TIDE: THE REACHABILITY–LEARNABILITY GAP IN TEMPORAL INTENT CONTROL

Anonymous authors
Paper under double-blind review

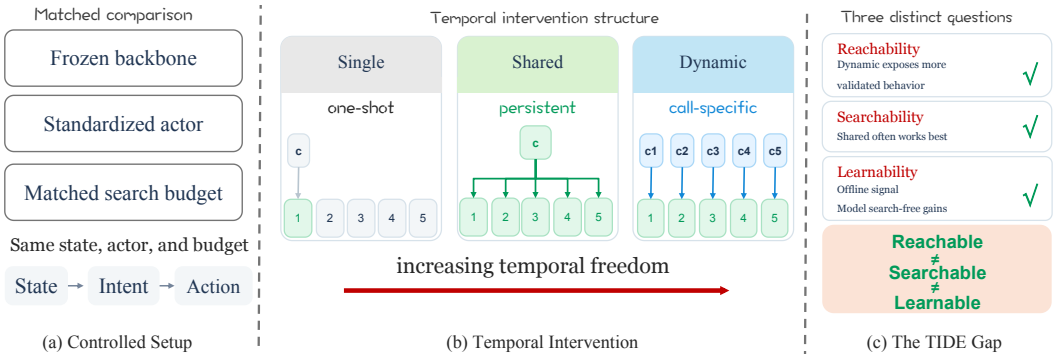

Figure 1: **Temporal freedom exposes the TIDE gap.** (a) TIDE fixes the backbone, standardized actor, and search budget. (b) SINGLE, SHARED, and DYNAMIC vary the temporal structure of bounded intent corrections. (c) Across our controlled studies, call-specific corrections expose more validated successful behavior, while persistent Shared corrections are often easier to discover under finite search and yield only modest search-free gains when amortized. Together, these results distinguish reachability, searchability, and learnability.

ABSTRACT

Visual world models support goal-directed control through learned latent representations used for direct execution or decision-time search, yet search may succeed when direct execution fails. This discrepancy raises three questions: What behaviors can a frozen system execute? Which can finite-budget search discover? And which can a shared model recover without online search? Focusing on goal-conditioned visual control with frozen, action-conditioned joint-embedding predictive world models, we introduce **TIDE**, a controlled empirical framework that equips each frozen representation with a standardized intent-conditioned execution head and varies only the temporal structure of bounded intent corrections. Specifically, we compare SINGLE one-shot corrections, SHARED corrections reused across execution calls, and DYNAMIC call-specific corrections to study how temporal intervention structure affects reachability, searchability, and learnability. In a matched intervention evaluation based on Push-T, DYNAMIC call-specific corrections uncover more validated successful behaviors. Across four control tasks (Push-T, Cube, Reacher, and TwoRoom) and two frozen backbones (LeWM and Fast-LeWM), persistent SHARED corrections are more consistently effective under matched finite search budgets. Validated SHARED interventions can also be partially amortized into search-free control, although the gains remain modest and offline learning signals alone do not reliably predict closed-loop improvement. These findings expose a control-interface gap in this class of frozen visual world models: predictive latent representations do not by themselves ensure that reachable behavior is readily searchable or learnable.

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.

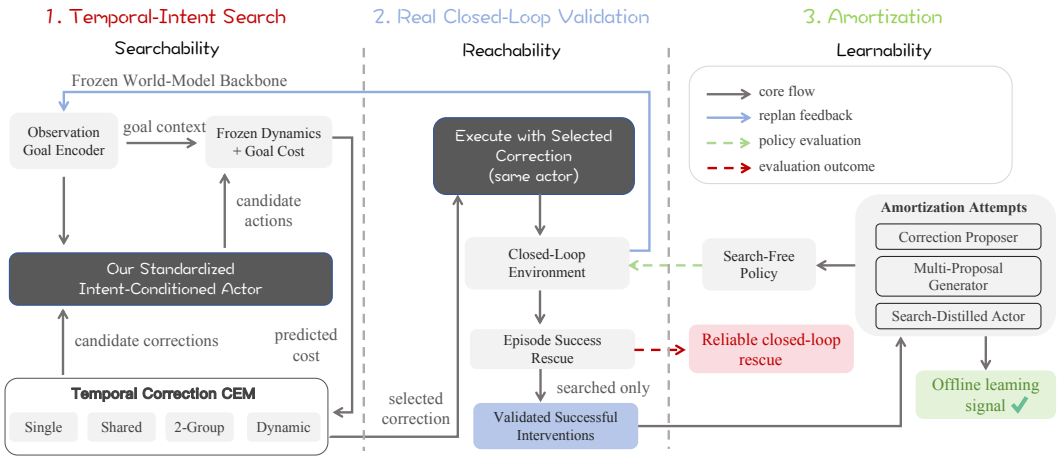

Figure 2: **TIDE framework and evaluation pipeline.** A standardized intent-conditioned execution head provides a common actionable interface over frozen world-model representations. TIDE then separates temporal-intent *searchability*, real closed-loop *reachability*, and the *learnability* of validated successful interventions.

1 Introduction

World models provide a powerful framework for visual decision making by learning predictive latent representations that support planning and control from high-dimensional observations. Recent action-conditioned joint-embedding predictive architectures offer a particularly relevant instance of this paradigm (Ha & Schmidhuber, 2018b; Hafner et al., 2019; 2020; Zhou et al., 2025a; Maes et al., 2026). Recent work has improved representation quality, predictive accuracy, and planning efficiency, making increasingly capable latent dynamics available for downstream control (Zhou et al., 2025a; Assran et al., 2025; Maes et al., 2026). Yet stronger predictive models alone do not guarantee better closed-loop behavior: information encoded by a frozen representation must still be exposed through an executable control interface before it can affect actions. This gap matters when comparing online planning, which explicitly optimizes candidate action sequences, with amortized control, which produces actions directly from a learned mapping (de Boer et al., 2005; Chua et al., 2018; Sun et al., 2026).

Consider a Push-T state in which the frozen actor fails to move the T-shaped block to its goal configuration under direct execution. A bounded correction to the actor's latent intent may nevertheless alter its predicted action blocks and produce a successful real rollout. This discrepancy raises three distinct questions: What behaviors can the frozen system execute? Which of these behaviors can finite-budget search discover? And which can a shared model recover without online search?

If any admissible correction schedule succeeds, the corresponding behavior is *reachable* through the frozen system. Whether a fixed-budget optimizer can discover such a schedule determines its *searchability*. Whether a shared model can predict a successful correction for an unseen state and execute it without online search determines its *learnability*. Thus, the existence of a successful intervention does not imply that finite search can find it, and successful search does not imply that the intervention can be amortized across states.

To study these distinctions under controlled conditions, we introduce **TIDE**, a framework for temporal intent intervention over frozen world-model representations (Fig. 2). We instantiate TIDE on four control tasks—Push-T, Cube, Reacher, and TwoRoom—using frozen LeWM and Fast-LeWM backbones, each equipped with the same standardized intent-conditioned execution-head architecture. TIDE trains and freezes the resulting representation–execution systems before varying how a bounded intent correction is distributed across execution calls.

As illustrated in Fig. 1(b), SINGLE corrects only the first call before returning to the native intent, SHARED reuses one normalized correction throughout the rollout, and DYNAMIC assigns an independent correction to each call as the state evolves. For the Push-T example, these structures respectively test whether an initial redirection, a persistent adjustment, or a sequence of call-specific adjust-

ments can recover the failed behavior. By holding the representation–execution system fixed within each comparison, TIDE evaluates validated reachability, finite-budget searchability, and search-free learnability separately. Our contributions are threefold:

- **The TIDE gap in frozen control.** We reveal a three-way gap between reachability, finite-budget searchability, and learnability: what a frozen system can execute, what finite-budget search can discover, and what shared learning can recover need not coincide.
- **A controlled framework for temporal intent intervention.** We introduce TIDE, which equips frozen world-model representations with a standardized intent-conditioned execution interface and studies bounded temporal corrections through matched search, real closed-loop validation, and search-free amortization.
- **A systematic empirical evaluation.** Across multiple tasks and frozen representations, we evaluate temporal interventions under matched budgets, paired closed-loop protocols, and independent held-out confirmation, providing empirical evidence for the separation among the three properties.

Based on TIDE, we find that reachability, searchability, and learnability need not align. Specifically, behaviors executable by a frozen system may remain difficult to discover under finite search, while successfully searched behaviors may be only partially recoverable through shared search-free control. Three main empirical findings are shown as below:

1. Temporal intent intervention exposes executable capacity. Successful closed-loop behavior can be recovered through the frozen execution system even when Direct execution fails. In the matched reachability study, call-specific DYNAMIC corrections establish a larger validated reachability lower bound than the more constrained structures (Tab. 3). Thus, Direct performance alone can underestimate the executable capacity of a fixed system.

2. Greater reachability does not imply better finite-budget searchability. Across four tasks and two frozen backbones, persistent SHARED corrections generally outperform one-shot SINGLE corrections, while the more flexible DYNAMIC structure obtains no consistent advantage over Shared under matched budgets (Fig. 3; Tab. 1). The temporal-freedom ablation exhibits the same non-monotonic pattern (Tab. 4; Fig. 4).

3. Searched interventions are partially amortizable. Across protocol-eligible task–backbone settings, standardized proposers trained from validated SHARED interventions produce positive net paired gains without online search (Tab. 2). However, the improvements remain modest and heterogeneous. Complementary mechanism analyses further show that offline learning progress alone does not reliably predict closed-loop rescue and that a state may admit multiple successful corrections.

2 RELATED WORK

World models, planning, and direct control. World models learn predictive representations for latent planning and control (Ha & Schmidhuber, 2018b;a; Hafner et al., 2019; 2020; 2021; 2023). Recent work further improves representation quality, scalability, and visual planning (Hansen et al., 2024; Zhou et al., 2025a; Assran et al., 2025; Maes et al., 2026). Model-predictive control typically selects actions online using sampling-based optimizers such as CEM, MPPI, PETS, and iCEM (de Boer et al., 2005; Williams et al., 2017; Chua et al., 2018; Pinneri et al., 2020), while policy priors and differentiable optimization can improve planning efficiency (Wang & Ba, 2020; Argenson & Dulac-Arnold, 2020; Amos & Yarats, 2020; Bharadhwaj et al., 2020; Huang et al., 2021). Direct policies instead amortize action selection into a learned mapping; INTACT, for example, maps latent goal displacement directly to actions (Sun et al., 2026). TIDE studies the gap between these modes: when planning succeeds but Direct execution fails, the behavior may be unavailable to the frozen policy, reachable but hidden by its default conditioning, or difficult to discover under a finite search budget.

Latent interventions and amortized adaptation. Joint-embedding and object-centric predictive models learn structured latent representations for prediction and control (Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025; Locatello et al., 2020; Kipf et al., 2022; Wu et al., 2023; Mosbach et al., 2025), while Causal-JEPA intervenes on object-level latents during representation learning (Nam et al., 2026). TIDE instead freezes the representation, dynamics model, and execution policy, and modifies only the temporal structure of a bounded goal-conditioned intent. By comparing

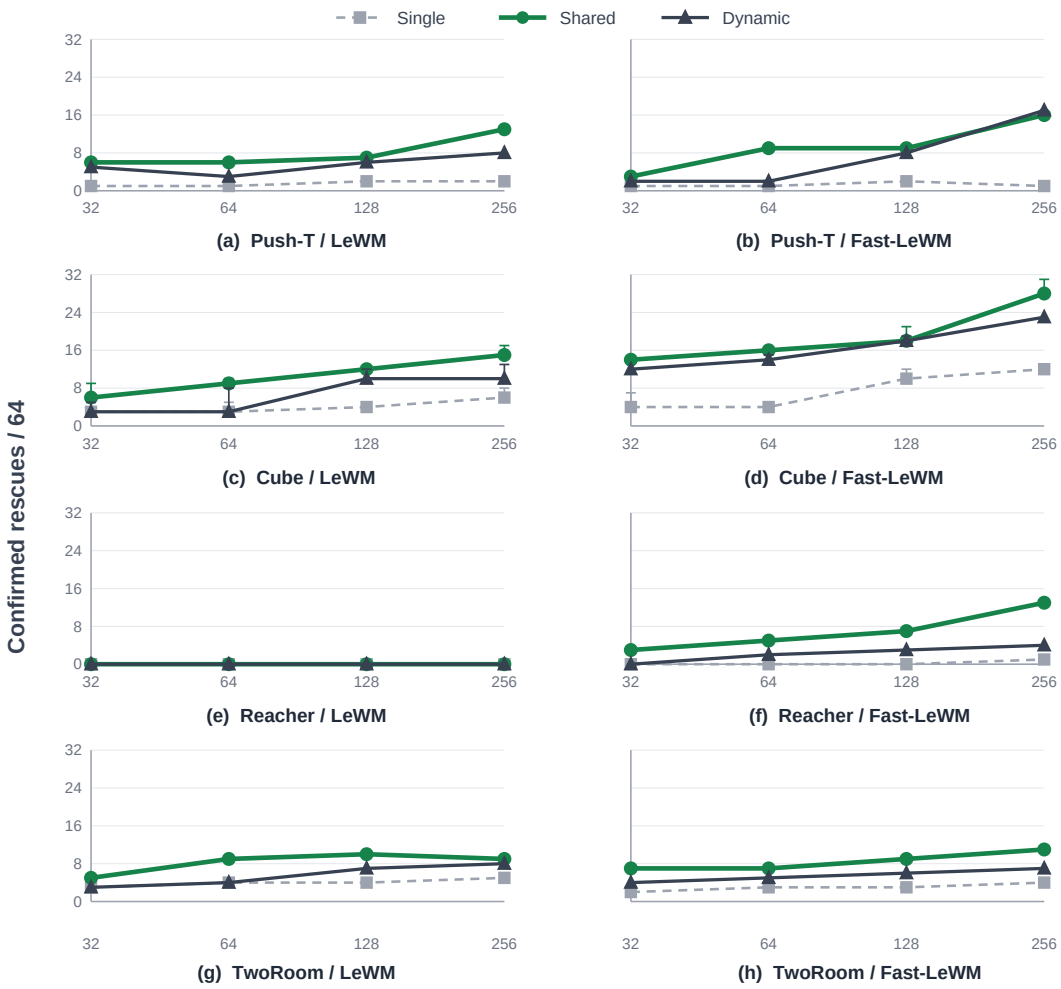

Figure 3: **Finite-budget searchability of temporal-intent parameterizations.** We compare SINGLE, SHARED, and DYNAMIC corrections on matched cohorts of 64 direct execution failures across four tasks and two frozen backbones. Markers report confirmed closed-loop rescues and vertical ranges show unresolved-aware bounds. SHARED generally outperforms SINGLE, whereas DYNAMIC yields no consistent advantage over SHARED.

SINGLE, SHARED, and call-specific DYNAMIC corrections, we examine whether additional temporal freedom exposes executable behavior, whether finite-budget optimization can discover it, and whether validated interventions can be amortized into a shared search-free predictor.

3 PROBLEM FORMULATION

Direct failure alone does not identify the source of the failure. A successful behavior may be unavailable to the frozen system, may exist but remain undiscovered under a finite search budget, or may be found only as a state-specific intervention that a shared model cannot reproduce. To formalize these possibilities, Sec. 3.1 specifies the frozen control system and the intent variable that may change. Sec. 3.2 defines reachability, searchability, and learnability, while Sec. 3.3 introduces the temporal structures used to probe them.

3.1 FROZEN WORLD-MODEL CONTROL

We consider goal-conditioned control through a frozen visual representation, latent dynamics model, and execution policy. A control instance is $x = (o^{\mathrm{cur}}, o^{\mathrm{goal}})$, where o^{cur} is the current observation

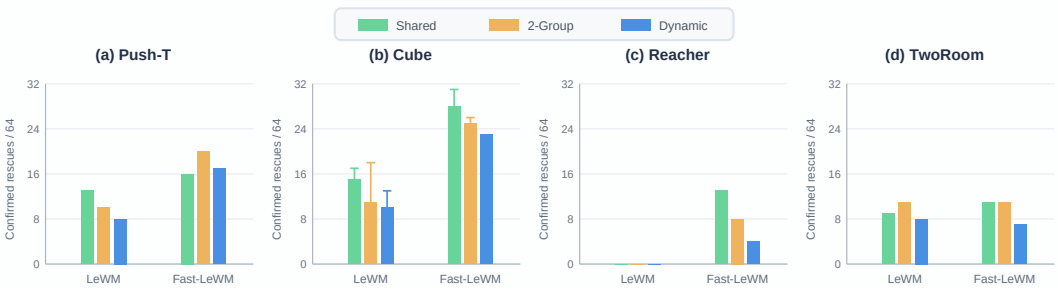

Figure 4: **Temporal-freedom ablation.** SHARED, 2-Group, and DYNAMIC receive 256 evaluations on matched 64-failure cohorts. Bars show confirmed rescues and whiskers unresolved-aware bounds; additional freedom yields no consistent search gain.

at the beginning of an execution solve and o_{goal} is the corresponding goal observation or goal specification. The frozen observation encoder E_o and goal encoder E_g map them to the current latent and goal latent:

$$z_0 = E_o(o^{\text{cur}}), \qquad z_g = E_g(o^{\text{goal}}). \tag{1}$$

Here, z_0 is the latent state at the beginning of the current execution solve, which may occur after the start of the environment episode. The solve consists of J execution calls indexed by $j = 0, \ldots, J-1$. At call j, under direct execution, the frozen system with intent rule I, execution policy π_ϕ, and latent transition F follows

$$m_j = I(z_j, z_g), \qquad u_j = \pi_\phi(z_j, m_j, c_j, j), \qquad z_{j+1} = F(z_j, u_j), \tag{2}$$

where $m_j \in \mathbb{R}^{d_m}$ is the native intent, $u_j \in \mathbb{R}^{d_u}$ is a packed action block, and c_j is the available action context. Note that E_o, E_g, I, π_ϕ, and F are fixed before intervention, and TIDE changes only the intent:

$$q_j = \Pi_\rho(s_j), \qquad \Delta m_j = \text{RMS}(m_j) q_j, \qquad m_j^\star = m_j + \Delta m_j, \qquad \text{RMS}(\Delta m_j) \le \rho \, \text{RMS}(m_j). \tag{3}$$

Here, $s_j \in \mathbb{R}^{d_m}$ is an unconstrained dimensionless variable, and q_j is its projected effective correction with the projection operator $\Pi_\rho(\cdot)$. $\rho > 0$ is the relative-RMS radius, and controls the relative magnitude of the intervention. During intervention, the policy receives the corrected intent $m_j^\star$ instead of the native intent m_j. Setting every $s_j = 0$ recovers direct execution. The exact projection and its properties are given in Appendix C.1.

3.2 Reachability, Searchability, and Learnability

Let $\mathbf{q} = (q_0, \ldots, q_{J-1})$ be an effective correction schedule, $\mathcal{Q}_M(\rho)$ the schedules admitted by temporal structure M, and $S(x; \mathbf{q}) \in \{0, 1\}$ the outcome of real closed-loop execution. We next define reachability, searchability, and learnability.

Reachability. Reachability measures whether at least one successful correction exists within the intervention class $\mathcal{Q}_M(\rho)$. A successful intervention is reachable under M if

$$R_M(x; \rho) = \mathbb{I}\left[\exists \, \mathbf{q} \in \mathcal{Q}_M(\rho) \text{ such that } S(x; \mathbf{q}) = 1\right]. \tag{4}$$

A real success certifies reachability; finite search failure does not certify unreachability.

Searchability. Searchability measures whether a specified optimizer can discover a successful correction within a fixed objective-evaluation budget. For search algorithm A, frozen predicted objective $\mathcal{J}$, and objective-evaluation budget B, let

$$\widehat{\mathbf{q}}_{A,B,M}(x) = A_B(x; \mathcal{J}, \mathcal{Q}_M(\rho)), \qquad H_{A,B,M}(x) = S(x; \widehat{\mathbf{q}}_{A,B,M}(x)). \tag{5}$$

Searchability depends on the intervention class, optimizer, objective, and finite budget, not only on the existence of a successful schedule.

Table 1: **Cross-backbone closed-loop search.** Search methods use matched 64-failure cohorts and 256 objective evaluations. Entries $r[L, U]$ report confirmed rescues with unresolved-aware bounds; bold marks the best result within each search interface.

		Direct execution	Search on 64 frozen Direct failures (256 evaluations)				
Task	**Frozen Backbone**	**SR / 750 (%)**	**Raw-CEM$_0$**	**Raw-CEM$_{\text{Direct-init}}$**	**Single**	**Shared**	**Dynamic**
Push-T	LeWM	69.33	**19**	13	2	**13**	8
	Fast-LeWM	62.53	**19**	**19**	1	16	**17**
Cube	LeWM	79.33	**17 [17,21]**	**17 [17,19]**	6 [6,8]	**15 [15,17]**	10 [10,13]
	Fast-LeWM	90.27	11 [11,12]	21 [21,22]	12	**28 [28,31]**	23
Reacher	LeWM	5.60	51	**52**	0	0	0
	Fast-LeWM	24.00	**26**	19	1	**13**	4
TwoRoom	LeWM	52.53	36	**43**	5	**9**	8
	Fast-LeWM	59.87	46	**52**	4	**11**	7

Learnability. Learnability measures whether validated searched interventions can be recovered by a shared model on episode-disjoint states without online search. Its criterion is search-free real closed-loop performance, rather than offline agreement with a searched correction alone. The Shared proposer used in our experiments is defined in Sec. 4.3.

Paired evaluation. For a frozen population of size N, let n_{ab} denote the number of state pairs with direct outcome a and intervention outcome b, where $a, b \in \{0, 1\}$ indicates failure and success, respectively. Thus, n_{01} counts states where direct fails but intervention succeeds, while n_{10} counts states where direct succeeds but intervention fails. The fixed-denominator paired change is

$$\Delta_{\text{paired}} = \frac{n_{01} - n_{10}}{N}. \tag{6}$$

Unresolved units remain in N and induce the conservative bounds in Appendix C.6.

Together, these definitions separate what the frozen system can execute, what finite-budget search can discover, and what a shared model can recover without search. A successful intervention may therefore be reachable without being searchable, and searchable without being reliably learnable.

3.3 Temporal Intervention Structures

We use $J = 5$ execution calls by default in our experiments and represent an intervention with an effective correction schedule $\mathbf{q} = (q_0, \ldots, q_{J-1})$, where each nonzero $q_j \in \mathbb{R}^{d_m}$ is an admissible normalized correction satisfying the constraint in Eq. 3. Given $\eta, \eta^{(1)}, \eta^{(2)}, \eta_j \in \mathbb{R}^{d_m}$ denotes admissible effective corrections, we compare

$$\begin{aligned} \mathbf{q}_{\text{Single}}(\eta) &= (\eta, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}), & \mathbf{q}_{\text{Shared}}(\eta) &= (\eta, \eta, \eta, \eta, \eta), \\ \mathbf{q}_{\text{2G}}\big(\eta^{(1)}, \eta^{(2)}\big) &= \big(\eta^{(1)}, \eta^{(1)}, \eta^{(2)}, \eta^{(2)}, \eta^{(2)}\big), & \mathbf{q}_{\text{Dynamic}}(\eta_0, \ldots, \eta_4) &= (\eta_0, \eta_1, \eta_2, \eta_3, \eta_4). \end{aligned} \tag{7}$$

Here, 2-Group ($\mathbf{q}_{\text{2G}}$) divides the five execution calls into two groups: the first two calls share $\eta^{(1)}$, while the last three calls share $\eta^{(2)}$. SINGLE and SHARED each expose one d_m-dimensional variable and differ in temporal persistence. SHARED, 2-Group, and DYNAMIC expose d_m, $2d_m$, and $5d_m$ variables, respectively. Their class-containment properties are established in Appendix C.2.

4 TIDE

As shown in Fig. 2, TIDE operationalizes the three questions through a standardized execution interface, matched finite-budget search with real closed-loop validation, and search-free amortization.

4.1 Standardized Execution and Intervention

We attach the same execution-head architecture and training protocol to each frozen backbone. The heads have separately trained parameters, while the representations and latent dynamics remain

Table 2: **Cross-backbone amortization of validated Shared interventions.** A standardized proposer is evaluated search-free on 750 development units. n_{01} and n_{10} denote rescues and harms; Cube cells are protocol-ineligible for having fewer than 24 validated teachers.

Task	Backbone	Direct	Proposer	$\mathbf{n_{01}}$	$\mathbf{n_{10}}$	$\mathbf{\Delta SR}$
A. Search-free paired evaluation						
Push-T	LeWM	520/750	524/750	7	3	**+0.53 pp**
Push-T	Fast-LeWM	469/750	475/750	11	5	**+0.80 pp**
TwoRoom	LeWM	394/750	398/750	6	2	**+0.53 pp**
TwoRoom	Fast-LeWM	449/750	462/750	15	2	**+1.73 pp**
B. Preregistered eligibility audit						
Cube	LeWM	*Ineligible: 22 validated teachers* < 24				
Cube	Fast-LeWM	*Ineligible: 15 validated teachers* < 24				
Eligible total	**4 cells**	**1832/3000**	**1859/3000**	**39**	**12**	**+0.90 pp**

frozen. After checkpoint selection, the complete representation–execution systems are frozen before intervention.

For the standardized interface, the native intent is $m_j = z_g - z_j$. For any intent input $\widetilde{m}_j$, define

$$h_j(\widetilde{m}_j) = \big[z_j;\, \widetilde{m}_j;\, z_j \odot \widetilde{m}_j;\, \bar{u}_{j-1};\, e_j\big], \qquad u_j(\widetilde{m}_j) = \mathcal{B}(\psi_\phi(h_j(\widetilde{m}_j)))\,, \tag{8}$$

where $\bar{u}_{j-1}$ is the normalized previous action block, e_j is the call-index embedding, ψ_ϕ is the trained execution head, $\odot$ denotes element-wise multiplication and $\mathcal{B}$ maps its output to the valid action range. Direct execution uses $\widetilde{m}_j = m_j$, whereas intervention uses $\widetilde{m}_j = m_j^\star$ from Eq. 3. Thus, the correction changes only the intent-dependent head input.

For SHARED, the same effective normalized correction is reused across calls, while its absolute correction $\Delta m_j = \mathrm{RMS}(m_j)q$ is recomputed from the current intent. The complete head architecture and recurrent rollout are given in Appendix A.3.

4.2 Finite-Budget Search and Real Validation

For candidate schedule $\mathbf{q}$, the frozen system produces predicted latent and action-block sequences $\widehat{z}_{1:J}(\mathbf{q})$ and $\widehat{u}_{0:J-1}(\mathbf{q})$. Their frozen goal-directed cost is

$$\mathcal{J}(\mathbf{q}; x) = C(\widehat{z}_{1:J}(\mathbf{q}), \widehat{u}_{0:J-1}(\mathbf{q}), z_g)\,, \tag{9}$$

where C is the scalar predicted rollout cost and lower values are preferred. CEM returns the lowest-cost candidate among B objective evaluations. All temporal structures receive the same B; this budget counts frozen objective evaluations, not real environment rollouts. Predicted cost ranks candidates only. A rescue is credited only after the selected correction succeeds under real closed-loop execution. Direct execution and intervention are evaluated as matched causal twins and must agree before treatment. Identity violations are marked unresolved, retained in the fixed denominator, and handled through the bounds in Appendix C.6. Paired effects follow Eq. 6.

4.3 Search-Free Amortization

TIDE retains only searched SHARED interventions confirmed by real closed-loop success. At the treated solver call, z_0 denotes the latent encoding of the current real observation, not necessarily the initial state of the episode. The learnable proposer f_θ takes the current latent and goal latent as input and outputs an unconstrained correction s_θ. We project this output using the same operator as in Eq. 3:

$$s_\theta = f_\theta(z_0, z_g), \qquad q_\theta = \Pi_\rho(s_\theta), \qquad \mathbf{q}_\theta = (q_\theta, \ldots, q_\theta), \tag{10}$$

where s_θ is the raw output, q_θ its projected effective correction, and $\mathbf{q}_\theta$ the resulting SHARED schedule. The training objective is

$$\mathcal{L}_{\text{amort}} = \mathcal{L}_{\text{pos}} + \mathcal{L}_{\text{pres}} + \lambda\, \mathcal{L}_{\text{energy}}. \tag{11}$$

Table 3: **Matched validation on the conditioned INTACT (Sun et al., 2026) Push-T cohort.** All structures use the same 136 states. Optimizer bounds include unresolved units; class lower bounds also count validated interventions found by other structures when admissible to that class.

Structure	Confirmed	Optimizer bounds	Class lower bound
Single	$25/136$	$[25, 27]/136$	$\geq 25/136$
Shared	$36/136$	$[36, 38]/136$	$\geq 36/136$
Dynamic	$\mathbf{58/136}$	$\mathbf{[58, 60]/136}$	$\mathbf{\geq 63/136}$

Here, λ is set to 0.1 by default, $\mathcal{L}_{\text{pos}}$ matches behavior from validated searched interventions, $\mathcal{L}_{\text{pres}}$ preserves same-state direct behavior, and $\mathcal{L}_{\text{energy}}$ penalizes raw correction magnitude on preservation states. Positive supervision covers only the searched prefix executed during validation; an unexecuted suffix uses the same-state direct target. Exact targets, loss definitions, and optimization settings are given in Appendix A.8. Only proposer parameters are optimized, and checkpoint selection uses an episode-disjoint training-side calibration split. The selected proposer is evaluated without search on the complete paired development population; learnability is determined by the resulting closed-loop rescues and harms under Eq. 6.

5 Experiments

5.1 Experimental Setup

We study Push-T (Zhou et al., 2025b), Cube (Park et al., 2025), Reacher (Tassa et al., 2018), and TwoRoom (Sobal et al., 2025) with frozen LeWM (Maes et al., 2026) and Fast-LeWM (Gao & Xu, 2026) backbones, each equipped with the same standardized execution-head architecture. After training the heads, all representation–execution components are frozen. Direct is evaluated on 750 development units per cell; Single, Shared, and Dynamic are searched on the same 64 Direct failures at $B \in \{32, 64, 128, 256\}$ objective evaluations. Only real closed-loop outcomes count as rescues, and unresolved pairs remain in the fixed denominator. Reachability uses a separate matched 136-state cohort; amortization uses train-side teachers and the full development population. After freezing models and protocols, we also perform a one-time confirmation on 448 previously untouched, episode-disjoint units per task (Appendix B.3).

5.2 Reachability: Validated Temporal Interventions

As shown in Tab. 3, matched inversion followed by fresh closed-loop validation yields 25, 36, and 58 successes for SINGLE, SHARED, and DYNAMIC, respectively, on a 136-state INTACT Push-T cohort conditioned on prior successful reference search. This cohort is separate from the cross-backbone evaluation cohorts. DYNAMIC succeeds on 26 resolved states missed by SHARED, versus four in the opposite direction: a fixed-denominator difference of $22/136$, with bounds $[20/136, 24/136]$ (exact paired $p = 5.95 \times 10^{-5}$). Its difference from SINGLE is $33/136$, with bounds $[31/136, 35/136]$. Five states found by Single but missed by the DYNAMIC optimizer raise the DYNAMICclass lower bound to at least $63/136$, because SINGLE corrections are admissible under DYNAMIC. These successes certify reachable witnesses; unsuccessful finite search does not establish intrinsic unreachability.

5.3 Searchability: Temporal Freedom Is Non-Monotonic

As shown in Fig. 3 and Tab. 1, SINGLE, SHARED, and DYNAMIC use matched budgets on the same frozen direct-failure units within each task–backbone cell; cohorts are not paired across backbones. At $B = 256$, SHARED exceeds SINGLE in seven of eight cells; DYNAMIC trails SHARED in six, exceeds it in one, and ties it in one. Thus, additional call-specific freedom does not monotonically improve discovery under this optimizer and budget. The $B = 256$ multi-seed analysis (supplementary Fig. A) and independent held-out comparison (Appendix B.3) support this pattern. SHARED-only high-budget and C-JEPA probes establish stability and representation breadth, but do not compare SHARED with DYNAMIC (Appendix D). Raw-action CEM is strong on Reacher and TwoRoom, so we do not claim universal superiority for intent search.

Table 4: **Temporal persistence and freedom ablation.** Results report confirmed rescues on matched 64-failure cohorts at $B = 256$. Entries $r[L, U]$ include unresolved-aware bounds, and Δn denotes differences in rescue counts between adjacent structures.

Task	Backbone	A. Temporal persistence			B. Temporal freedom			
		Single	Shared	Δn Shared–Single	2-Group	Dynamic	Δn 2G–Shared	Δn Dynamic–2G
Push-T	LeWM	2	13	$[+11, +11]$	10	8	$[-3, -3]$	$[-2, -2]$
	Fast-LeWM	1	16	$[+15, +15]$	20	17	$[+4, +4]$	$[-3, -3]$
Cube	LeWM	$6[6, 8]$	$15[15, 17]$	$[+7, +11]$	$11[11, 18]$	$10[10, 13]$	$[-6, +3]$	$[-8, +2]$
	Fast-LeWM	12	$28[28, 31]$	$[+16, +19]$	$25[25, 26]$	23	$[-6, -2]$	$[-3, -2]$
Reacher	LeWM	0	0	$[0, 0]$	0	0	$[0, 0]$	$[0, 0]$
	Fast-LeWM	1	13	$[+12, +12]$	8	4	$[-5, -5]$	$[-4, -4]$
TwoRoom	LeWM	5	9	$[+4, +4]$	11	8	$[+2, +2]$	$[-3, -3]$
	Fast-LeWM	4	11	$[+7, +7]$	11	7	$[0, 0]$	$[-4, -4]$

5.4 Learnability: Partial Search-Free Amortization

We train SHARED-correction proposers from train-side interventions confirmed by real closed-loop success. Four task–backbone cells meet the preregistered minimum of 24 validated teachers. The two Cube cells fall below this threshold, with 22 and 15 teachers, while Reacher lies outside the amortization matrix. As shown in Tab. 2, all four eligible cells show positive observed net gains on their complete 750-unit development populations. Together, the proposers produce 39 rescues and 12 harms, a descriptive gain of $+0.90$ pp. Cell-level uncertainty remains substantial: only TwoRoom/Fast-LeWM is individually significant under a two-sided exact McNemar test. The dependence-preserving joint paired bootstrap gives an aggregate 95% CI of $[+0.43, +1.37]$ pp (Appendix B.4). The selected proposers also retain a modest $+0.84$ pp gain on independent held-out episodes, with a 95% CI of $[+0.28, +1.40]$ pp (Appendix B.3). These results support partial search-free amortization in eligible settings, but the gains are small and vary across cells. An aligned-cohort analysis and further mechanism diagnostics are reported in Appendix B.1 and Tab. C.

5.5 Ablation Study

As shown in Tab. 4, we first compare SINGLE and SHARED to isolate temporal persistence under a matched search budget. Both expose one d_m-dimensional correction variable, but only SHARED reuses it across calls and generally achieves more confirmed rescues. Its advantage therefore cannot be attributed to a larger number of search variables. We next compare SHARED, 2-Group, and DYNAMIC, which expose d_m, $2d_m$, and $5d_m$ scalar degrees of freedom, respectively. As shown in Fig. 4 and quantified in Tab. 4, 2-Group improves over SHARED in some settings but underperforms it in others, while DYNAMIC provides no consistent advantage. Thus, additional call-specific freedom does not monotonically improve finite-budget searchability, although this ablation does not establish dimensionality as the cause of the observed non-monotonicity.

6 Conclusion

In frozen world-model control, Direct failure does not reveal whether successful behavior is unexecutable, difficult to discover, or difficult to amortize. We introduced **TIDE**, which studies these possibilities through a standardized execution interface and bounded temporal intent interventions, while holding the representation, dynamics, and execution policy fixed. Call-specific corrections expose more real, validated behavior in matched evaluation, whereas finite-budget search across tasks and frozen representations more consistently favors persistent SHARED corrections. Validated interventions are only partially amortized, with modest and heterogeneous gains. These results separate executable capacity, finite-budget discoverability, and amortized control.

AI USE STATEMENT

Generative AI tools assisted with language editing, LaTeX formatting, code review, debugging, and the organization of experimental analyses. The authors determined the research design, executed and verified all experiments, reviewed all AI-assisted content, and take full responsibility for the submission.

ETHICS STATEMENT

This work uses existing research data, pretrained checkpoints, and simulated control environments; it involves no human participants, personal data, or physical-system deployment. Our claims are limited to the evaluated tasks and protocols, and physical or safety-critical deployment would require additional validation.

REPRODUCIBILITY STATEMENT

Appendix A documents the datasets, architecture, optimization, budgets, seeds, checkpoint selection, causal-twin checks, and missing-outcome rules. The supplement additionally reports paired statistics, held-out confirmation, and artifact provenance. Upon publication, we intend to release the implementation, configurations, population identifiers, and aggregation scripts, subject to upstream licenses.

REFERENCES

Brandon Amos and Denis Yarats. The differentiable cross-entropy method. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 291–302. PMLR, 2020.

Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. *CoRR*, abs/2008.05556, 2020.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael G. Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pp. 15619–15629. IEEE, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew J. Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. *CoRR*, abs/2506.09985, 2025.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Trans. Mach. Learn. Res.*, 2024, 2024.

Homanga Bharadhwaj, Kevin Xie, and Florian Shkurti. Model-predictive control via cross-entropy and gradient-based optimization. In *Proceedings of the 2nd Annual Conference on Learning for Dynamics and Control, L4DC 2020, Online Event, Berkeley, CA, USA, 11-12 June 2020*, volume 120, pp. 277–286. PMLR, 2020.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 4759–4770, 2018.

Pieter-Tjerk de Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Ann. Oper. Res.*, 134(1):19–67, 2005.

Yuntian Gao and Xiangyu Xu. Fast leworldmodel. *CoRR*, abs/2606.26217, 2026.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 2455–2467, 2018a.

David Ha and Jürgen Schmidhuber. World models. *CoRR*, abs/1803.10122, 2018b.

Danijar Hafner, Timothy P. Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97, pp. 2555–2565. PMLR, 2019.

Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Danijar Hafner, Timothy P. Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy P. Lillicrap. Mastering diverse domains through world models. *CoRR*, abs/2301.04104, 2023.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

Kevin Huang, Sahin Lale, Ugo Rosolia, Yuanyuan Shi, and Anima Anandkumar. CEM-GD: cross-entropy method with gradient descent planner for model-based reinforcement learning. *CoRR*, abs/2112.07746, 2021.

Thomas Kipf, Gamaleldin Fathy Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional object-centric learning from video. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels. *CoRR*, abs/2603.19312, 2026.

Malte Mosbach, Jan Niklas Ewertz, Angel Villar-Corrales, and Sven Behnke. SOLD: slot object-centric latent dynamics models for relational manipulation learning from pixels. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025.

Heejeong Nam, Quentin Le Lidec, Lucas Maes, Yann LeCun, and Randall Balestriero. Causal-jepa: Learning world models through object-level latent interventions. *CoRR*, abs/2602.11389, 2026.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations (ICLR)*, 2025.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Jörg Stückler, Michal Rolínek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. *CoRR*, abs/2008.06389, 2020.

Vlad Sobal, Wancong Zhang, Kynghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. *arXiv preprint arXiv:2502.14819*, 2025.

Junhan Sun, Hao Zhao, and Guofeng Zhang. INTACT: isomorphic intent-to-action learning for search-free world models. *CoRR*, abs/2607.26056, 2026.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy P. Lillicrap, and Martin A. Riedmiller. Deepmind control suite. *CoRR*, abs/1801.00690, 2018.

Tingwu Wang and Jimmy Ba. Exploring model-based planning with policy networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M. Rehg, Byron Boots, and Evangelos A. Theodorou. Information theoretic MPC for model-based reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*, pp. 1714–1721. IEEE, 2017.

Ziyi Wu, Nikita Dvornik, Klaus Greff, Thomas Kipf, and Animesh Garg. Slotformer: Unsupervised visual dynamics simulation with object-centric models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: world models on pre-trained visual features enable zero-shot planning. In *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267. PMLR / OpenReview.net, 2025a.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: world models on pre-trained visual features enable zero-shot planning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Forty-second International Conference on Machine Learning, ICML 2025, Vancouver, BC, Canada, July 13-19, 2025*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025b.

Supplementary Material

This supplementary material is organized as follows:

- Appendix A provides implementation and evaluation details.
- Appendix B presents additional experimental analyses and statistical results.
- Appendix C provides formal definitions and theoretical properties.
- Appendix D reports additional robustness and representation-breadth evaluations.
- Appendix E discusses the scope and limitations of the study.
- Appendix F provides the large-language-model usage statement.

A Implementation Details

This section provides the architecture, optimization, search, and evaluation details omitted from the main paper. Unless stated otherwise, all data populations, model checkpoints, hyperparameters, and random seeds are fixed before the corresponding scientific outcomes are inspected.

A.1 Datasets and Evaluation Protocol

Tasks and datasets. We evaluate Push-T, Cube, Reacher, and TwoRoom using their corresponding expert datasets and simulator environments. Each evaluation unit consists of an initial state, an associated observation, and a goal taken 25 primitive steps later in the same demonstration. We retain only episodes containing enough valid steps to construct the state–goal pair and its 25-step expert action target.

Table A summarizes the task-specific action dimensions. Every execution-head call predicts five primitive actions; hence the packed action dimension is five times the primitive action dimension.

Train and development populations. For each task, we construct an episode-disjoint reservoir containing 3,600 training units and 750 development units. The same ordered unit identifiers are used for LeWM and Fast-LeWM. No episode contributes units to both splits.

The 3,600 training units and 750 development units are used to train and select the standardized execution head. Direct closed-loop performance is subsequently evaluated on all 750 development units. For temporal-correction search, we freeze

$$N_b = \min(64, F_b)$$

Direct failures for each task–backbone cell, where F_b is the number of Direct failures in that cell. A cell is search-eligible when $F_b \geq 30$. All eight reported cells contain at least 64 Direct failures and therefore use the common denominator $N_b = 64$. Units are selected without replacement before any temporal-correction outcome is inspected.

Success criterion. We retain the success and termination predicates implemented by each task environment. The predicted world-model cost is used only to rank search candidates and is never treated as task success. A rescue is credited only when the selected intervention succeeds under real simulator execution. Closed-loop evaluation is capped at 50 executed environment steps.

Evaluation boundaries. The matched reachability study uses a separate frozen cohort of 136 Push-T states. The searchability study uses the 64 Direct failures frozen independently for each task–backbone cell. The amortization study acquires teachers from an episode-disjoint train-side population and evaluates the selected proposer without search on the complete 750-unit development population. After all models and protocols were frozen, we performed a one-time confirmation on 448 previously untouched, episode-disjoint held-out units per task (Appendix B.3). These units were not used for training or checkpoint selection.

A.2 Frozen Backbones and Execution Interface

Frozen representations. We use pretrained LeWM and Fast-LeWM instances for every task. Both backbones produce 192-dimensional latent representations. Their encoders, action encoders, goal representations, and latent dynamics remain frozen throughout execution-head training, temporal search, and amortization.

The resulting experiments therefore compare frozen world-model representations through a common learned execution interface. They should not be interpreted as comparisons between independently designed controllers associated with the two backbones.

Action-block interface. The standardized head performs $J = 5$ recurrent calls. Each call predicts five primitive actions, producing a 25-step planned sequence. The packed action dimension is 10 for Push-T, Reacher, and TwoRoom and 25 for Cube.

The head operates in normalized action coordinates. Raw actions are bounded coordinate-wise by the task action box, while the corresponding normalized lower and upper bounds are computed using the frozen dataset statistics. The same bounded normalized action block is used as:

1. the action context for the next execution-head call;
2. the input to the frozen latent transition; and
3. the action block returned for environment execution after applying the frozen inverse normalization.

No separate action tensor is used for recurrence, latent prediction, and real execution.

Backbone-specific latent transitions. For LeWM, each predicted block is appended to its native autoregressive state–action history before applying the frozen predictor. For Fast-LeWM, the predicted block is processed by its frozen action encoder and dense latent transition. These adapters retain each backbone's native transition semantics while presenting the same execution-head contract.

A.3 Standardized Intent-Conditioned Execution Head

Architecture. At execution-head call j, let $z_j, z_g \in \mathbb{R}^{d_z}$ denote the current latent state and goal representation, respectively. For the standardized interface, $d_m = d_z = 192$. The native and corrected intents are

$$m_j = z_g - z_j, \qquad m_j^\star = m_j + \Delta m_j, \qquad m_j, m_j^\star \in \mathbb{R}^{d_m}. \tag{12}$$

The execution-head input is

$$h_j = \left[z_j;\, m_j^\star;\, z_j \odot m_j^\star;\, \bar{u}_{j-1};\, e_j \right], \tag{13}$$

where $\odot$ denotes elementwise multiplication, $\bar{u}_{j-1} \in \mathbb{R}^{d_u}$ is the previous bounded action block in normalized action coordinates, and $e_j = E_{\text{call}}[j] \in \mathbb{R}^{32}$ is the learned embedding of call index j. We initialize $\bar{u}_{-1} = \mathbf{0}$ at the beginning of each solve and set $\bar{u}_j := u_j$ after each head prediction.

The execution head contains three hidden layers of width 1,024. Each hidden layer uses a linear transformation, LayerNorm, and GELU. The output is divided into a pre-squash action mean $\widetilde{u}_j \in \mathbb{R}^{d_u}$ and a log standard deviation $\log \sigma_j \in \mathbb{R}^{d_u}$. The latter is clipped componentwise to $[-5, 2]$.

Let $a_{\min}, a_{\max} \in \mathbb{R}^{d_u}$ denote the componentwise task-specific bounds in normalized action coordinates. We define

$$a_{\text{mid}} = \frac{a_{\max} + a_{\min}}{2}, \qquad a_{\text{scale}} = \frac{a_{\max} - a_{\min}}{2}.$$

The bounded action mean is

$$u_j = a_{\text{mid}} + a_{\text{scale}} \odot \tanh(\widetilde{u}_j).$$

Deterministic evaluation executes this bounded mean rather than sampling from the predicted Gaussian.

Algorithm 1 Standardized execution-head latent rollout

Require: Initial latent z_0, goal latent z_g, frozen transition F, execution head f_ϕ, call embeddings E_{call}, effective correction schedule $\mathbf{q}$, normalized action bounds $(a_{\min}, a_{\max})$, and number of calls J

Ensure: Action blocks $u_{0:J-1}$ and predicted latent states $z_{1:J}$

$\bar{u}_{-1} \leftarrow \mathbf{0}$
$a_{\text{mid}} \leftarrow (a_{\max} + a_{\min})/2$
$a_{\text{scale}} \leftarrow (a_{\max} - a_{\min})/2$
for $j = 0, \ldots, J-1$ **do**
  $m_j \leftarrow z_g - z_j$
  $\Delta m_j \leftarrow \mathrm{RMS}(m_j)\, q_j$
  $m_j^\star \leftarrow m_j + \Delta m_j$
  $e_j \leftarrow E_{\text{call}}[j]$
  $h_j \leftarrow [z_j;\, m_j^\star;\, z_j \odot m_j^\star;\, \bar{u}_{j-1};\, e_j]$
  $(\widetilde{u}_j, \log \sigma_j) \leftarrow f_\phi(h_j)$
  $\log \sigma_j \leftarrow \mathrm{clip}(\log \sigma_j, -5, 2)$
  $u_j \leftarrow a_{\text{mid}} + a_{\text{scale}} \odot \tanh(\widetilde{u}_j)$
  $\bar{u}_j \leftarrow u_j$
  $z_{j+1} \leftarrow F(z_j, u_j)$
end for
return $u_{0:J-1}, z_{1:J}$

Student-unrolled training. The head is trained by unrolling its own predictions through the frozen backbone transition. After every predicted action block, the frozen transition produces the next latent state. The following head call therefore depends on the preceding predicted action and predicted latent state rather than teacher-forced alternatives.

Algorithm 1 summarizes this latent rollout. The effective normalized correction schedule $\mathbf{q} = (q_0, \ldots, q_{J-1})$ is assumed to satisfy the relative-RMS constraint defined in Sec. A.4. Direct rollout is recovered by setting $q_j = \mathbf{0} \in \mathbb{R}^{d_m}$ for every j.

Algorithm 1 describes the latent rollout used for execution-head training and candidate evaluation. It does not replace real closed-loop validation. During real evaluation, selected actions advance the simulator, and subsequent replanning uses the resulting real observation.

Training objective. For each call, the execution head predicts an action mean $\mu_\phi \in \mathbb{R}^{d_u}$ and log standard deviation $\log \sigma_\phi \in \mathbb{R}^{d_u}$. Let $u^{\text{exp}} \in \mathbb{R}^{d_u}$ denote the corresponding expert action block in normalized coordinates. We minimize the diagonal-Gaussian negative log-likelihood

$$\mathcal{L}_{\text{head}} = \frac{1}{2}\mathbb{E}_{\text{train}}\left[(u^{\text{exp}} - \mu_\phi)^2 \odot \exp(-2\log\sigma_\phi) + 2\log\sigma_\phi\right], \tag{14}$$

where the expectation averages over training examples, execution calls, and action coordinates. Only the execution-head parameters ϕ are optimized. All backbone parameters remain in evaluation mode with gradients disabled.

Optimization and checkpoint selection. We train for 1,000 updates using AdamW with learning rate 3×10^{-4}, weight decay 10^{-2}, batch size 128, and gradient-norm clipping at 1.0. The execution-head seed is 20260908. We save checkpoints at

$$\{10, 25, 50, 100, 200, 300, 500, 750, 1000\}$$

updates.

The selected checkpoint minimizes bounded-action RMSE on the frozen episode-disjoint development split. Ties are resolved in favor of the earliest checkpoint. After selection, the complete representation–execution system is frozen.

A.4 TEMPORAL INTENT CORRECTIONS

Relative-RMS constraint. Let $s_j \in \mathbb{R}^{d_m}$ denote an unconstrained dimensionless correction variable, where $d_m = 192$ in all standardized-interface experiments. We project it into the normalized relative-RMS ball:

$$q_j = \Pi_\rho(s_j), \qquad \mathrm{RMS}(q_j) \le \rho,$$

with $\rho = 0.25$. The absolute correction applied at call j is

$$\Delta m_j = \mathrm{RMS}(m_j)\, q_j.$$

Therefore,

$$\mathrm{RMS}(\Delta m_j) \le 0.25\ \mathrm{RMS}(m_j).$$

The native-intent RMS is recomputed from the current latent state at every call. Shared consequently reuses one dimensionless correction variable while its absolute correction scale adapts to the evolving latent state. No additional coordinate-wise hard bound is imposed on s_j.

Temporal structures. For $J = 5$ calls, let $\mathbf{0} \in \mathbb{R}^{d_m}$ denote the zero correction. The effective schedules are

$$\begin{aligned}
\mathbf{q}_{\mathrm{Single}} &= (q, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}),\\
\mathbf{q}_{\mathrm{Shared}} &= (q, q, q, q, q),\\
\mathbf{q}_{\mathrm{2G}} &= (q^{(1)}, q^{(1)}, q^{(2)}, q^{(2)}, q^{(2)}),\\
\mathbf{q}_{\mathrm{Dynamic}} &= (q_0, q_1, q_2, q_3, q_4).
\end{aligned}$$

Single and Shared each expose d_m scalar degrees of freedom. The 2-Group and Dynamic variants expose $2d_m$ and $5d_m$ scalar degrees of freedom, respectively. The 2-Group structure is used only for the temporal-freedom ablation.

A.5 FINITE-BUDGET CEM SEARCH

Objective. Each candidate correction schedule is passed through Algorithm 1. Candidates are ranked by the mean-squared distance between the terminal predicted latent and the encoded goal:

$$\mathcal{J}(\mathbf{q}; x) = \|\widehat{z}_J(\mathbf{q}) - z_g\|_2^2 .$$

This objective is used only to rank candidates. Reachability and rescue are credited only after real environment execution.

CEM configuration. Every CEM iteration evaluates 32 candidates and retains 8 elites. The initial distribution has zero mean and coordinate-wise standard deviation 0.25. The distribution mean and scale are recomputed from the elites after each iteration, with the scale lower-bounded by 10^{-4}.

The zero-correction candidate is evaluated in every iteration. Beginning with the second iteration, the current CEM mean is also inserted as an explicit candidate. The final output is the best candidate actually evaluated across all iterations.

We use 1, 2, 4, and 8 iterations, corresponding to

$$B \in \{32, 64, 128, 256\}$$

objective evaluations. Candidate count, elite count, and evaluation budget are identical across temporal structures. The larger search dimension of Dynamic is part of the finite-budget searchability question and is not compensated with additional evaluations. The primary search seed is 20260906.

A.6 REAL CLOSED-LOOP AND PAIRED EVALUATION

Execution and replanning. A solver call produces five action blocks, each containing five primitive actions. The resulting 25-action plan is executed in the real environment until its end or earlier termination. If the episode remains active, the next solver call encodes the new real observation rather than continuing from the predicted latent state. Evaluation permits at most two solver calls and 50 executed primitive actions. Single, Shared, and Dynamic specify corrections across the five execution-head calls *within* the treated solver call; the treatment is introduced only at the second solver call.

Table A: **Task-specific implementation details.** Each execution-head call predicts five primitive actions.

Task	Dataset identifier	Action dim.	Packed dim.
Push-T	`pusht_expert_train`	2	10
Cube	`cube_single_expert`	5	25
Reacher	`dmc/reacher_random`	2	10
TwoRoom	`tworoom`	2	10

Causal twins. Direct and treatment run as same-run causal twins. Before treatment, they must agree on the restored state, goal, first solver input, activity status, and entire first action plan. At the second solver call, both twins must remain active and their factual states must pass the identity check. The absolute tolerance is 10^{-9} for Push-T, Reacher, and TwoRoom, and 10^{-5} for Cube; the latter was fixed before intervention outcomes to accommodate small deterministic MuJoCo drift near contact.

Resolved and unresolved units. A pair is resolved if both twins reach the second call, pass identity, and receive the intended treatment. If neither reaches that call and their final outcomes agree, it is a resolved early outcome without treatment. A pair is unresolved if only one twin reaches the second call, identity fails, treatment is not applied despite both being active, or the early outcomes disagree. Unresolved units are never counted as failures.

Paired accounting. For resolved pairs, n_{01} counts Direct failures rescued by treatment and n_{10} counts Direct successes harmed by treatment. With the original denominator N, the paired effect follows Eq. 6. If U units are unresolved, we report the fixed-denominator worst-case interval

$$\left[\frac{n_{01} - n_{10} - U}{N}, \frac{n_{01} - n_{10} + U}{N}\right].$$

This is a missing-outcome bound, not a sampling confidence interval. Method-to-method comparisons retain the same unit identifiers.

A.7 Matched Reachability Protocol

This is a separate INTACT Push-T mechanism study, not a cell of the LeWM/Fast-LeWM standardized-head search matrix. Its frozen native Base and residual execution checkpoints are identical across Single, Shared, and Dynamic; their identities and hashes are fixed in the experiment protocol.

The 136-state cohort consists of all frozen Direct failures for which a preceding Actor-CEM reference search produced a real successful execution. It contains 105 train-side and 31 development-side states and was fixed before the matched comparison. Each structure starts from the same restored state and goal, uses eight deterministic inversion restarts with 300 Adam updates per restart, and is validated by fresh real execution. Validation executes only the prefix covered by the frozen reference action tape; actions beyond that horizon are not credited.

Because the cohort is conditioned on prior reference-search success, this is a matched mechanism comparison, not an unconditional Push-T success-rate estimate. A validated intervention certifies a successful witness, whereas failure under finite inversion does not establish intrinsic unreachability.

A.8 Search-Free Amortization

Teacher population. The preregistered amortization matrix covers Push-T, Cube, and TwoRoom with both frozen backbones; Reacher was not subjected to teacher acquisition and is outside the eligibility audit. For each cell, we freeze 750 episode-disjoint train-side units and retain the existing 750-unit development population. Shared search with 256 objective evaluations is applied to all and only train-side Direct failures. A correction is retained as a teacher only after real closed-loop success.

For preservation, at most 128 Direct-success states are selected by a frozen hash order. Training requires at least 24 validated teachers and 24 valid preservation states. No population is expanded or replaced after observing eligibility.

Training targets. Positive targets retain the searched action prefix actually executed during validation. Any unexecuted suffix uses zero-correction Direct actions recomputed for the same states in the current minibatch. Preservation targets are likewise recomputed from zero-correction Direct execution in that minibatch. Thus, stored actions from a different acquisition batch are not used as Direct targets.

Proposer. At the treated second solver call, z_0 denotes the latent representation of the *current real observation*, not the episode's initial observation. The proposer receives

$$[z_0;\, m_0;\, z_0 \odot m_0], \qquad m_0 = z_g - z_0,$$

and predicts one 192-dimensional raw Shared variable s_θ. It has two hidden layers of width 512 with LayerNorm and GELU, followed by a 192-dimensional output layer. Zero initialization of the final layer makes the initial proposer exactly equivalent to Direct.

The raw output has no coordinate-wise bound. It is projected by the same relative-RMS rule used in Shared search; the resulting effective correction is reused across the five execution-head calls and student-unrolled through the frozen execution system.

Objective and checkpoint selection. Let $\mathcal{D}_{\mathrm{pos}}$ denote the frozen validated-teacher population and let $\mathcal{D}_{\mathrm{pres}}$ denote the frozen Direct preservation population. The terms $\mathcal{L}_{\mathrm{pos}}$ and $\mathcal{L}_{\mathrm{pres}}$ are the corresponding minibatch-averaged Smooth L1 action losses. The implemented proposer loss is

$$\mathcal{L}_{\mathrm{amort}} = \mathcal{L}_{\mathrm{pos}} + \mathcal{L}_{\mathrm{pres}} + 0.1\,\mathcal{L}_{\mathrm{energy}},$$

where the first two terms use Smooth L1 action loss. The final term penalizes the *raw*, unprojected proposer output on preservation states:

$$\mathcal{L}_{\mathrm{energy}} = \mathbb{E}_{x\sim\mathcal{D}_{\mathrm{pres}}}\left[\frac{\|s_\theta(x)\|_2^2}{d_m}\right]. \tag{15}$$

Only proposer parameters are updated. Each step samples 32 positive and 32 preservation examples. We use AdamW for 1,000 updates with learning rate 3×10^{-4}, weight decay 10^{-2}, gradient-norm clipping at 1.0, and seed 20260914.

Checkpoints are saved at updates $\{10, 25, 50, 100, 200, 300, 500, 750, 1000\}$. Teacher and preservation populations each use a deterministic 80/20 fit and train-side calibration split, with at least four calibration examples per group. The checkpoint minimizing calibration loss is selected, with ties resolved by the earliest update. Development outcomes are not used for selection. The selected proposer is evaluated once without search on all 750 development units under the paired protocol in Sec. A.6.

A.9 Ablations and Search References

2-Group ablation. The 2-Group condition is evaluated at $B = 256$ on the same frozen 64-state Direct-failure populations used by Single, Shared, and Dynamic. The execution interface, correction radius, candidate count, objective, and real validation protocol remain unchanged.

Raw-action references. We evaluate zero-initialized and standardized-head-initialized raw-action CEM. Both operate in the same bounded normalized action domain and use 32 candidates, 8 elites, and the same objective-evaluation budgets as temporal intent search. Because these references change the optimized interface, they are not treated as TIDE ablations.

Multi-seed robustness. The main search uses seed 20260906. We repeat the $B = 256$ Single, Shared, and Dynamic conditions using seeds 20260907 and 20260908. All repetitions reuse the frozen populations, checkpoints, radius, and evaluation code. No seed is selected based on its outcome.

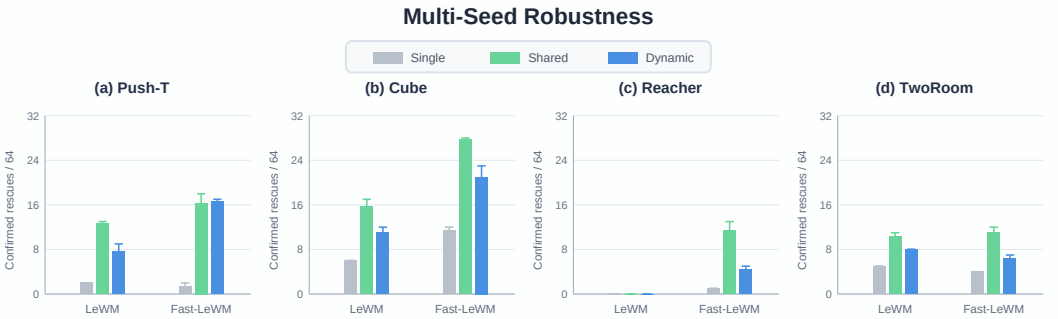

Figure A: **Multi-seed robustness of temporal-correction search.** Single, Shared, and Dynamic are evaluated with three search seeds at 256 objective evaluations on the same 64 Direct failures per task–backbone pair. Bars show mean confirmed rescues and whiskers indicate the seed range. Shared remains stronger than Single, while Dynamic provides no consistent advantage over Shared.

Table B: **Aligned evidence on a common frozen cohort.** H denotes validated success under 2,048 objective evaluations, whereas S denotes success under $B = 256$. The proposer is defined only for Shared. Its two successes both belong to H, and one also belongs to S.

Structure	$\lvert H \rvert$	$\lvert S \rvert$	$\lvert S \cap H \rvert$	Proposer
Single	2	2	2	–
Shared	16	13	13	2
Dynamic	14	8	8	–

A.10 Determinism and Artifact Provenance

Formal runs use Python 3.10 and PyTorch 2.6.0 with CUDA 12.4. Python, NumPy, PyTorch, CUDA, simulator, and search random states are set explicitly. PyTorch deterministic algorithms are enabled, and the CuBLAS workspace is configured before importing PyTorch.

Each 750-unit evaluation is divided into 30 shards of 25 units. A fresh solver and environment are constructed for every shard. Simulator and HDF5 stages use at most two concurrent workers, while cached training jobs may run independently on different GPUs.

Every formal stage writes to a new output directory and refuses to overwrite existing artifacts. Technical failures terminate the affected execution chain and are never recorded as scientific failures. Frozen protocol manifests store hashes of datasets, ordered populations, backbone states, execution-head checkpoints, code modules, and evaluation reports. These hashes are verified before each formal stage, and backbone hashes are checked again afterward to confirm that the frozen parameters remain unchanged.

B More Experiments

B.1 Aligned Three-Way Analysis on a Common Cohort

We align the three probes on the same pre-frozen cohort of 64 Push-T/LeWM Direct failures, selected without reference to intervention outcomes. For Single, Shared, and Dynamic, high-budget validation (2,048 evaluations) finds 2, 16, and 14 successes; finite-budget search ($B = 256$) finds 2, 13, and 8, respectively (Tab. B). Every finite-budget success belongs to its corresponding high-budget set. The frozen Shared proposer succeeds without search on two units, both in the high-budget Shared set and one also in the finite-budget set. All units are resolved and pass causal-identity and frozen-model checks. Thus, the three procedures recover different success sets even on identical states. High-budget success is a validated reachability lower bound, not a complete characterization of the intervention class or a universal ordering of Shared and Dynamic.

Table C: **Additional amortization diagnostics on the INTACT Push-T cohort.** Offline progress does not reliably predict closed-loop improvement, and same-state analysis reveals non-unique successful corrections. Here, n_{01} and n_{10} denote rescues and harms, respectively.

Probe	Offline signal	Closed-loop / behavioral result	Diagnosis
A. Amortization probes			
Correction regression	Cosine: 0.805 train, 0.100 dev.	Direct $622/750$, proposer $622/750$; $(n_{01}, n_{10}) = (3, 3)$.	Poor transfer; zero net gain.
Student-unrolled objective	Moves toward an executable teacher on $9/12$ states.	$(n_{01}, n_{10}) = (0, 0)$ in the paired screen.	Offline progress without rescue.
Search-distilled policy	Distillation objective is optimized.	Direct $570/750$, distilled $570/750$; $(n_{01}, n_{10}) = (0, 0)$ among 656 resolved pairs; 94 unresolved.	No observed paired change on resolved pairs.
B. Mechanistic diagnosis			
Same-state solutions	15 states, 43 teachers, 59 pairs; 11 states have multiple teachers.	Distinct corrections induce similar actions and physical behavior.	Successful targets are non-unique.

B.2 Additional Amortization Diagnostics

Tab. C reports complementary INTACT Push-T probes. Correction regression, student-unrolled supervision, and search-distillation exhibit offline progress without a corresponding net improvement among their evaluated pairs. Successful same-state corrections can also be non-unique, making pointwise correction targets potentially ambiguous. These diagnostics concern particular learning procedures; they do not negate the modest positive gains of the standardized cross-backbone proposer or establish that amortization is generally impossible.

B.3 Independent Held-Out Confirmation

After freezing models and protocols, we select 448 previously untouched, episode-disjoint units per task, with the same ordered IDs across backbones. The denominator was set before outcome inspection: an availability audit found only 478 untouched Push-T episodes. Direct uses all 448 units; each search structure receives $B = 256$ on up to 64 hash-selected Direct failures per cell. Frozen Shared proposers are evaluated search-free on all 448 units where available (Tab. D).

Shared confirms more rescues than Dynamic in seven of eight cells, without establishing universal dominance. Across four proposer cells, 22 rescues and 7 harms yield a descriptive $+0.84$ pp gain. A dependence-preserving joint paired bootstrap gives a 95% CI of $[+0.28, +1.40]$ pp. Push-T/LeWM has zero net gain, so the aggregate should not be read as a uniform effect across cells.

B.4 Amortization Statistics

Tab. E reports paired-bootstrap percentile 95% intervals from 200,000 replicates and cell-level two-sided exact McNemar tests. For aggregate uncertainty, we jointly resample shared unit IDs within each task, retaining both backbone outcomes for every sampled unit; tasks are resampled independently. Each unit belongs to a distinct episode, so unit-level and episode-cluster resampling coincide here. The four-cell aggregate is descriptive rather than an estimate for a universal task population.

C Formal Properties of Temporal Intent Intervention

This appendix formalizes the properties required by the TIDE analysis. The results distinguish intervention capacity, finite-budget search, and amortization. They do not assert that CEM is optimal,

Table D: **Independent held-out confirmation.** Direct reports successes among 448 units. Search columns report confirmed rescues on the frozen Direct-failure cohort of size N_s, with $B = 256$ evaluations per structure. Entries $r[L, U]$ give confirmed rescues and fixed-denominator bounds when units are unresolved; plain counts are fully resolved. The final two columns evaluate frozen Shared proposers without search on all 448 units. Dashes indicate that no proposer was evaluated. The total is descriptive and includes only the four proposer-eligible cells.

Task	Backbone	Direct	$\mathbf{N_s}$	Single	Shared	Dynamic	$(\mathbf{n_{01}}, \mathbf{n_{10}})$	$\mathbf{\Delta}_{\text{amort}}$
Push-T	LeWM	311/448	64	1	7	6	$(1, 1)$	0.00 pp
	Fast-LeWM	297/448	64	5	16	17	$(6, 2)$	+0.89 pp
Cube	LeWM	339/448	64	2[2, 3]	11[11, 12]	3[3, 4]	–	–
	Fast-LeWM	405/448	43	9[9, 14]	20[20, 21]	18[18, 20]	–	–
Reacher	LeWM	17/448	64	1	4	1	–	–
	Fast-LeWM	89/448	64	5	17	11	–	–
TwoRoom	LeWM	238/448	64	2	11	5	$(6, 0)$	+1.34 pp
	Fast-LeWM	273/448	64	2	10	6	$(9, 4)$	+1.12 pp
Eligible total	**4 cells**	**1119/1792**	–	–	–	–	$(\mathbf{22}, \mathbf{7})$	+0.84 **pp**

Table E: **Sampling uncertainty for search-free amortization.** Cell intervals use 200,000 paired-bootstrap replicates; the aggregate jointly resamples shared unit IDs within each task. Exact p-values are from two-sided McNemar tests.

Task	Backbone	N	$\mathbf{n_{01}}$	$\mathbf{n_{10}}$	$\mathbf{\Delta}$SR (pp)	95% CI (pp)	Exact p
Push-T	LeWM	750	7	3	+0.53	$[-0.27, +1.33]$	0.3438
Push-T	Fast-LeWM	750	11	5	+0.80	$[-0.27, +1.87]$	0.2101
TwoRoom	LeWM	750	6	2	+0.53	$[-0.13, +1.33]$	0.2891
TwoRoom	Fast-LeWM	750	15	2	+1.73	$[+0.67, +2.80]$	0.00235
Joint aggregate	**4 cells**	**3000**	**39**	**12**	+0.90	$[+0.43, +1.37]$	–

that greater temporal freedom is universally harder to optimize, or that temporal interventions cannot be learned by other model classes.

C.1 Relative-RMS Projection

Let $m \in \mathbb{R}^{d_m}$ be the native intent at one execution call, where d_m is the intent dimension defined in Sec. 3.1. Let $s \in \mathbb{R}^{d_m}$ be an unconstrained, dimensionless intervention variable. For any $v \in \mathbb{R}^{d_m}$, define

$$r(v) := \mathrm{RMS}(v) = \sqrt{\frac{1}{d_m}\|v\|_2^2}. \tag{16}$$

We obtain an effective normalized correction by projecting s onto the RMS ball of radius ρ:

$$q := \Pi_\rho(s) = \begin{cases} s, & r(s) \le \rho, \\ \rho \dfrac{s}{r(s)}, & r(s) > \rho. \end{cases} \tag{17}$$

Thus, $r(q) \le \rho$. The applied correction is

$$\Delta(m, s) = r(m)\Pi_\rho(s) = r(m)q. \tag{18}$$

When $r(m) = 0$, we define $\Delta(m, s) = \mathbf{0} \in \mathbb{R}^{d_m}$. At execution call j, the notation in the main paper is recovered by

$$q_j := \Pi_\rho(s_j), \qquad \Delta m_j := \Delta(m_j, s_j). \tag{19}$$

A small numerical constant is used in implementation only to ensure numerical stability and does not alter the prescribed relative-RMS constraint.

Proposition 1 (Admissibility and scale normalization). *For every $m, s \in \mathbb{R}^{d_m}$, the correction in Eq. 18 satisfies*

$$r(\Delta(m, s)) \le \rho\, r(m). \tag{20}$$

Moreover, for every $\alpha > 0$,

$$\Delta(\alpha m, s) = \alpha\, \Delta(m, s). \tag{21}$$

When $r(m) > 0$, the normalized correction $\Delta(m, s)/r(m) = \Pi_\rho(s)$ is independent of the magnitude of m.

Proof. By Eq. 17, $r(\Pi_\rho(s)) \le \rho$. Positive homogeneity of RMS gives

$$r(\Delta(m, s)) = r(m)\, r(\Pi_\rho(s)) \le \rho\, r(m).$$

Since $r(\alpha m) = \alpha r(m)$ for $\alpha > 0$,

$$\Delta(\alpha m, s) = r(\alpha m)\Pi_\rho(s) = \alpha r(m)\Pi_\rho(s) = \alpha\Delta(m, s).$$

The normalized identity follows from Eq. 18. □

Proposition 1 explains the semantics of a Shared intervention. The same effective normalized correction q is reused across calls, but the applied correction

$$\Delta m_j = r(m_j)q \tag{22}$$

is recomputed using the current intent. Shared therefore preserves a common normalized correction while adapting its absolute scale to the evolving latent state.

C.2 Temporal Intervention Classes

For the intervention-class analysis below, $q_j \in \mathbb{R}^{d_m}$ denotes the effective normalized correction after projection, satisfying $r(q_j) \le \rho$. Let $\mathbf{0} \in \mathbb{R}^{d_m}$ denote the zero correction. For $J = 5$, define

$$\begin{aligned}
\mathcal{Q}_{\text{Single}}(\rho) &= \{(q, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}) : r(q) \le \rho\}, \\
\mathcal{Q}_{\text{Shared}}(\rho) &= \{(q, q, q, q, q) : r(q) \le \rho\}, \\
\mathcal{Q}_{\text{2G}}(\rho) &= \left\{(q^{(1)}, q^{(1)}, q^{(2)}, q^{(2)}, q^{(2)}) : r(q^{(1)}) \le \rho,\ r(q^{(2)}) \le \rho\right\}, \\
\mathcal{Q}_{\text{Dynamic}}(\rho) &= \{(q_0, q_1, q_2, q_3, q_4) : r(q_j) \le \rho \text{ for every } j\}.
\end{aligned} \tag{23}$$

Proposition 2 (Containment of temporal classes). *For every $\rho > 0$, the intervention classes satisfy*

$$\mathcal{Q}_{\text{Shared}}(\rho) \subseteq \mathcal{Q}_{\text{2G}}(\rho) \subseteq \mathcal{Q}_{\text{Dynamic}}(\rho), \qquad \mathcal{Q}_{\text{Single}}(\rho) \subseteq \mathcal{Q}_{\text{Dynamic}}(\rho). \tag{24}$$

In general, $\mathcal{Q}_{\text{Single}}(\rho)$ and $\mathcal{Q}_{\text{Shared}}(\rho)$ are incomparable, as are $\mathcal{Q}_{\text{Single}}(\rho)$ and $\mathcal{Q}_{\text{2G}}(\rho)$.

Proof. Setting $q^{(1)} = q^{(2)} = q$ recovers every Shared schedule from the 2-Group class. Every 2-Group schedule belongs to the Dynamic class by choosing

$$q_0 = q_1 = q^{(1)}, \qquad q_2 = q_3 = q_4 = q^{(2)}.$$

Every Single schedule belongs to the Dynamic class by choosing

$$(q_0, q_1, q_2, q_3, q_4) = (q, \mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{0}).$$

A nonzero Single schedule is not Shared because its later calls receive zero correction. Conversely, a nonzero Shared schedule is not Single because it remains active after the first call. The same argument establishes the incomparability between Single and 2-Group. □

The class definitions induce two different controlled comparisons:

- **Temporal persistence:** Single and Shared each optimize one d_m-dimensional variable but differ in whether the intervention persists across execution calls.
- **Temporal freedom:** Shared, 2-Group, and Dynamic expose d_m, $2d_m$, and $5d_m$ scalar degrees of freedom, respectively, while remaining active throughout execution.

Single should therefore not be interpreted as a subset of Shared. The valid nested temporal-freedom comparison is the first containment chain in Eq. 24.

C.3 Reachability Is Monotone under Class Inclusion

For evaluation state x, define the successful-intervention set

$$\mathcal{S}_x = \{\mathbf{q} : S(x;\mathbf{q}) = 1\}. \tag{25}$$

For any admissible intervention class $\mathcal{Q}$, define its state-wise reachability indicator as

$$R_{\mathcal{Q}}(x) := \mathbb{I}[\mathcal{S}_x \cap \mathcal{Q} \neq \varnothing] = \mathbb{I}[\exists\, \mathbf{q} \in \mathcal{Q} \text{ such that } S(x;\mathbf{q}) = 1]. \tag{26}$$

For the temporal parameterization M used in the main paper,

$$R_M(x;\rho) = R_{\mathcal{Q}_M(\rho)}(x). \tag{27}$$

Proposition 3 (Monotonicity of reachability)**.** *If $\mathcal{Q}_1 \subseteq \mathcal{Q}_2$, then for every state x,*

$$R_{\mathcal{Q}_1}(x) \leq R_{\mathcal{Q}_2}(x). \tag{28}$$

Consequently,

$$R_{\mathcal{Q}_{\mathrm{Shared}}(\rho)}(x) \leq R_{\mathcal{Q}_{\mathrm{2G}}(\rho)}(x) \leq R_{\mathcal{Q}_{\mathrm{Dynamic}}(\rho)}(x), \qquad R_{\mathcal{Q}_{\mathrm{Single}}(\rho)}(x) \leq R_{\mathcal{Q}_{\mathrm{Dynamic}}(\rho)}(x). \tag{29}$$

Proof. If $R_{\mathcal{Q}_1}(x) = 1$, there exists $\mathbf{q}^\star \in \mathcal{Q}_1$ such that $S(x;\mathbf{q}^\star) = 1$. Because $\mathcal{Q}_1 \subseteq \mathcal{Q}_2$, the same intervention belongs to $\mathcal{Q}_2$, and hence $R_{\mathcal{Q}_2}(x) = 1$. □

Remark 1 (Empirical reachability is a lower bound)**.** *Proposition 3 concerns the underlying intervention classes, not a particular optimizer. A real closed-loop success certifies reachability. Failure of finite-budget search establishes only that the chosen procedure did not discover a successful intervention.*

C.4 Searchability Need Not Be Monotone

Class inclusion guarantees monotone reachability but does not guarantee monotone finite-budget searchability. For an admissible intervention class $\mathcal{Q}$, let

$$\widehat{\mathbf{q}}_{A,B,\mathcal{Q}}(x) = A_B(x;\mathcal{J},\mathcal{Q}) \tag{30}$$

denote the correction schedule returned by optimizer A, using frozen objective $\mathcal{J}$, after B objective evaluations.

Let $L \geq 1$ and $B \geq 1$. Consider the finite intervention classes

$$\mathcal{Q}_1 = \{\mathbf{q}^\star\}, \qquad \mathcal{Q}_2 = \{\mathbf{q}^\star, \mathbf{q}_1, \ldots, \mathbf{q}_L\}, \qquad \mathcal{Q}_1 \subset \mathcal{Q}_2, \tag{31}$$

and suppose that only $\mathbf{q}^\star$ is successful. For independent uniform sampling over a class, define

$$p_B(\mathcal{Q};x) := \Pr_A\left[S(x;\widehat{\mathbf{q}}_{A,B,\mathcal{Q}}(x)) = 1\right], \tag{32}$$

where the probability is over the sampling randomness of A. If the optimizer returns $\mathbf{q}^\star$ whenever it is sampled, then

$$p_B(\mathcal{Q}_1;x) = 1, \qquad p_B(\mathcal{Q}_2;x) = 1 - \left(1 - \frac{1}{L+1}\right)^B < 1. \tag{33}$$

This construction is illustrative rather than a model of CEM. CEM adapts its proposal using objective values, and its behavior also depends on objective geometry, initialization, and update dynamics. The relevant conclusion is only that

$$\mathcal{Q}_1 \subseteq \mathcal{Q}_2 \;\not\Longrightarrow\; \Pr_A\left[S(x;\widehat{\mathbf{q}}_{A,B,\mathcal{Q}_1}(x)) = 1\right] \leq \Pr_A\left[S(x;\widehat{\mathbf{q}}_{A,B,\mathcal{Q}_2}(x)) = 1\right]. \tag{34}$$

The relative searchability of Shared, 2-Group, and Dynamic must therefore be measured empirically under matched optimizers and objective-evaluation budgets.

C.5 Non-Unique Teachers and Amortization

Search may return different successful corrections for the same control instance. Let $\tau_x(\mathbf{q})$ denote the real closed-loop behavior induced by correction schedule $\mathbf{q}$ from instance x, such as its executed action sequence or physical trajectory. Define behavioral equivalence by

$$\mathbf{q} \sim_x \mathbf{q}' \quad \Longleftrightarrow \quad \tau_x(\mathbf{q}) = \tau_x(\mathbf{q}'). \tag{35}$$

A successful behavior may therefore correspond to multiple latent corrections rather than a unique point target.

Proposition 4 (Successful teachers need not yield a successful point-regression target)**.** *There exist state-conditional teacher distributions for which every observed teacher is successful, while the minimizer of squared correction-regression loss is unsuccessful.*

Proof. Consider a one-dimensional correction with two equally likely successful teachers,

$$q^\star = -a \qquad \text{and} \qquad q^\star = +a, \qquad 0 < a \le \rho.$$

Suppose corrections in a neighborhood of zero are unsuccessful. The corresponding squared-regression risk is

$$\mathcal{R}_{\mathrm{reg}}(q) = \frac{1}{2}(q+a)^2 + \frac{1}{2}(q-a)^2. \tag{36}$$

Its unique minimizer is the conditional mean $q = 0$, which is unsuccessful by construction. Thus, access to successful but multimodal teachers does not guarantee that point regression produces a successful intervention. □

Proposition 4 does not establish that amortization is impossible. It shows only that multimodal successful teachers may fail to define a regression-compatible point target. TIDE therefore student-unrolls the proposer through the frozen execution system and matches validated behavior rather than relying only on raw correction regression. Whether this behavior-aware objective generalizes remains an empirical question evaluated through search-free closed-loop execution.

C.6 Paired Closed-Loop Accounting

For a fixed evaluation population of size N, let $(n_{00}, n_{01}, n_{10}, n_{11})$ denote the resolved paired outcomes, where the first index corresponds to Direct execution and the second to treatment. If every unit is resolved,

$$\mathrm{SR}_{\mathrm{Direct}} = \frac{n_{10} + n_{11}}{N}, \qquad \mathrm{SR}_{\mathrm{Treatment}} = \frac{n_{01} + n_{11}}{N}. \tag{37}$$

Proposition 5 (Net paired success-rate change)**.** *For a fully resolved population,*

$$\mathrm{SR}_{\mathrm{Treatment}} - \mathrm{SR}_{\mathrm{Direct}} = \frac{n_{01} - n_{10}}{N}. \tag{38}$$

Proof. Using Eq. 37, the common-success term n_{11} cancels:

$$\frac{n_{01} + n_{11}}{N} - \frac{n_{10} + n_{11}}{N} = \frac{n_{01} - n_{10}}{N}.$$

□

If U units are unresolved but remain in the fixed denominator N, each unresolved unit can change the paired numerator by at most one in either direction. Therefore,

$$\Delta_{\mathrm{paired}} \in \left[\frac{n_{01} - n_{10} - U}{N}, \frac{n_{01} - n_{10} + U}{N} \right]. \tag{39}$$

This interval separates unresolved evidence from observed failure and prevents identity or execution errors from being interpreted as scientific outcomes.

Table F: **Multi-seed stability of the aligned Shared search funnel.** Shared search is repeated with three independent search seeds on the same frozen 64-unit Direct-failure cohort for each task–backbone pair. Each unit receives 2,048 objective evaluations. Entries report confirmed rescues out of 64; all units are resolved.

Task	Backbone	Seed 1	Seed 2	Seed 3	Mean ± SD
Push-T	LeWM	16/64	16/64	16/64	16.0 ± 0.0
	Fast-LeWM	22/64	21/64	21/64	21.3 ± 0.6
TwoRoom	LeWM	11/64	9/64	12/64	10.7 ± 1.5
	Fast-LeWM	13/64	13/64	13/64	13.0 ± 0.0

C.7 Scope of the Formal Results

The preceding results establish that:

1. relative-RMS projection guarantees that every applied intervention satisfies the prescribed scale bound;
2. enlarging a temporal intervention class cannot reduce existential reachability;
3. class inclusion provides no monotonic guarantee for finite-budget searchability;
4. successful multimodal teachers need not yield a successful point-regression target; and
5. offline learning quality must be distinguished from paired, search-free closed-loop improvement.

These statements justify TIDE's decomposition into reachability, searchability, and learnability. They do not prove that Dynamic interventions are universally harder to optimize, that Shared corrections are universally preferable, or that temporal interventions cannot be amortized by other model classes.

D Additional Robustness and Representation Breadth

We provide two extensions of the main evaluation: a high-budget stability check for Shared search and an evaluation with C-JEPA representations. Both retain frozen populations, real closed-loop validation, and fixed-denominator accounting.

D.1 High-Budget Shared Stability

Complementing the finite-budget multi-seed analysis in the main paper, we repeat Shared search with three independent seeds on the same frozen 64-unit Direct-failure cohort. Each unit receives 2,048 objective evaluations. As shown in Tab. F, all conditions are fully resolved and rescue counts vary by at most three across seeds. The reported standard deviations measure optimizer-seed variation rather than sampling uncertainty.

D.2 Representation Breadth with C-JEPA

We test whether Shared search extends beyond the LeWM family by equipping C-JEPA with the same standardized execution-head architecture and training protocol. We use the released Push-T checkpoint and train task-specific TwoRoom dynamics on train-side data with a frozen visual tokenizer. All components are frozen and pass causal-twin checks before evaluation. Under the same 256-evaluation Shared protocol, C-JEPA obtains $4[4, 6]$ rescues on Push-T and $15[15, 17]$ on TwoRoom (Tab. G). Because failure cohorts are frozen separately within each backbone, these results establish representation breadth rather than a direct backbone ranking.

Table G: **Representation breadth under the standardized TIDE interface.** Direct is evaluated on the complete 750-unit development population. Shared uses 256 objective evaluations on a separately frozen 64-unit cohort drawn exclusively from each backbone's Direct failures. Entries $r[L, U]$ report confirmed rescues and fixed-denominator bounds under unresolved paired units. Cohorts are frozen within backbone and are not cross-backbone paired.

Task	Frozen backbone	Direct	Resolved	Shared rescues
Push-T	LeWM	520/750	64/64	$13[13, 13]$
	Fast-LeWM	469/750	64/64	$16[16, 16]$
	C-JEPA	384/750	62/64	$4[4, 6]$
TwoRoom	LeWM	394/750	64/64	$9[9, 9]$
	Fast-LeWM	449/750	64/64	$11[11, 11]$
	C-JEPA	626/750	62/64	$15[15, 17]$

E LIMITATIONS

TIDE provides controlled empirical evidence rather than a complete characterization of executable behavior. A validated intervention certifies reachability, but failure under any finite optimization procedure cannot establish that no successful intervention exists. Accordingly, our reachability results should be interpreted as validated lower bounds. Our conclusions are also conditioned on the evaluated tasks, frozen representations, standardized execution interface, and intervention radius. Searchability is measured using finite-budget CEM, while learnability is evaluated using a standardized Shared-correction proposer. Other optimizers, budgets, or amortization architectures may yield different tradeoffs. Moreover, cross-backbone amortization is evaluated only for cells that satisfy the preregistered minimum number of validated teachers; the Cube cells remain protocol-ineligible and therefore do not support a learnability conclusion. Broader representation families, longer-horizon or stochastic environments, and alternative search and amortization mechanisms are needed to determine how generally the observed separation between reachability, searchability, and learnability persists.

F USE OF LARGE LANGUAGE MODELS

A large language model (LLM) was used solely for language-level assistance, such as improving readability, fluency of the text and formatting LaTeX tables and retrieve related works. The research ideas, experiments, and results are entirely the work of the authors, who bear full responsibility for the content of this submission.