 45-step physical horizon. Training-only statistics normalize actions and position; raw action norm is at most 1.8. Physical success is final goal distance strictly below 4.5 simulator units.

For candidate j, the base score b_j is terminal predicted-feature MSE to the goal, averaged over 196 patches and 384 channels. The evaluation cost c_j uses the executed terminal image in the same feature space. These are distinct from physical goal distance. All planning is open-loop.

Each query has four uncorrected CEM streams: two models crossed with cold or visually retrieved initialization. Each stream uses population 16, four elites, 12 iterations, initial standard deviation 1 and floor 0.1. Mean and incumbent occupy two population slots; including the initial score gives 193 slots per stream and 772 per task. Paired random draws are retained. All proposals are merged by exact float32 action identity; provenance is preserved. The common pool therefore costs *four searches*. Its generation uses predicted scores alone and retains every proposal.

For a nonempty finite pool and any finite score vector s, let $j(s)$ be its first minimum in fixed pool order. Selected-feature regret is

$$R(s) = c_{j(s)} - \min_j c_j. \tag{1}$$

Executed outcomes enter evaluation after selection. We also measure whether any candidate succeeds physically and whether exact executed-feature scoring chooses physical success. Both diagnostics require privileged access to executed outcomes.

3.2 Matched Collection and Retrieval

The first study collects two disjoint banks of 16 new start/goal contexts each. Each source supplies eight entries per context and model: 128 entries per bank, but only 16 independent starting contexts. *Recorded* memory uses the eight nearest start/goal image pairs in a 256-episode training-only proposal library and re-executes their actions from the new collection start. *Optimized* memory initializes CEM with the nearest recorded plan, scores 97 slots over six iterations, and retains the eight distinct lowest-model-cost actions. Realized cost and success never select entries. Both sources therefore have fresh measurements at the same collection contexts and equal capacity, but not necessarily equal diversity or predicted-score distributions.

An entry stores starting features and position, actions, and predicted and executed terminal features, $\hat{z}_i$ and z_i. For a query goal z_g, its cost error is

$$e_i(z_g) = \mathrm{MSE}(z_i, z_g) - \mathrm{MSE}(\hat{z}_i, z_g). \tag{2}$$

Neighbor distance sums scaled RMS distances in starting visual features, normalized initial position, and normalized actions. Four nearest entries get normalized exponential distance weights w_{ji}. Scales and a 95th-percentile leave-one-out nearest-distance threshold are fixed from the preceding training bank and remain unchanged for the new collections and queries. Let m_j indicate that candidate j passes this support threshold. The correction is

$$d_j = m_j g_j, \qquad g_j = \sum_{i \in N(j)} w_{ji} e_i(z_g), \quad s_j = b_j + \alpha d_j. \tag{3}$$

Neighbor identities are goal-independent; the retrieved error values depend on the goal. Support is determined by a heuristic distance threshold. Scores may be negative. Query endpoints are excluded from memory. A fixed error permutation preserves keys and support while breaking their correspondence with stored errors. We use this as a descriptive shuffle control. Appendix E specifies the exact metric, fitted constants, weighting kernel, normalization and tie rules. Large errors under a collection goal need not remain large or useful under a different query goal.

4 Decision-Focused Audit

4.1 Separate a Support Offset From Error Variation

An identical shift of *all* scores cannot change an ordering. Shifting only supported scores can change which plan wins, even if ordering within that set does not improve. To distinguish these effects, let $S = \{j : m_j = 1\}$ and $\mu_S = |S|^{-1} \sum_{j \in S} g_j$, or zero if S is empty. Decompose

$$u_j = m_j \mu_S, \qquad v_j = m_j (g_j - \mu_S), \qquad d_j = u_j + v_j. \tag{4}$$

Four selectors use b, $b+u$, $b+v$, and $b+d$. We call these uncorrected, support offset, within-support, and full correction. Strength is fixed at $\alpha = 1$; the offset uses the same retrieved errors and work as the full correction. Its pool mean depends on the candidate set. The primary comparison $R(b+d) - R(b+u)$ measures the incremental effect of within-support variation *in*

the presence of this offset. The nonlinear argmin couples this effect to the offset and support mask.

Proposition 1 (Correction variance)**.** *For a uniform distribution over the finite pool and $p = |S|/n$,*

$$\mathrm{Var}(d) = \mu_S^2 p(1-p) + \mathbb{E}[v^2]. \tag{5}$$

The two terms are the centered support-offset variance and the within-support variance; their covariance is zero.

The proof is in Appendix A. This exact decomposition says where score variation comes from. Accuracy requires comparison with executed costs. For any score residual $r = c - s$, $\mathbb{E}[r^2] = \mathbb{E}[r]^2 + \mathrm{Var}(r)$. Thus, lower RMSE can be driven by a removable common bias. Lower centered error also does not guarantee lower regret for a particular pool; errors near the selected minimum matter.

4.2 Limits Imposed by Support

Consider any finite correction that is zero outside S, allowing arbitrary values inside it. Unsupported scores retain their ordering. Define

$$A_S = S \cup \{j_U\}, \quad j_U = \arg\min_{j \notin S} b_j, \tag{6}$$

omitting j_U if every candidate is supported and using the same first-minimum tie rule.

Proposition 2 (Support-constrained attainable set)**.** *A_S is exactly the set of candidates selectable by arbitrary corrections restricted to S. Consequently, for every such score vector,*

$$\begin{aligned} R(s) &= R_{\text{restriction}} + R_{\text{excess}}(s), \\ R_{\text{restriction}} &= \min_{j \in A_S} c_j - \min_j c_j \ \geq 0, \\ R_{\text{excess}}(s) &= c_{j(s)} - \min_{j \in A_S} c_j \ \geq 0. \end{aligned} \tag{7}$$

This oracle allows arbitrary supported corrections, a broader class than our weighted rule, and uses privileged executed outcomes. A positive restriction term means that improving error estimates *without changing the mask* cannot recover the pool's best executed-feature plan. Zero restriction means that this plan is attainable; the scorer must still select it. Physical success attainability is $\mathbf{1}\{\exists j \in A_S : \mathrm{dist}_j < 4.5\}$. We also record whether the feature-cost oracle actually selects a physically successful plan.

5 Experiments and Analysis

Study 1: memory source. The frozen source comparison uses 32 collection, eight development and 24 evaluation tasks from an opposite-side endpoint sampler. Each model/bank/source chooses $\alpha \in \{0, .25, .5, 1\}$ by minimum mean selected executed-feature cost on development tasks, with lower-strength ties. Shuffled controls are separately calibrated. Evaluation tasks do not choose strengths. The primary contrast is optimized-minus-recorded regret, averaged over four model/bank cells within a task, followed by a 20,000-resample percentile task bootstrap. The predeclared escalation gate requires optimized memory to improve regret over recorded, uncorrected, and shuffled scoring on both models without fewer successful selections. This gate fails, ending the planned escalation to corrected in-loop search.

Study 2: prospective component audit. The source study motivates a separate component comparison. Before scoring, we commit 24 fresh tasks (Q01–Q24), the component protocol, source snapshots and hashes. We reuse both models, both banks for each source, the same threshold and proposal generator. No new training, collection, key learning, support calibration, strength sweep or shuffle selection occurs. All four components have fixed $\alpha = 1$; the same components are computed for the existing shuffles. The primary contrast averages full-minus-support-offset regret over eight model/bank/source cells per task. The *24 task means* are the bootstrap units, with 20,000 draws and seed 2026092417. We report all task signs and descriptive model/source breakdowns. The interval conditions on the two fixed models and banks; it excludes uncertainty from retraining, memory collection and recalibration.

Integrity and scope. For both studies, exact endpoint/start overlaps with earlier tasks are rejected by stopping the run. Reference actions establish reachability but are never candidates or memory entries. All tasks remain in analysis regardless of success availability. Saved action traces have full-pool state replay; selected component plans and oracle choices also have ordinary Gym replay. Cached model scoring is compared with ordinary scoring. A portable checker verifies seals, complete task/cell grids, action bounds, retrieval arithmetic and summaries; a second direct implementation reconstructs the new primary contrast and attainable set. These local consistency checks cover the saved records and analysis code. Each study retains its own statistical analysis.

Table 1. Two questions, two disjoint evaluation samples. Models and collection banks are shared; evaluation tasks and primary contrasts are not. Repeated cells are averaged within a task before resampling.

Design choice	Study 1: memory source	Study 2: correction components
Intervention	Optimizer-selected vs. recorded memory	Full correction vs. common support offset
New endpoint tasks	32 collection, 8 development, 24 evaluation	24 fresh evaluation tasks
Models and banks	2 models; 2 banks per source/model	Same models and banks; no new collection
Memory capacity	128 entries from 16 starts per bank	Same capacity and retrieval geometry
Correction strength	Selected on development tasks	Fixed at $\alpha = 1$ before scoring
Repeated evaluation cells	96 per scoring method	192 per scoring method
Primary uncertainty unit	24 task means, 4 cells per mean	24 task means, 8 cells per mean

6 Results

6.1 Study 1: Memory Source

Optimizer-selected memory contains larger cost errors under its collection goals. Mean optimism (executed minus predicted cost) rises from 0.0896 to 0.2434 for model 1 and from 0.0755 to 0.1793 for model 2. Actual collection costs also decrease, with a larger decrease in predicted costs. Optimizer-selected plans therefore expose greater model optimism despite their better executed costs.

On the 24 held-out tasks, calibrated recorded and optimized memories have mean regret 0.24713 and 0.24878, versus 0.24711 uncorrected. The primary optimized-minus-recorded contrast is +0.00165, with 95% task-bootstrap interval $[-0.00271, +0.00622]$. Seven task means favor optimized memory, eight favor recorded memory, and nine tie. The interval leaves benefit and harm unresolved. All calibrated methods retain the same physical-success sets: model 1 succeeds on no task and model 2 on the same two tasks in each bank, or 4/96 repeated choices. Six pools contain physical success; exact executed- feature scoring selects it in all six. Eighteen pools offer no successful plan.

The predeclared $\alpha = 1$ sensitivity reduces average within-task score RMSE for both sources, yet worsens its centered component. For optimized memory, model 1's uncentered/centered RMSE changes from 0.2262/0.1520 to 0.2205/0.1693; model 2 changes from 0.1623/0.1046 to 0.1549/0.1150. This descriptive observation motivates the fresh component test. Study 1's primary test remains inconclusive. Support covers about 30% of optimized-memory queries versus 55.5% for recorded memory, and optimized lookups usually retrieve four entries from one collection start. These patterns suggest several possible limits on transfer, whose individual effects remain unresolved.

6.2 Two Measured Plan Choices

Figure 1 follows two competing action sequences in each of two navigation problems. A shared correction gives every eligible plan the same adjustment and selects Plan A. Allowing a separate retrieved adjustment for each plan changes the selection to Plan B. We execute both plans to measure the cost of their final images relative to the goal image.

In the first example, the changed choice lowers executed feature cost from 0.750 to 0.312. In the second, it raises the cost from 0.830 to 0.864. These examples show what a change in ordering means for selection; the following section reports the comparison across all 24 tasks.

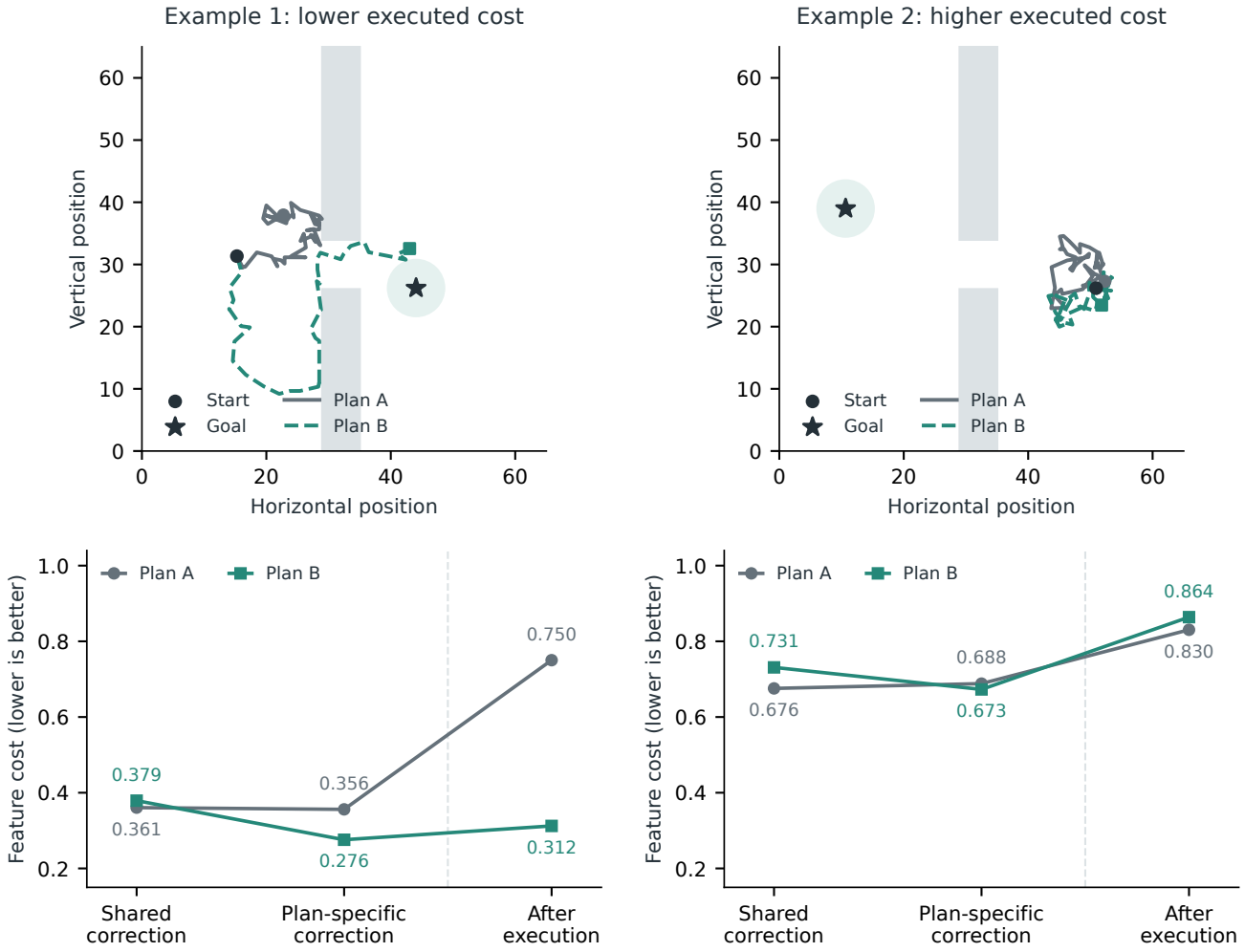

Figure 1. Two measured changes in plan choice. Top: executed paths for each example's start and goal. Bottom: estimated costs under the shared and plan-specific corrections, followed by measured costs after execution. Each rule chooses its lowest-scoring plan: A under the shared correction, B under the plan-specific correction. Lower feature cost is better; none of these four plans reaches the physical-success threshold. The cases were chosen post-hoc to illustrate both effect directions; Appendix B gives their identifiers and selection rule.

6.3 Study 2: Prospective Component Comparison

All 24 tasks complete with 690 distinct plans each, or 16,560 executed candidates. Full correction has mean regret 0.24163 versus 0.25714 for the common support offset. The prespecified paired difference is −0.01552, with 95% task-bootstrap interval [−0.03519, −0.00026]. Full correction changes the offset's choice in 32/192 cells. The 24 task means include 11 improvements, 6 regressions and 7 zeros (Figure 2A).

The aggregate interval lies only just below zero. Q03 and Q05 have much larger improvements than most tasks, and all four model/source-specific intervals include zero despite negative point estimates (Table 3 and Figure 3). A post-hoc leave-one-task-out influence check retains negative point estimates throughout, with four of its 24 intervals including zero. The full-sample estimate remains primary; Appendix B reports the influence check.

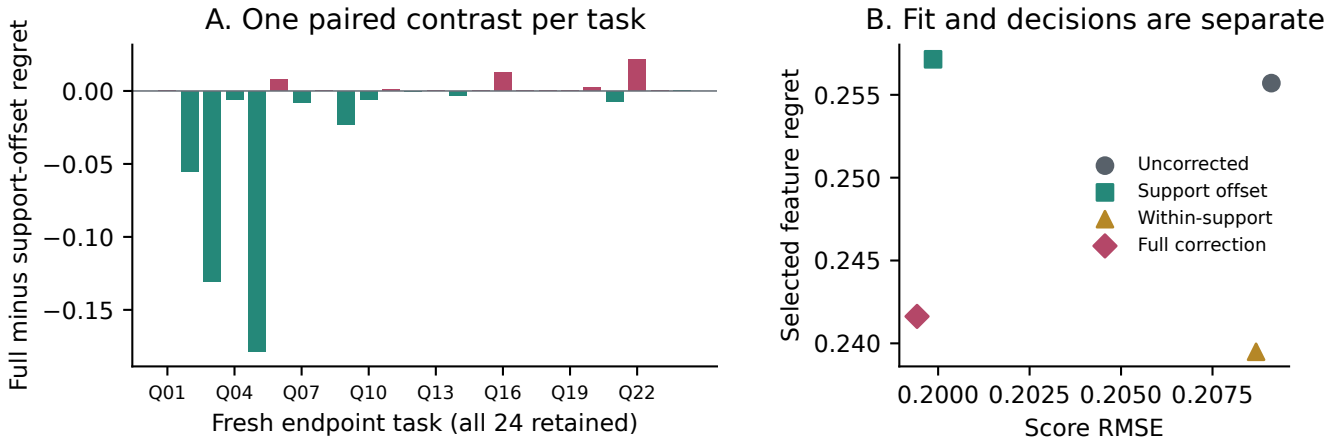

Figure 2. Correction effects and predictive fit. (A) All 24 paired task means, each averaging eight fixed model/bank/source cells; negative values favor full correction over the support-offset control. (B) Aggregate fit and selected-feature regret for the four matched components. Panel B is descriptive; RMSE is computed after averaging cell MSEs.

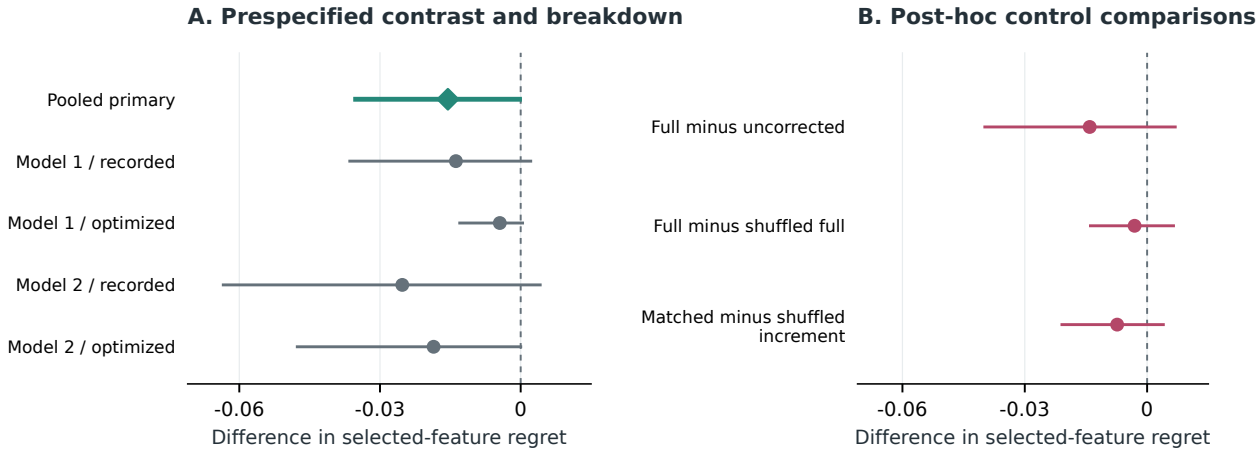

Figure 3. Uncertainty in the primary and secondary comparisons. (A) The pooled primary contrast and four descriptive model/source breakdowns. (B) Separately labeled post-hoc controls. Lines are 95% task-bootstrap intervals; zero is the dashed reference. Secondary intervals are descriptive and unadjusted for multiple comparisons. The pooled primary interval barely excludes zero.

6.4 Controls and Detailed Estimates

The incremental component effect depends on the comparator. Uncorrected and shuffled-full regret are 0.25571 and 0.24470, respectively. Post-hoc descriptive task intervals for full minus uncorrected and full minus shuffled scoring are $[-0.03979, +0.00689]$ and $[-0.01389, +0.00646]$; both include zero. Comparing the matched and shuffled full-minus-offset increments also includes zero. These post-hoc intervals are descriptive and unadjusted for multiple comparisons. Appendix D shows all 192 cell-level contrasts, including ties and regressions, without outcome-based reordering.

Prediction error and selected cost. The support offset lowers score RMSE from 0.2091 to 0.1999 and slightly raises mean regret from 0.25571 to 0.25714. Adding within-support variation barely changes RMSE further, to 0.1994, while giving the negative primary regret contrast. Within-support-only scoring has RMSE 0.2087 and the lowest descriptive regret, 0.23949. The methods thus have different rankings under prediction error and selected cost. Centered RMSE gives another example: full correction's 0.1539 exceeds uncorrected scoring's 0.1513 despite its lower regret point estimate. The support offset accounts for 74.8% of centered correction variance.

Physical outcomes remain modest. Uncorrected, support-offset, within-support-only and full scoring succeed in 20, 19, 22 and 21 of 192 repeated cells, respectively; shuffled full scoring also has 21 successes. Mean physical distance is slightly worse for full than uncorrected scoring, 24.00 versus 23.77. The physical outcomes provide little evidence of a navigation benefit. Table 2 reports every matched and shuffled component under the original analysis plan.

Table 2. Fresh prospective component audit. All selectors score the same candidate pool. The 192 selections comprise eight repeated cells for each of 24 tasks. Uncorrected scores repeat across sources and banks. RMSE takes the square root after averaging cell MSEs. Shuffles retain fixed keys, support, and error permutations.

Errors	Score component	Regret ↓	RMSE ↓	Centered RMSE ↓	Distance ↓	Success
Matched	Uncorrected	0.25571	0.2091	0.1513	23.77	20/192
Matched	Support offset	0.25714	0.1999	0.1545	24.64	19/192
Matched	Within-support	0.23949	0.2087	0.1507	23.37	22/192
Matched	Full correction	0.24163	0.1994	0.1539	24.00	21/192
Shuffled	Support offset	0.25287	0.1944	0.1530	24.16	19/192
Shuffled	Within-support	0.24773	0.2105	0.1532	23.92	21/192
Shuffled	Full correction	0.24470	0.1958	0.1548	24.05	21/192

Table 3. Descriptive model/source breakdown on the same 24 tasks. Task-bootstrap intervals are unadjusted for multiple comparisons. Success attainability counts cells whose attainable set contains a physically successful plan, using privileged executed outcomes. Each row repeats two banks per task.

Model	Memory	Full minus offset [95% interval]	Supported	Restriction regret	Success attainable
1	Recorded	-0.01381 [-0.03643, +0.00214]	60.9%	0.06034	18/48
1	Optimized	-0.00445 [-0.01299, +0.00040]	31.4%	0.14787	12/48
2	Recorded	-0.02525 [-0.06341, +0.00416]	60.9%	0.06825	16/48
2	Optimized	-0.01857 [-0.04763, +0.00001]	31.6%	0.14634	10/48

6.5 Support Restriction and Selection Headroom

Mean support coverage is 46.2%, varying by source: about 60.9% for recorded memory and 31.4–31.6% for optimized memory. Across the 192 repeated cells, full correction's mean regret separates into 0.10570 restriction cost and 0.13592 excess selection cost. The restriction accounts for 43.7% of the total. Optimized-memory restriction regret is about 0.147 for both models, compared with 0.0603/0.0682 for recorded memory (Figure 4A). These estimates describe the realized masks and pools. The effect of changing the threshold remains untested.

Physical success occurs in 10/24 pools, giving 80 success-containing repeated cells. The support-constrained attainable set contains success in 56/192 cells; it therefore blocks success in 24 of those 80 otherwise eligible cells. Exact support-oracle feature selection also succeeds in 56/192, while full correction succeeds in only 21/192 (Figure 4B). Hence both support restriction and selection within the attainable set leave headroom. The oracle calculations use executed query outcomes. Figure 6 makes the task-level availability and repeated-cell selection counts explicit.

The unrestricted executed-feature oracle succeeds on 9/24 tasks out of ten success-containing pools. On Q15, the plan minimizing executed feature cost misses the physical-success threshold. This establishes a mismatch between the two objectives in this pool. The remaining fourteen pools contain no physical success under any scorer. Figure 5 shows the Q15 trajectories and every candidate's feature cost versus physical distance. The 10^{-8} unsupported near-tie sensitivity adds no support-attainable success cells. Appendix B reports degeneracy, numerical and post-hoc sensitivity checks.

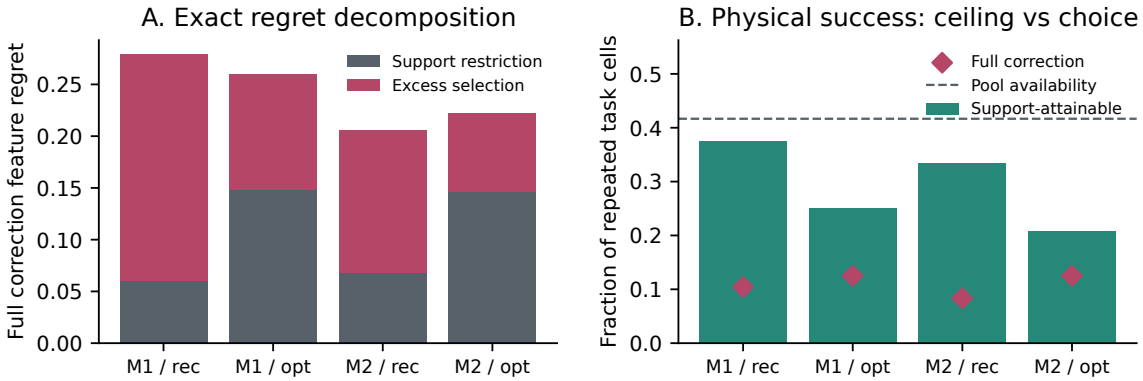

Figure 4. Support restriction and selection headroom. (A) Full-correction regret separates exactly into restriction cost and excess selection cost. (B) Bars show whether any physically successful plan is attainable by an arbitrary correction with this support mask; diamonds show full correction's realized success. The dashed line is success availability in the whole pool. Oracles use executed outcomes unavailable to a planner.

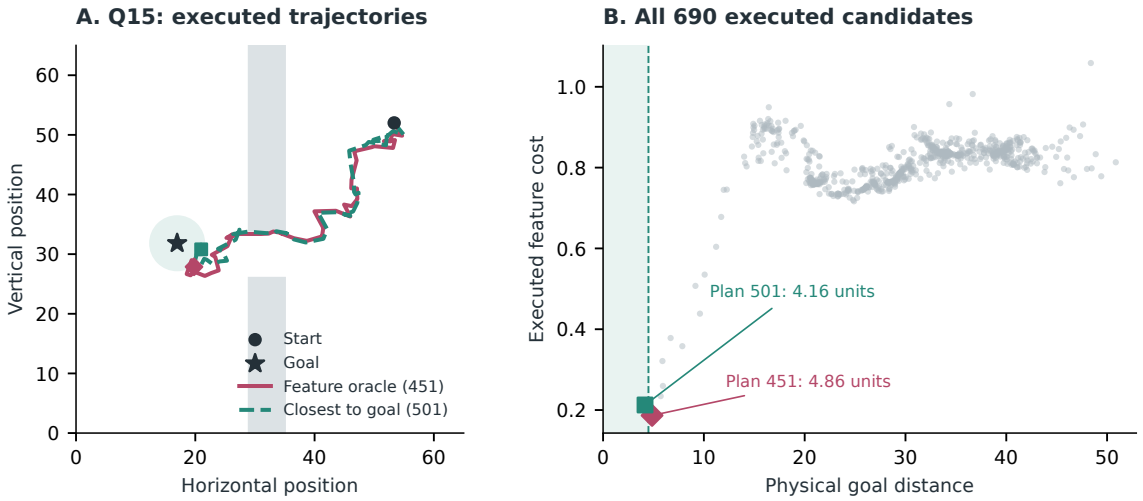

Figure 5. A feature-space optimum can miss the physical goal. Post-hoc illustration of Q15, the sole success-containing pool where the executed-feature oracle fails physically. (A) Saved 45-step trajectories for the minimum-feature-cost plan and the plan closest to the goal. The shaded goal disk has radius 4.5 simulator units. (B) All 690 candidates, with the physical-success region shaded. Both selections use executed outcomes; this case illustrates the distinction between the two objectives.

7 Discussion and Limitations

Locating the error. The audit measures candidate availability, support restrictions and the cost of the selected plan separately. The first two diagnostics use executed outcomes to locate potential improvements. The component comparison then tests whether transferred errors help the scorer realize an improvement using information available at decision time.

Large restriction regret motivates examining the support mask. Large excess regret motivates examining selection within that mask, including the retrieval keys, memory diversity and correction estimator. These possible explanations require controlled follow-up experiments with fresh development and evaluation data. Enlarging support lowers or preserves the oracle restriction term. Its practical effect also depends on the reliability of the newly admitted corrections, so removing the filter could worsen selection.

Interpreting the component comparison. Both the mask and common offset depend on retrieval: the mask uses action similarity and the offset averages retrieved errors. Full minus offset measures the incremental effect of within-support variation conditional on that mask and offset. Its impact can involve supported versus unsupported competition as well as ordering within the supported set. Argmin interactions mean that the within-support-only arm need not predict that incremental effect. The pool-dependent offset was designed for this ablation. Deploying it in a streaming planner would require a separate design. Shuffling disrupts the error-to-key association while leaving possible distributional and diversity differences between memory sources.

Generalization and inference. The results concern one arena, one endpoint sampler, two modest local models sharing a training split, two realized banks per source/model, one distance geometry, and a fixed CEM budget. Entry count exaggerates independent context coverage: 128 entries come from 16 starts per bank. Repeated model, source and bank cells share each task, which is the unit of inference. The task-bootstrap intervals omit uncertainty from retraining, recollection and calibration; with 24 task units they can be imprecise. Intervals crossing zero leave both benefit and harm unresolved. Each study uses a separate task sample and primary question. All tasks remain in the analysis, including pools without physical success.

Extension to planning. The candidate generator is fixed throughout this experiment. In adaptive CEM or receding-horizon control, corrected scores would affect later proposals, requiring an in-loop evaluation. Comparisons with iCEM and full IMWM, stronger models, additional environments and runtime measurements remain open. Compute accounting would need to include all four proposal streams and the different memory-collection costs. Physical success and executed-feature regret also require separate evaluation. The support oracle permits arbitrary corrections and knows the executed outcomes, giving an optimistic bound on what a practical estimator with this bank could achieve. These results apply to the tested recipe and fixed inputs.

8 Conclusion

Memory collected from optimizer-selected actions contains larger errors, with inconclusive effects on held-out selection. The subsequent component experiment finds a small, borderline improvement from plan-specific correction over a shared support offset. Comparisons with uncorrected and shuffled scoring remain unresolved, and physical success changes little. The support mask excludes useful alternatives and accounts for 43.7% of full correction's feature regret. Together, these observations motivate evaluating memory correction at three levels: the plans available, the plans the support rule permits, and the plan the scorer selects.

Reproducibility and Assistance

The saved-score evidence bundle is available on Zenodo at doi:10.5281/zenodo.22948935 (version 1.0.0). It contains both frozen protocols, candidate-level scores and outcomes, retrieval records, calibration inputs, model identities and a NumPy-only reproduction command. It recomputes both studies' selections and primary intervals, the collection check, support diagnostics and post-hoc controls. Appendix E gives the exact retrieval recipe and the bundle's verification boundary. The extracted bundle has been tested outside the repository without Torch or model weights. Analysis code is licensed under MIT; data and documentation use CC BY 4.0.

The original author-held artifact additionally preserves proposal-slot provenance, source snapshots and replay checks. Protocol commits predate scoring in its private history; there was no independent public preregistration. The portable bundle checks recorded-score arithmetic and does not regenerate images, neural predictions, neighbor searches or physical trajectories. Those steps require additional model, data and feature-cache assets. Independent scientific replication remains outstanding.

OpenAI Codex assisted with implementation, experimental checks, analysis, visualization, literature lookup and manuscript drafting.

References

[1] Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World Models on Pre-trained Visual Features enable Zero-shot Planning. *arXiv:2411.04983*, 2024. https://arxiv.org/abs/2411.04983.

[2] Maxime Oquab et al. DINOv2: Learning Robust Visual Features without Supervision. *arXiv:2304.07193*, 2023. https://arxiv.org/abs/2304.07193.

[3] Baoqi Gao, Ruize Han, Miao Wang, and Song Wang. IMWM: Intuition Models Complement World Models for Latent Planning. *arXiv:2606.01626*, 2026. https://arxiv.org/abs/2606.01626.

[4] Basile Terver, Tsung-Yen Yang, Jean Ponce, Adrien Bardes, and Yann LeCun. What Drives Success in Physical Planning with Joint-Embedding Predictive World Models? *arXiv:2512.24497*, 2025; revised September 2026. https://arxiv.org/abs/2512.24497.

[5] Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A Tutorial on the Cross-Entropy Method. *Annals of Operations Research*, 134:19–67, 2005. https://doi.org/10.1007/s10479-005-5724-z.

[6] Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient Cross-Entropy Method for Real-time Planning. *arXiv:2008.06389*, 2020. https://arxiv.org/abs/2008.06389.

[7] Alexander Pritzel, Benigno Uria, Sriram Srinivasan, Adrià Puigdomènech Badia, Oriol Vinyals, Demis Hassabis, Daan Wierstra, and Charles Blundell. Neural Episodic Control. In *ICML*, PMLR 70:2827–2836, 2017. https://proceedings.mlr.press/v70/pritzel17a.html.

[8] Amir-Massoud Farahmand, André Barreto, and Daniel Nikovski. Value-Aware Loss Function for Model-based Reinforcement Learning. In *AISTATS*, PMLR 54:1486–1494, 2017. https://proceedings.mlr.press/v54/farahmand17a.html.

[9] Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective Mismatch in Model-based Reinforcement Learning. In *Learning for Dynamics and Control*, PMLR 120:761–770, 2020. https://proceedings.mlr.press/v120/lambert20a.html.

[10] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models. In *NeurIPS*, 2018. https://arxiv.org/abs/1805.12114.

[11] JB Lanier, Kyungmin Kim, Armin Karamzade, Yifei Liu, Ankita Sinha, Kat He, Davide Corsi, and Roy Fox. Adapting World Models with Latent-State Dynamics Residuals. *arXiv:2504.02252*, 2025. https://arxiv.org/abs/2504.02252.

[12] Donna Vakalis. The Intervention Gap in Latent World Models. *arXiv:2608.29998*, 2026. https://arxiv.org/abs/2608.29998.

[13] Junyu Wu, Shiqin Nie, Youyi Kou, Baohua Yin, Guocai Yao, Qingyu Chen, Jingheng Ma, Shiji Zhou, Hongyong Song, Mingchen Zhuge, Sen Cui, and Changshui Zhang. VERDI: Retrieval Is Not Transfer for Continual World Model Optimization. *arXiv:2608.09537*, 2026. https://arxiv.org/abs/2608.09537.

[14] Jintao Xu, Zhengyu Chen, Ben Zhang, Yongzhi Qi, and Jianshen Zhang. When Should a World Model Move? Loss-Conditioned State Execution. *arXiv:2609.15801*, 2026. https://arxiv.org/abs/2609.15801.

A Elementary Diagnostic Identities

Proof of Proposition 1. All expectations are uniform finite-pool averages. Since v is zero outside S and sums to zero inside it, $\mathbb{E}[v] = 0$ and $\mathbb{E}[mv] = 0$. Also $\mathbb{E}[u] = p\mu_S$, so $u - \mathbb{E}[u] = \mu_S(m - p)$. The covariance of this quantity with v is $\mu_S(\mathbb{E}[mv] - p\mathbb{E}[v]) = 0$. Expanding $\mathrm{Var}(d) = \mathbb{E}[((u - \mathbb{E}[u]) + v)^2]$ yields $\mu_S^2 \mathrm{Var}(m) + \mathbb{E}[v^2] = \mu_S^2 p(1 - p) + \mathbb{E}[v^2]$. For empty support, all terms are zero; for full support, the support offset is a global shift and its centered variance vanishes. □

Proof of Proposition 2. Every unsupported score equals its base score. Therefore no unsupported candidate except the first minimum j_U among them can be selected. Every supported candidate can be selected by making its own correction sufficiently negative. When unsupported candidates exist, j_U can be selected by making all supported scores strictly larger than b_{j_U}. These constructions prove that the attainable set is exactly A_S, including empty and full support. Adding and subtracting $\min_{j \in A_S} c_j$ gives the regret decomposition; both terms are nonnegative because the chosen candidate belongs to A_S. These statements use exact real arithmetic with a fixed tie rule. Numerical near ties are audited separately. □

For nested masks $S \subseteq S'$, $A_S \subseteq A_{S'}$: the previous best unsupported candidate either becomes supported or remains the best unsupported candidate. Hence enlarging support cannot increase oracle restriction regret. It need not improve the choices of a particular correction estimator.

Centered error and selection. Let $r_j = c_j - s_j$, let $j^* = \arg\min c_j$, and let $\hat{j} = \arg\min s_j$. Then

$$c_{\hat{j}} - c_{j^*} = s_{\hat{j}} - s_{j^*} + r_{\hat{j}} - r_{j^*} \leq r_{\hat{j}} - r_{j^*} \leq \max_j r_j - \min_j r_j.$$

A constant residual shift cancels in this elementary upper bound. Mean squared centered residuals, however, do not determine which candidates have extreme errors or which one minimizes the score. Neither a smaller centered MSE nor better average pairwise agreement alone guarantees improved top-choice regret. We therefore measure selection regret directly.

Minimal outcome-free implementation. The following NumPy function expresses the component control. Its inputs are predicted costs, stored-memory corrections and a frozen support mask; executed query costs are deliberately absent. The artifact implementation additionally validates dimensions, finite values, complete task grids and numerical identities.

```
import numpy as np

def component_scores(base, delta, support):
    base, delta = np.asarray(base, float), np.asarray(delta, float)
    support = np.asarray(support, bool)
    assert np.all(delta[~support] == 0)
    mean = delta[support].mean() if support.any() else 0.0
    offset = support * mean
    return dict(base=base, offset=base + offset,
                within=base + (delta - offset), full=base + delta)
```

B Protocol and Computational Details

Illustrative case identities. Figure 1 uses the first improving and first worsening task in the original task order for model 1 (seed 20260928), recorded memory, bank 1 (artifact index 0). Example 1 is Q05, with Plan A/B at candidate indices 169/283; Example 2 is Q11, with indices 162/39. This post-hoc selection illustrates the mechanism. Aggregate estimates include every task and model/bank/source cell.

Units and task construction. Both studies use one fixed Wall arena and an opposite-side random endpoint sampler. Query outcomes are fully simulated only to measure counterfactual decisions. Reference action sequences pass the same horizon/action constraints and are withheld from all memories and query pools. New task identities are exact-disjoint within the same environment and sampler. Models share the same training data, limiting inference across model architectures. All query corrections use starting observation/position and the entire proposed action sequence. Mean and standard deviation for actions and initial position come only from the aligned training episodes.

CEM conventions. Actions are optimized in normalized units and projected to the raw norm bound. Cold initialization is zero *normalized* action, which maps to the training mean in raw units. Visual initialization retrieves the closest training start/goal pair. The common pool includes intermediate as well as final proposals from all four streams, with exact duplicate action arrays merged. Correction methods cannot change those streams; the reported outcomes measure fixed-pool selection. An operational planner would need to account for collection, encoding, retrieval, and possibly four-stream generation costs.

Study 1 calibration. Each source/model/bank is independently calibrated on eight development tasks; zero is legitimate abstention. Banks shown as 1/2 below use artifact indices 0/1. Models 1/2 use training seeds 20260928/20260929.

Model	Bank	Recorded	Optimized	Recorded shuffled	Optimized shuffled
1	1	0.25	0.25	1.00	0.25
1	2	1.00	0.00	1.00	0.00
2	1	0.00	1.00	0.25	0.25
2	2	0.50	0.00	0.00	0.00

The predeclared unscaled sensitivity gives recorded/optimized regret 0.24136/0.25277 and their shuffled counterparts 0.24982/0.23977. Each retains 4/96 repeated physical successes. Study 1 retains its calibrated primary result. The fresh component audit fixes $\alpha = 1$ before scoring and therefore requires no development set for strength selection. The inherited query runner also replays choices at the four legacy strengths as an execution-consistency check. Those additional replays neither select a strength nor enter the fresh component comparison. Study 1's reported RMSEs average per-cell roots; the prospective audit instead averages cell MSEs before taking a root, as specified in its new protocol. These RMSE summaries should not be compared as a cross-study treatment effect.

Accounting and verification. Study 1 uses 52,544 new trajectory predictions plus 256 parity checks, including collection and development. It executes 22,080 distinct query plans across development and evaluation, of which 16,560 are evaluation plans. Its local collection/query timers sum to 7,657 seconds, including checks; 131 distinct selected query plans and 64 collection entries have ordinary Gym replay. Study 2 adds 34,368 trajectory predictions for 16,560 distinct plans; its component/oracle replay covers 209 distinct task-plan choices. No new memory entries are collected. Local timing is reported in the machine-readable artifact and is specific to the local hardware. Saved-state replay uses the same transition implementation; validation against an independent simulator remains outstanding.

Study 1 seals 165 primary JSON records; Study 2 seals 76. Verification requires the exact task/model/source/bank grid, every proposed slot's provenance, unchanged frozen snapshots and task manifests, bounded raw actions, consistent actual costs across model/bank rows, and independently recomputed correction and selection arithmetic. The full neural pipeline still depends on local checkpoint and feature-cache assets. Portable verification covers the saved records. Regenerating predictions and validating feature quality require separate checks with the neural assets.

Numerical checks. Selectors use the first minimum in pool order, never physical outcomes to resolve ties. Global score shifts and centering preserve decisions mathematically; finite-precision changes in the selected index are counted explicitly. A separate 10^{-8} near-tie attainable set admits every unsupported candidate within that tolerance of the lowest unsupported base score. This sensitivity expands the diagnostic attainable set; the primary set and selector remain exact. Across 384 matched/shuffled cells, global-offset and centered-full choice changes number 0 and 0, respectively. The generous unsupported near-tie set adds physical-success attainability in 0/192 cells. There are 34/192 empty-support cells, 48/192 full-support cells, and 34/192 exactly constant matched corrections. All four matched component methods choose the same plan in 133/192 cells.

Post-hoc influence and comparison checks. After the complete frozen analysis, we inspected all 24 leave-one-task-out versions using the same percentile-bootstrap procedure. These are descriptive influence checks; all tasks remain in the primary analysis. The leave-one-out means range from -0.01714 to -0.00842; 4/24 intervals contain zero. The original 24-task estimate and interval remain the primary result.

Post-hoc paired contrast	Mean	95% task interval
Full minus uncorrected	-0.01408	[-0.03979, +0.00689]
Full minus shuffled full	-0.00307	[-0.01389, +0.00646]
Matched minus shuffled full-minus-offset increment	-0.00735	[-0.02088, +0.00397]

These descriptive intervals are unadjusted for multiple comparisons. The full-sample component comparison remains the primary test. The difference of increments contrasts the full-minus-offset effect in matched and shuffled memories; each condition retains its own outcome-free offset.

C Earlier Studies and Research Provenance

The present questions emerged from a sequence of learning-oriented studies. Their exploratory results are reported separately from the two main studies. An early data audit found a mismatch between locally stored coordinate and RGB trajectories after the initial frame, motivating aligned training. This was not a controlled causal test of labeling conventions. On a separate 32-episode prediction test, the two aligned local models average nine-step feature MSE 0.216, against 0.628 for unchanged images, 0.339 for a constant training-feature mean, and 0.571 after replacing each episode's actions with another coherent sequence while retaining its targets. The last control tests conditioning, not accuracy on the substituted actions' true outcomes. A released checkpoint has 0.043 under different exposure. These comparisons characterize the local models.

A preceding 12-task planning study (two models, two planner seeds) gives cold and visual initialization 5/48 successful runs each, versus 0/48 with random memory initialization. The visual-minus-cold interval is [-14.6,+14.6] percentage points at the predeclared 97.5% level and leaves a wide range of effects unresolved. A post-hoc sign-flip sensitivity weakens the visual-versus-random superiority interpretation. An exploratory terminal-feature residual probe improves average MSE but changes neither model's success set. A separate short-search goal-score probe has eight fresh evaluation pools, none containing physical success; it cannot establish navigation recovery. Study 1 then asks whether optimizer-selected memory fixes error transfer. Its failure motivates the prospectively frozen component audit in Study 2. No earlier task is reused as a fresh endpoint task in these studies.

All protocols, including failed hypotheses, remain in the artifact. The preceding manuscript preserves the source comparison and the chronology of the research. The fixed fresh 24-task analysis ends regardless of outcome; additional collection or learned correction would require a new development/evaluation protocol.

D All-Cell Decision Atlas

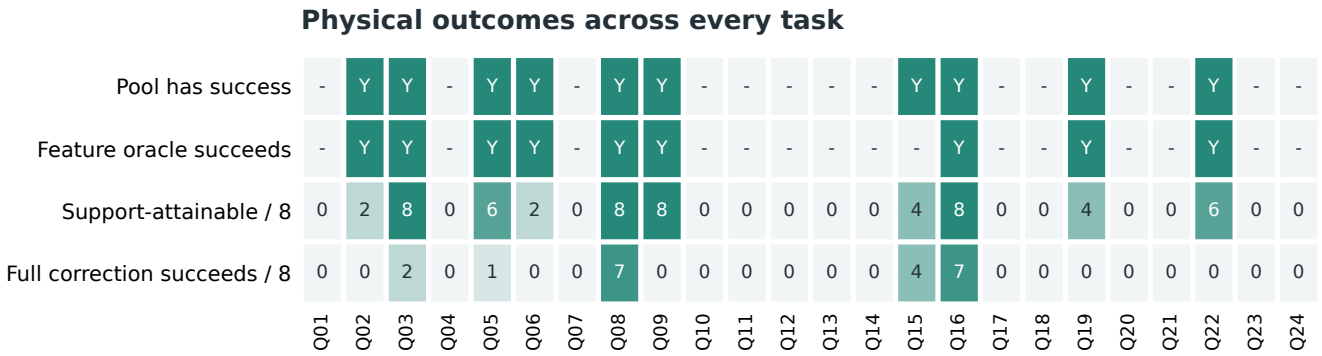

Figure 6. Availability, attainability and selection are different. All 24 tasks are retained in original order. The first two rows are binary pool-level diagnostics; the final two count successes across eight repeated model/bank/source cells per task. Darker cells indicate a larger fraction. Q15 contains a physical success, but the executed-feature oracle does not select it. Support and oracle rows use privileged outcomes.

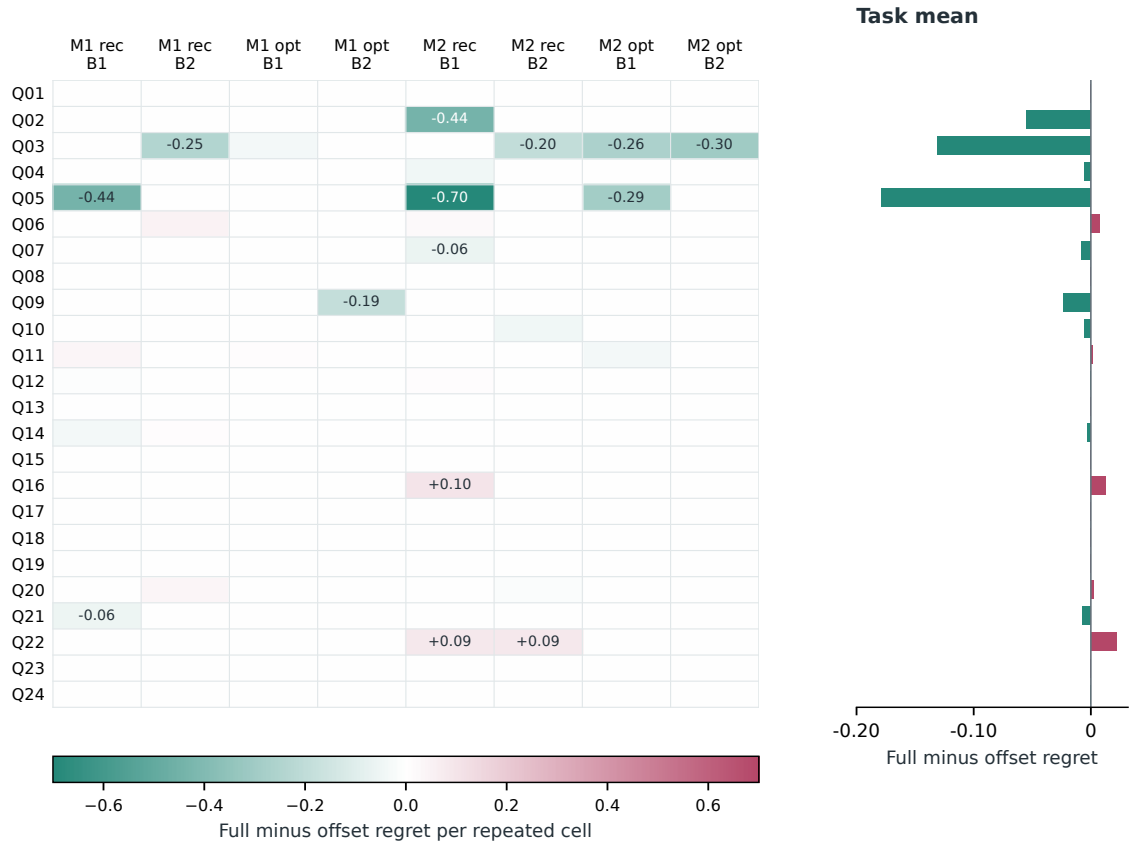

Figure 7. Every repeated-cell contrast behind the primary result. Rows follow the original Q01–Q24 task order; columns cross model (M), memory source (rec: recorded; opt: optimized) and bank (B). Teal is lower regret for full correction, rose is higher regret, and white is zero. The color scale is symmetric and uses the full observed range, with no clipping. Cell values are annotated when their absolute contrast is at least 0.05; all values enter the analysis. The right-hand bars are the eight-cell task means used in the primary bootstrap.

E Exact Retrieval Recipe and Saved-Score Reproduction

Keys and normalization. The three key components are the flattened initial DINOv2 patch features $x^{(v)} \in \mathbb{R}^{75264}$ (196×384), normalized initial position $x^{(p)} \in \mathbb{R}^2$, and the flattened normalized action sequence $x^{(a)} \in \mathbb{R}^{90}$ (45×2). Visual features are used directly; position and raw actions use elementwise $(x - \mu)/\sigma$, with training-only statistics below. These keys contain no query terminal observation or query goal. The goal enters the memory cost error in Equation (2).

Quantity	First coordinate	Second coordinate
Position mean	30.910608291625977	34.370277404785156
Position standard deviation	18.197277069091797	18.97412109375
Action mean	-0.04727219045162201	0.016181424260139465
Action standard deviation	0.7523762583732605	0.7407532334327698

Distance and frozen geometry. For component $\ell \in \{v, p, a\}$ of dimension d_ℓ, define

$$\rho_\ell(x, y) = \sqrt{\frac{1}{d_\ell} \sum_{k=1}^{d_\ell} (x_k - y_k)^2}, \qquad D(x, y) = \sum_{\ell \in \{v,p,a\}} \frac{\rho_\ell(x^{(\ell)}, y^{(\ell)})}{s_\ell}.$$

Each s_ℓ is the median of off-diagonal pairwise component distances in the preceding 256-entry training reference bank. After scaling, self-distances are excluded and the support radius r is the 0.95 quantile of that bank's leave-one-out nearest distances, using NumPy's linear quantile interpolation. Neither the new banks nor query outcomes refit these quantities. Their saved values are

$$(s_v, s_p, s_a) = (0.8479446658493253,\ 1.1118204852941576,\ 1.4060540524069816), \qquad r = 1.5181767214800028.$$

The implementation evaluates RMS distances in float64 using the squared-norm identity and clamps negative roundoff to zero before taking the square root.

Neighbors, weights and support. A stable ascending sort of D_{ji} selects four bank entries, with exact ties resolved by original entry order. Entries are ordered by collection context and then by their eight selected actions. Write $D_{j(1)}$ for the nearest distance. For the four selected neighbors $N(j)$,

$$w_{ji} = \frac{\exp[-(D_{ji} - D_{j(1)})]}{\sum_{k \in N(j)} \exp[-(D_{jk} - D_{j(1)})]}, \qquad m_j = \mathbf{1}\{D_{j(1)} \le r\}.$$

The kernel has unit temperature in the scaled distance. Subtracting the nearest distance stabilizes the exponential without changing normalized weights. The stored correction is zero when $m_j = 0$; there is no fallback correction. Memory cost errors and correction arithmetic use float64. Query model scores and executed feature costs are computed in float32 before conversion to float64. Shuffled controls use `default_rng(2026092403 + bank).permutation(128)`, with bank indices 0/1. This permutes error values while retaining the keys, neighbor weights and mask. All selectors choose the first minimum in the original candidate order.

What a reader can recompute. The companion archive, `memory_decisions_evidence.zip`, contains 56 query records: eight development and 24 evaluation pools for Study 1, and 24 fresh pools for Study 2. Each pool has all 690 candidates. It includes predicted and executed feature costs, terminal positions and physical distances, neighbor indices and weights, nearest distances, support masks, memory cost errors and matched/shuffled corrections. A separate file contains all 128 source/model/context rows of the collection check. Candidate action hashes preserve identities; full action arrays and trajectories are excluded.

With Python 3.10–3.12 and NumPy 1.26.4, `python -B reproduce.py` verifies file hashes, reconstructs corrections, recalibrates Study 1 on development data, and reproduces the original full summaries, primary task-bootstrap intervals, Tables 2–3, support diagnostics and post-hoc controls. An additional direct calculation checks Study 2's primary contrast. All selections and counts must agree exactly; floating summaries use absolute and relative tolerances of 10^{-12}. The schema and optional CSV exports make individual choices inspectable. Full-precision constants and task/model identities appear in `methods.json`.

Verification boundary and availability. The archive supports analysis of saved scores. It does not regenerate model predictions, memory feature errors, nearest-neighbor identities, candidate searches or simulator trajectories. Its hashes identify author-supplied records; they do not certify experimental authenticity or independent preregistration. Images, feature arrays, model weights and the original dataset are excluded. Version 1.0.0 is available at doi:10.5281/zenodo.22948935, with MIT-licensed analysis code and CC BY 4.0 data and documentation. Earlier exploratory studies in Appendix C are outside its scope.