A PERFECT MODEL IS NOT ENOUGH: MEASURING AND LOCATING LONG-HORIZON FAILURE IN LEWORLDMODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

We study what limits long-horizon planning in JEPA world models through a case study on LeWorldModel (LeWM), a compact world model trained end-to-end offline on pixels whose evaluation protocol has become standard in subsequent work. Several of LeWM's successors attribute its failures at long horizons to compounding rollout error. We first show that extensions of the standard protocol that simply increase the goal offset cannot test this claim: seemingly minor details of several environments and datasets make them poor tests of long-horizon planning at larger goal offsets. On environments that avoid these pitfalls, we evaluate LeWM on held-out episodes across increasing start-goal offsets, varying its rollout source and terminal cost separately. Our results show that (a) planning with a perfect model (by executing each candidate in the simulator) improves long-horizon success only modestly, while (b) replacing only the Euclidean terminal cost with a learned cost-to-go can achieve equivalent or greater outcomes without retraining the world model. At long horizons, the predictor is therefore not the bottleneck. These insights suggest that evaluation protocol and planner costs inherited by follow-up work deserve as much scrutiny as the predictive world model itself.

1 INTRODUCTION

World models trained offline on pixels have become a promising route to reward-free goal-conditioned control: given an image of the goal, a planner searches for actions whose predicted outcome lies close to the goal in embedding space. LeWorldModel (LeWM) (Maes et al., 2026a), which trains a compact JEPA-based world model on a single GPU, has made this recipe widely accessible and has become the base for a large body[1] of follow-up work. However, despite these successes, planning with these models remains confined to short horizons (Maes et al., 2026a;c).

Responses to this limitation have included hierarchical planners, variable-length predictors, and reshaped latent geometries (Masip et al., 2026; Cheng et al., 2026; Du et al., 2026), with several successors attributing long-horizon failures to compounding rollout error (Du et al., 2026; Gao & Xu, 2026), latent geometries that are predictive but not plannable (Li et al., 2026b), or difficulty in optimizing long action sequences (Cheng et al., 2026). The variety of responses and protocol extensions[2] points to a broader problem: we do not know to what extent limitations of the planning pipeline should be attributed to the model, the planner, or the task.

Our contributions. We study LeWM by measuring the properties of its chosen environments. We perturb these properties and measure the effects on success rates across different goal offsets. Accounting for these effects, we then ask where progress would matter most: we simulate a perfected predictor (by executing candidate actions directly in environments, then encoding the resulting latents) and an improved cost (by replacing Euclidean distance with a learned cost-to-go) and compare their effects on long-horizon success alone and combined.

[1] 195 citing works on Semantic Scholar as of Sept. 24, 2026 (Appendix A.1).

[2] At least 33 successors use LeWM's protocol, of which 9 extend it beyond $\Delta = 25$ (Appendix A.2).

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

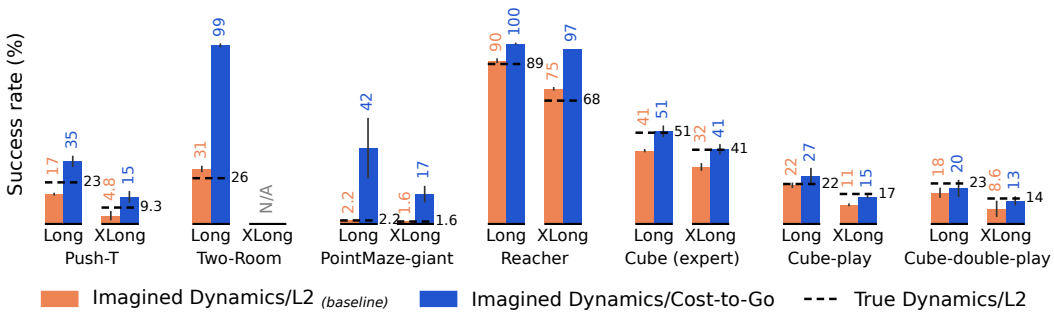

Figure 1: A learned cost can help more than a rollout error-free predictor. Success at goal offsets $\Delta = 75$ (Long) and $\Delta = 125$ (XLong) steps. Orange: LeWM with latent L2 cost. Blue: learned cost-to-go on frozen embeddings. Dashed: latent L2 cost with rollouts executed in the simulator and states re-encoded. Blue exceeds dashed in **10 of 13** settings. In contrast, a perfect predictor provides only modest benefits. Bars: mean over three backbone seeds; error bars: SD across seeds, which measures variation between training runs (all seeds plan on the same tasks). Two-Room episodes are too short for XLong.

We find that on several of these environments a longer goal offset does not make a longer task: tasks start already solved, demonstrations reach their goals early and then rest, or the offset reflects how long the demonstrator took rather than how far the goal is. Our results show that at long horizons, even a perfect predictor helps only modestly and can even hurt, while the learned cost can match or exceed the perfect predictor's success rates, even though the comparison is tilted toward the predictor (the dynamics of the environment itself are the best any predictor could do, while our learned cost is a diagnostic probe with known limitations). Therefore, **we overturn the common notion that rollout error is the bottleneck for long-horizon failure on LeWM**. Fixing the cost first also makes improvements to the predictor worth more, suggesting that despite not being the bottleneck, changes in rollout error can compound with other improvements.

These observations emphasize the importance of treating the planning objective and the evaluation as peer components to be studied. A new predictor or training objective tells us less about long-horizon planning when it plans with an anemic cost and tests on environments a single plan can reach. Inspirations from goal-conditioned RL offer future candidate costs (quasimetrics) and environments alike. By these measures, the gap between today's JEPA world models and long-horizon planning may be smaller than it looks.

2 Background

JEPA world models. Our analysis concerns world models based on the joint-embedding predictive architecture (JEPA), where an encoder f_θ maps an observation o to a latent z and a predictor p_ϕ predicts the next latent given an action. Unlike generative world models, JEPAs are trained entirely in latent space (LeCun, 2022; Assran et al., 2023), without reconstructing pixels or predicting rewards:

$$z_t = f_\theta(o_t) \qquad \hat{z}_{t+1} = p_\phi(z_t, a_t) \qquad (o_t, a_t, o_{t+1}) \sim \mathcal{D}$$
$$\mathcal{L}(\theta, \phi) = \|\hat{z}_{t+1} - f_\theta(o_{t+1})\|^2 + \lambda \mathcal{R}(z) \tag{1}$$

LeWM uses SIGReg (Balestriero & LeCun, 2025), which matches the distribution of embeddings to an isotropic Gaussian, and trains f_θ and p_ϕ end-to-end from pixels with regularizer $\mathcal{R}$ preventing trivial encoders (Maes et al., 2026a). We train LeWorldModel according to hyperparameters (appendix zyz) and freeze the encoder and predictor for the remainder of our study.

Planning with world models. At inference time, the world model is used for goal-conditioned planning by model predictive control (Mayne et al., 2000). Given a start observation o_0 and a goal image o_g, the planner rolls candidate action sequences through the predictor and minimizes a

terminal cost C on the predicted latent:

$$\begin{aligned} \hat{z}_0 = f_\theta(o_0) \qquad \hat{z}_{k+5} = p_\phi(\hat{z}_k, a_{k:k+4}) \qquad z_g = f_\theta(o_g) \\ a^*_{0:L-1} = \arg\min_{a_{0:L-1}} C(\hat{z}_L, z_g) \qquad C_{\mathrm{L2}}(\hat{z}_L, z_g) = \|\hat{z}_L - z_g\|^2 \end{aligned} \tag{2}$$

Without rewards, a JEPA world model has no value function. The field defaults to planning with Euclidean distance as the terminal objective (Zhou et al., 2024; Sobal et al., 2025; Assran et al., 2025; Maes et al., 2026a), a common convention in planning from pixels (Watter et al., 2015; Srinivas et al., 2018). The minimization uses the cross-entropy method (CEM), a sampling-based optimizer, and replans after executing each full plan. The predictor advances 5 environment steps per call, so one plan looks $L = 25$ environment steps ahead in 5 calls.

Terminal costs and cost-to-go. Because each plan looks only L steps ahead, the terminal cost must account for everything beyond the lookahead. The ideal terminal cost is the cost-to-go: the number of steps still needed to reach the goal from where the plan ends. World models trained with rewards, such as TD-MPC2, learn it as a value function. Offline goal-conditioned RL learns such a quantity without rewards, by treating future states of each trajectory as goals and fitting a goal-conditioned value by temporal-difference learning, where $\mathrm{sg}(\cdot)$ stops the gradient:

$$\begin{aligned} (z_t, z_{t+1}, z_g) \sim \mathcal{D} \qquad g \geq t \\ \mathcal{L}(\psi) = \Big(V_\psi(z_t, z_g) - \big[1 + \mathrm{sg}\big(V_\psi(z_{t+1}, z_g)\big)\big] \Big)^2 \qquad V_\psi(z_g, z_g) = 0 \end{aligned} \tag{3}$$

Inspired by Li et al. (2026a) who trains a value network offline on the dataset to = direct temporal distance, we train V_ψ according to appendix abc. Consequently, it learns to estimate $V^\beta(s, g)$, the expected number of steps β takes to reach g, rather than the shortest distance. It therefore overestimates distances where demonstrations detour.

LeWM evaluations. Planning for LeWM and derivative work is evaluated almost universally[3] with a protocol introduced by DINO-WM (Zhou et al., 2024) for Push-T and standardized by LeWM (Maes et al., 2026a) and the stable-worldmodel library (Maes et al., 2026c). Given a set $\mathcal{D}_{\text{eval}}$ of episodes $\tau = (s_0, o_0, a_0, \ldots, s_T, o_T)$ with simulator states s and images o, a task is formed by sampling an episode and a start time, and taking the frame Δ environment steps later as the goal:

$$\tau \sim \mathcal{D}_{\text{eval}}, \qquad t \sim \mathcal{U}\{0, \ldots, T - \Delta\}, \qquad (s_0, o_0) = (s^\tau_t, o^\tau_t), \qquad (s_g, o_g) = (s^\tau_{t+\Delta}, o^\tau_{t+\Delta}). \tag{4}$$

The planner sees only the images o_0 and o_g and acts in the simulator for a budget of $B = 50$ steps. The task succeeds if the simulator state satisfies an environment-specific test at any step $k \leq B$. LeWM draws $n = 50$ tasks and counts their average success rate. Under LeWM, $\mathcal{D}_{\text{eval}}$ is equivalent to $\mathcal{D}_{\text{train}}$, which we show has consequences (Table 1).

3 Evaluation Protocols

Our objective is to better understand the long-horizon behavior of LeWM planning, which may be influenced not only by the quality of a model or planner but also by choices in the dataset or protocols used. We find that LeWM's default environments and protocols contain certain choices with consequences for measured success rates:

1. **Environments can start with pre-solved tasks:** Because task success is counted for any step where success criteria are met, tasks that start in a successful state inflate a planning success rate p to $\hat{p} = \pi + (1 - \pi)\, p$ where π is the percentage of selected tasks that start solved. This choice primarily impacts Cube (expert), inflating its score by 24pp (Table 1). To account for this, we exclude these tasks and report the success rate over the remaining tasks that do not start solved.
2. **Task count:** LeWM uses $n = 50$ tasks per environment. A success rate p measured on n tasks has standard error $\sqrt{p(1-p)/n}$, so at $p \approx 0.8$, two methods must differ by about 16 points to be told apart. LeWM's 96% on Push-T is 14.5 points above the same checkpoint

[3] See Appendix A.2

on 200 tasks (Table 1). The authors' own released evaluator gives 92% on its default 50 tasks (Appendix E.4), so about 10 of these points come from the task draw and about 4 we could not reproduce. We use $n = 200$, which halves this margin.

3. **Choice of training episodes:** LeWM evaluates on the episodes its model trained on. On Push-T, whose 18,685 episodes share 185 initial configurations, this raises success from 76.5% to 87.0% (Table 1); elsewhere the change is within noise. We evaluate on a held-out 20% of initial-state groups.

Table 1: We ablate LeWM's protocols across environments by changing one thing from left to right. All checkpoints were trained on seed 3072. "Reported" is taken directly from LeWM Figure 6 (cite). "Official" is LeWM's released checkpoints evaluated on 200 tasks. "Ours" is our backbone evaluated on training, then held-out, episodes, which shows the impact of evaluating on training data. "Unsolved" further excludes pre-solved tasks. Brackets: 95% bootstrap CIs over source episodes (10,000 resamples).

	Success (%)					
	Reported n=50, 1 seed	Official n=200, 1 seed	Ours, train n=200, 1 seed	Ours, held-out n=200, 1 seed	Unsolved 1 seed	Solved at start of 200
Two-Room	87	82.5 [77.0, 87.6]	87.0 [82.2, 91.5]	88.5 [83.8, 92.9]	87.6 [82.5, 92.3]	14
Push-T	96	81.5 [76.0, 87.0]	87.0 [82.0, 91.5]	76.5 [70.4, 82.4]	76.4 [70.2, 82.3]	1
Reacher	86	79.0 [73.2, 84.4]	82.5 [77.1, 87.5]	79.5 [73.9, 84.9]	79.4 [73.7, 84.8]	1
Cube (expert)	74	62.0 [55.3, 68.7]	68.5 [62.0, 74.9]	66.0 [59.6, 72.3]	41.9 [33.1, 50.9]	**83**

Does a longer offset make a longer task? We now hold LeWM's protocols with improved accounting and investigate extending Δ beyond 25 to study long horizon planning (similar to our predecessors in Appendix A.2). We extend the protocol naively by evaluating with $\Delta = 75$ (which we call LONG) and $\Delta = 125$ (which we call XLONG) and set the planning budget $B = 2\Delta$[4] matching LeWM's doubled budget for $\Delta = 25$.

Loosely, consider a task horizon as the true number of steps required to reach it. For the purposes of evaluating planning within this paper, we call a task long horizon if the true number of steps to reach it is greater than the planner lookahead L.[5] One flaw in our naive extension is the assumption that a goal sampled farther along in an episode takes longer to reach, but this is not necessarily true. This construction only guarantees that the demonstrator reached it within Δ steps.

We bracket the true number of steps a task needs, h^*. From above: the demonstration first satisfied the success test after k^* steps, so no task needs more. From below: no policy can need fewer than h_{lb}, the task-relevant start-to-goal state difference divided by the largest distance the state can move in one step. So

$$h_{\text{lb}} \;\le\; h^* \;\le\; k^* \;\le\; \Delta. \tag{5}$$

With one plan looking $L = 25$ steps ahead, a task is *known short* if $k^* \le L$, since the demonstrator reached the goal within one plan, and *known long* if $h_{\text{lb}} > L$, since no single plan can. Every task at $\Delta = 25$ is known short. A dataset tests long horizons only if larger offsets produce known-long tasks. We investigate properties of environments and datasets that may affect the production of known-long tasks, though unlike the protocol choices, these may be harder to account for. Below, (row, column) refers to a cell of Table 5.

1. **Short episodes:** A task needs $t + \Delta \le T$, the episode length. The chain still holds for every task, but larger offsets draw on fewer episodes and later goals, so it describes a different set of tasks at each offset. Two-Room cannot host $\Delta = 125$ (Two-Room, Eligible), and Push-T's $\Delta = 125$ tasks come from 15 of its 37 held-out layouts (Push-T, Layouts), with unchanged success, so the result is narrower but not biased. On 201-step episodes, a $\Delta = 125$ goal falls in the last 76 steps, where item 2 applies.

[4] We ablate this choice in Appendix E.3

[5] we say this because after a certain point increasing L worsens performance, see appendix and also cite that point this out, so the need to plan when L¡h* is important

2. **Resting tails:** Once $t+\Delta$ passes the step τ at which the demonstrator stops, $k^* \leq \tau - t$ for any Δ, so $h^* \leq k^*$ stops growing with the offset. Cube (expert) rests for a quarter of every episode (Cube (expert), Tail): its k^* falls from 45 to 32 steps between $\Delta = 75$ and 125 (Cube (expert), k^*), half its tasks stay known short (Cube (expert), Short), and L2 solves 45% of tasks whose goal rests against 9% of the rest. Cube-play, the same task, rarely rests (Cube-play, Tail).
3. **Solved at start:** Here $h^* = k^* = 0$ at any offset. Resting keeps such tasks around: 16% of Cube (expert)'s tasks still start solved at $\Delta = 125$ (Cube (expert), Solved), against under 3% on play data from $\Delta = 75$ (Cube-play, Solved). We exclude them at every offset.
4. **Random demonstrations:** k^* is only an upper bound, and it says nothing when the demonstrator wanders. Reacher's actions are random (Reacher, Lag-1), so $k^* = \Delta$ exactly (Reacher, k^*), while the goal distance grows only as $\sqrt{\Delta}$, $2.2\times$ for $5\times$ the offset (Reacher, Dist.). Its h^* therefore grows far more slowly than k^*. PointMaze-giant's goal distance keeps pace with the offset (PointMaze-giant, Dist.).
5. **Loose lower bounds:** Only $h_{\mathrm{lb}} > L$ proves a task long, so a dataset needs a lower bound that grows with Δ. PointMaze-giant's shortest path through the maze does (PointMaze-giant, h_{lb}), and 81% and 85% of its tasks are known long (PointMaze-giant, Long). For manipulation, h_{lb} counts only how far objects travel, not the reaching and grasping needed to move them, and stays at 1 to 6 steps (Cube (expert), h_{lb}), so no task can be proven long even when it is.

Because its 2,001-step episodes leave room for large offsets, its demonstrator heads for goals without resting, and its walls make the maze's shortest path a lower bound that grows with the offset, we recommend PointMaze-giant, and environments at a similar scale such as OGBench's other giant mazes, for evaluating long-horizon planning.

The five items double as a checklist. Among LeWM's datasets, Two-Room and Push-T fail item 1, Cube (expert) items 2 and 3, and Reacher item 4; none has a task proven longer than one plan. A dataset for long-horizon evaluation needs episodes far longer than the largest offset, demonstrations that keep moving and head for their goals, and a lower bound on h^* that grows with Δ, which in practice requires a path distance such as a maze's shortest path. PointMaze-giant passes all five. Cube-play and Cube-double-play pass the first four, so their long offsets are plausibly but not provably long. We use these three OGBench datasets (Park et al., 2025) and keep LeWM's four for comparison.

4 Experimental Results

We now evaluate the performance of LeWM according to our longer horizon protocols, accounting for multiple seeds, increased episode counts, equivalent planning budgets, and held out training data. We do not account for resting tails, or random demonstrations on environments as we are limited in environment modification budget but instead choose to call attention to the issues with specific environments mentioned in the previous section. We recommend attention to PointMaze as adequate in our eyes for measuring long horizon performance. However, all other environments remain attached for comparison.

Is the planning horizon the bottleneck? If the goal lies beyond one plan, the obvious fix is to plan further. Holding everything else fixed, we lengthen the lookahead from $L = 25$ to up to 125 environment steps (Figure 3). In 13 of 14 environment and protocol settings, no longer horizon does significantly better, and success usually falls: Push-T SHORT drops from 79.2% to 18.1% at $L = 50$. The one gain, Two-Room LONG at $L = 50$ (30.6% to 39.4%), is gone at $L = 75$. CEM keeps the same number of samples at every L, and sampling-based optimizers need rapidly more samples as the plan grows (Pinneri et al., 2020), so a longer lookahead mostly makes the search harder.

Is the planning budget the bottleneck? A planner may also simply run out of steps. We rerun LONG with budgets of Δ, 2Δ (our default) and 3Δ (Appendix E.3). Raising the budget to 3Δ changes success by at most 3 points.

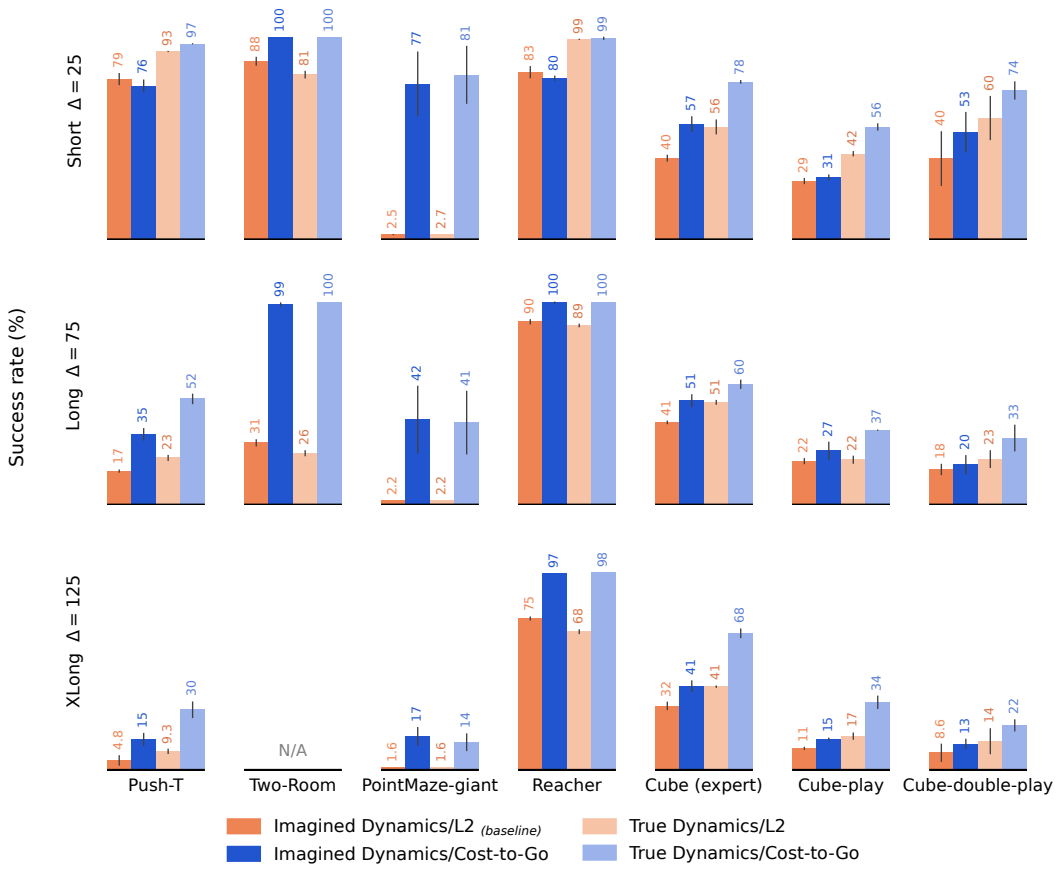

Figure 2: Success of all four planner conditions, imagined or true dynamics with the L2 or learned cost-to-go, for every environment and protocol. Bars: mean over three backbone seeds on the same tasks, with tasks solved at the start excluded; error bars: SD across seeds, which measures variation between training runs. Two-Room has no XLONG.

Is the predictor the bottleneck? LeWM's successors attribute long-horizon failure to compounding rollout error. We test this by giving the planner a perfect predictor: every candidate action sequence is executed in the true simulator from the current state, and the resulting frame is encoded with LeWM's encoder. The planner then scores exactly the latent a perfect predictor would produce, with the same cost and CEM settings. This helps only modestly (Table 8): at LONG and XLONG, L2 success rises by at most 6.5 points, stays at the floor on PointMaze-giant (2.2% and 1.6%), and falls on Two-Room LONG and on Reacher.

Is the cost the bottleneck? Replacing only the Euclidean terminal cost with the learned cost-to-go, and keeping LeWM's own predictor, changes far more. At LONG, Two-Room rises from 30.6% to 98.8%, PointMaze-giant from 2.2% to 42.1%, and Push-T doubles (16.7% to 34.8%). PointMaze-giant shows why. Its walls force detours, so a state can be close to the goal in a straight line but far from it along the maze. Latent L2 correlates with shortest-path distance for pairs 25 steps apart (Spearman $\rho = 0.50$) but not for random pairs ($\rho = 0.01$), so beyond one lookahead it gives almost no direction, and a perfect predictor only finds the dead ends more accurately. The learned cost is trained on how many steps the demonstrations took, and those demonstrations went around the walls.

The cost matters more than the predictor. At long offsets, the learned cost with LeWM's predictor beats L2 with the perfect predictor in 10 of 13 settings (Figure 1), although the comparison favors the predictor. The three exceptions, Cube-play XLONG and Cube-double-play LONG and

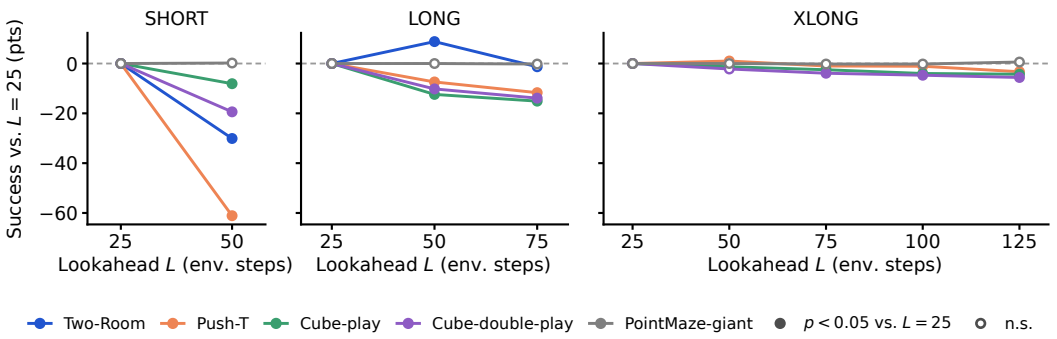

Figure 3: Change in L2 success, relative to $L = 25$, as the lookahead L (environment steps) grows, with imagined dynamics. Lines are means over three backbone seeds on the same tasks. Filled markers differ from $L = 25$ at $p < 0.05$ (task-clustered sign-flip test across seeds). Only Two-Room LONG improves, at $L = 50$, and it returns to baseline at $L = 75$. Full results are in Table 10 (Appendix E.1).

XLONG, are on the datasets whose long offsets we could not prove long (Section 3). With both the cost and the predictor fixed, success is highest in 11 of the 13 settings, so a better cost also makes a better predictor worth more.

Short offsets. At SHORT the pattern reverses. The perfect predictor helps L2 (Push-T 79.2% to 92.8%, Reacher 82.6% to 98.8%), and the learned cost sometimes does worse than L2 (Push-T 75.9%, Reacher 79.6%). Every SHORT task is known short, so these tasks fit within one plan and do not test the regime we study. One possible explanation is that within one lookahead the planner only needs to match the goal image, which latent L2 does well and exact dynamics do better, while the learned cost estimates the demonstrator's step count, which carries little information for goals only a few steps away. Curiously, the perfect predictor also lowers L2 success on Two-Room SHORT (88.0% to 81.4%), as it does on Two-Room LONG and Reacher LONG and XLONG. We have no explanation and leave it to future work.

5 Related Work

Attributing planning failure to the predictor. Model-based RL has long treated compounding multi-step model error as the main obstacle to long-horizon planning, and fixes it on the predictor side: training on the model's own rollouts (Venkatraman et al., 2015; Talvitie, 2017), constraining its smoothness (Asadi et al., 2018), or shortening rollouts (Janner et al., 2019). LeWM's successors make the same diagnosis and replace recursive one-step rollouts with variable-length jumps (Du et al., 2026; Gao & Xu, 2026) or hierarchical subgoals (Masip et al., 2026; Cheng et al., 2026), while keeping LeWM's Euclidean cost. Yet lower model error need not mean better control (Lambert et al., 2020), and MPC with ground-truth dynamics also degrades as the horizon grows (Wang et al., 2019). We test the diagnosis directly by swapping in the true simulator as a perfect predictor.

What the planner optimizes beyond its lookahead. Latent planners from early pixel-based control (Watter et al., 2015; Srinivas et al., 2018) to current JEPA world models (Zhou et al., 2024; Sobal et al., 2025; Assran et al., 2025; Maes et al., 2026a) assume that distance to the goal embedding is an adequate terminal cost. Control and model-based RL instead close the gap beyond a finite lookahead with a terminal value (Mayne et al., 2000), learned from rewards by temporal-difference learning (Lowrey et al., 2019; Hansen et al., 2022; 2024); offline goal-conditioned RL learns its reward-free analogue, a temporal distance, from unlabeled trajectories (Eysenbach et al., 2022; Wang et al., 2023; Park et al., 2023), and Li et al. (2026a) use one to re-rank a frozen LeWM planner. Lengthening the lookahead instead is costly, because CEM needs rapidly more samples as the plan grows (Pinneri et al., 2020). We keep the world model and lookahead fixed and change only the terminal cost.

How long-horizon planning is evaluated. Latent world models are evaluated on goals taken Δ steps later in the same recorded episode (Zhou et al., 2024; Maes et al., 2026a), whereas goal-reaching benchmarks fix start and goal pairs chosen to need many steps (Fu et al., 2020; Park et al., 2025). Critiques of RL evaluation have shown that conclusions depend on seeds, on how uncertainty is reported (Henderson et al., 2018; Agarwal et al., 2021), and on implementation details (Engstrom et al., 2020). We add a check specific to offset-based goals: bounds on the steps a task needs, which can be used to identify risks to whether a larger offset yields a task longer than one plan.

6 Conclusion

We take a first step towards understanding why planning with LeWM breaks down at long horizons, where prior work suggested compounding rollout error. Before testing this claim, we had to investigate the evaluation protocols. We found often overlooked details of LeWM's datasets and protocol can decide what a larger goal offset can reveal and affect success rates by large margins.

We overturn the notion that rollout error is the current bottleneck to long-horizon planning in LeWM by ablating solutions to it with alternative directions involving the terminal cost. If Euclidean distance says almost nothing about the distance to a goal beyond the planning horizon, a better predictor cannot compensate for a cost that does not represent information past planning. We therefore recommend that JEPA planners stop using it as the terminal cost.

We do not recommend our learned cost as a replacement, since it is suboptimal and estimates the demonstrator's step count and not the shortest distance. Choosing the right cost needs proper ablation, with quasimetric objectives as natural candidates for follow up work. Furthermore, we test only with base LeWM and a single kind of planner. With more compute we would have tested DINO-WM or other world models to see if this is true in further cases. The authors hope that the great accessibility of LeWM and the insights provided by this paper may motivate exploration of more diverse environments. Of interest besides long-horizon tasks may be partial observability or multi-agent scenarios.

More broadly, our results suggest that the planning cost and the evaluation protocol deserve the same scrutiny as the world model. Future work should choose environments who demonstrate known-long tasks easily, such as PointMaze-giant, and evaluate on randomly sampled goals, which stable-worldmodel already supports. If the reuse of goals from the training demonstrations as a form of evaluation persists, releasing the policies used to collect each dataset would make these evaluations reproducible. Without this care, it is hard to tell whether a new world model actually plans further, or whether the benchmark simply cannot measure the difference.

AI use statement

In this work, we used generative AI to provide feedback on methodology or experiments, implement methods, and support qualitative and thematic data analysis. We did not use generative AI to generate synthetic datasets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, clean and reformat datasets, or interpret results, and assistance with translation was not necessary for this work. Additionally, we used generative AI to summarize or analyse existing literature, create or modify scientific figures or images, create or edit software code, brainstorming, and identify relevant literature. We have reviewed all AI-assisted work. We manually reviewed AI generated code was verified and reviewed by 2 authors. We manually verified that figures accurately used quantitative data. Throughout the paper writing, authors reviewed each other's work independently to check for inaccuracies and qualitative mistakes. The authors worked together to manually review AI generated methodologies for correctness and scientific integrity. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

We will release all code, configuration files, held-out splits, trained backbones and cost heads, and per-task evaluation records. Until then, this work describes protocols in detail. Also see appendices

(a,b,c,d,e,f,g....) on how to replicate each part of the work exactly. We train three LeWM backbones per environment (seeds 3072–3074) with the hyperparameters in Appendix **??**, and one learned cost-to-go head per backbone with a fixed seed and 25,000 updates, chosen before any held-out evaluation (Appendix C). Environments, datasets and the held-out split (20pp of initial-state groups, fixed split seed) are described in Section 3 and Appendix B. Every cell evaluates 200 tasks sampled with a fixed seed from held-out episodes, with per-task CEM seeds; all four cost$\times$dynamics conditions use identical tasks, so every comparison is paired. We report mean $\pm$ SD across backbone seeds in the main text and 95pp bootstrap intervals clustered on source episode in Appendix D, where we also give rates including tasks solved at the start.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems*, 2021.

Kavosh Asadi, Dipendra Misra, and Michael L. Littman. Lipschitz continuity in model-based reinforcement learning. In *International Conference on Machine Learning*, 2018.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael 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*, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew 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. *arXiv preprint arXiv:2506.09985*, 2025.

Jiaxin Bai and Jiaxuan Xiong. Temporal-Distance JEPA: Plan-Aware Representation Learning for Latent World Model Predictive Control. *arXiv preprint arXiv:2607.25337*, 2026a.

Jiaxin Bai and Jiaxuan Xiong. VisualPatchWorld: Code World Models as Latent Structured Representations for Planning. *arXiv preprint arXiv:2607.25236*, 2026b.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025.

Saksham Bansal, Om Naphade, Chayan Aggarwal, and Vrishin M. LePlanner: An Iterative Amortized Controller For World Models. *arXiv preprint arXiv:2609.13845*, 2026.

Jack Boylan and Chris Hokamp. No Gaussian Required: Contrastive Inverse Dynamics for JEPA World Models. *arXiv preprint arXiv:2608.17542*, 2026.

Jiaheng Chen. ATM: Action-Consistency Transfer Matrix for Diagnosing and Improving Latent World Models. *arXiv preprint arXiv:2606.09028*, 2026.

Letian Cheng, Qi Zhang, and Yisen Wang. SAGE: Subgoal-Conditioned Action Generation for Latent World Model Planning. *arXiv preprint arXiv:2607.17973*, 2026.

Tianqi Du, Qi Zhang, Yifei Wang, and Yisen Wang. Beyond the Next Step: Variable-Length Latent World Models for Long-Horizon Planning. *arXiv preprint arXiv:2606.21775*, 2026.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep policy gradients: A case study on PPO and TRPO. In *International Conference on Learning Representations*, 2020.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Ruslan Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, pp. 35603–35620, 2022. doi: 10.52202/068431-2580.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Baoqi Gao, Ruize Han, Miao Wang, and Song Wang. IMWM: Intuition Models Complement World Models for Latent Planning. *arXiv preprint arXiv:2606.01626*, 2026.

Yuntian Gao and Xiangyu Xu. Fast LeWorldModel. *arXiv preprint arXiv:2606.26217*, 2026.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*, 2022.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In *AAAI Conference on Artificial Intelligence*, 2018.

Jiaming Hu, Yan Zheng, and Tian Wang. SCALE: State-Calibrated Latent Embeddings for JEPA Planning in the Right Geometry. *arXiv preprint arXiv:2608.16287*, 2026.

Hsiang-Wei Huang, Jianxu Shangguan, Junbin Lu, and Jenq-Neng Hwang. LeFlow: Generative Latent Flow Planning for World Models. *arXiv preprint arXiv:2608.24855*, 2026a.

Xiaodi Huang, Ziyi Ding, Jingtian Wan, Yuchen Liu, Yuan Zhang, Xiao-Ping Zhang, Jiayu Chen, Zhang Zhang, and Tao Huang. Traj-LeWM: Path-Aware World-Model Planning via Latent Trajectory Cost. *arXiv preprint arXiv:2608.14125*, 2026b.

Petr Ivashkov, Randall Balestriero, and Bernhard Schölkopf. Sensorimotor World Models: Perception for Action via Inverse Dynamics. *arXiv preprint arXiv:2606.20104*, 2026.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2019.

Yilun Kuang, Yash Dagade, Quentin Le Lidec, Lucas Maes, Randall Balestriero, and Yann LeCun. LpWM: A Case for Sparse Representations in World Models. *arXiv preprint arXiv:2608.22764*, 2026.

Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. In *Learning for Dynamics and Control*, 2020.

Yann LeCun. A path towards autonomous machine intelligence. OpenReview preprint, 2022.

Liangyu Li, Shengzhi Wang, Libin Qiu, Mingliang Xiong, and Qingwen Liu. World Model Control by Trajectory Reachability Metrics. *arXiv preprint arXiv:2605.22164*, 2026a.

Wenyuan Li, Guang Li, Keisuke Maeda, Takahiro Ogawa, and Miki Haseyama. Predictive but Not Plannable: RC-aux for Latent World Models. *arXiv preprint arXiv:2605.07278*, 2026b.

Muyuan Liu, Yue Huang, Zheng Liang, and Xiang Gao. Toward Physically Grounded JEPA World Models for Goal-Conditioned Robotic Planning. *arXiv preprint arXiv:2609.03565*, 2026.

Kendall Lowrey, Aravind Rajeswaran, Sham Kakade, Emanuel Todorov, and Igor Mordatch. Plan online, learn offline: Efficient learning and exploration via model-based control. In *International Conference on Learning Representations*, 2019.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. LeWorldModel: Stable End-to-End Joint-Embedding Predictive Architecture from Pixels. *arXiv preprint arXiv:2603.19312*, 2026a.

Lucas Maes, Quentin Le Lidec, Luiz Facury, Nassim Massaudi, Ayush Chaurasia, Francesco Capuano, Richard Gao, Taj Gillin, Dan Haramati, Damien Scieur, Yann LeCun, and Randall Balestriero. stable-worldmodel: A Platform for Reproducible World Modeling Research and Evaluation. *arXiv preprint arXiv:2605.21800*, 2026b.

Lucas Maes, Quentin Le Lidec, Dan Haramati, Nassim Massaudi, Damien Scieur, Yann LeCun, and Randall Balestriero. stable-worldmodel-v1: Reproducible World Modeling Research and Evaluation. *arXiv preprint arXiv:2602.08968*, 2026c.

Sergi Masip, Jonathan Swinnen, Yutong Hu, Renaud Detry, and Tinne Tuytelaars. FF-JEPA: Long-Horizon Planning in World Models with Latent Planners. *arXiv preprint arXiv:2606.09311*, 2026.

David Q. Mayne, James B. Rawlings, Christopher V. Rao, and Pierre O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

Devesh Nath, Anutam Srinivasan, Haoran Yin, Ruitong Jiang, Jeffrey Fang, and Glen Chou. Pixels to Proofs: Probabilistically-Safe Latent World Model Control via Parallel Conformal Robust MPC. *arXiv preprint arXiv:2606.15594*, 2026.

Hoang Nguyen, Xiaohao Xu, and Xiaonan Huang. Latent Geometry Beyond Search: Amortizing Planning in World Models. *arXiv preprint arXiv:2605.08732*, 2026.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. HIQL: Offline goal-conditioned RL with latent states as actions. In *Advances in Neural Information Processing Systems*, 2023.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *International Conference on Learning Representations*, 2025.

Phu Pham and Aniket Bera. Latent Energy Action Planning with World Models. *arXiv preprint arXiv:2609.03294*, 2026.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. In *Conference on Robot Learning*, 2020.

Ruslan Rakhimov, George Bredis, Yuriy Maksyuta, and Daniil Gavrilov. Qantara: Bridge-Flow Training for Multi-Paradigm JEPA Control. *arXiv preprint arXiv:2607.04978*, 2026.

Gawon Seo, Dongwon Kim, and Suha Kwak. ACID: Action Consistency via Inverse Dynamics for Planning with World Models. *arXiv preprint arXiv:2607.02403*, 2026.

Vlad Sobal, Wancong Zhang, Kyunghyun 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.

Armin Sommer and Jannik Schilling. Reinforced Planning with Latent World Models. *arXiv preprint arXiv:2608.18669*, 2026.

Krishnam Soni, Aditya Sehgal, Vedant Dave, and Elmar Rueckert. Sandwich-Residuals: Parameter-Efficient Test-time Adaptation of World Models. *arXiv preprint arXiv:2609.21740*, 2026.

Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks. In *International Conference on Machine Learning*, 2018.

Junhan Sun, Hao Zhao, and Guofeng Zhang. INTACT: Isomorphic Intent-to-Action Learning for Search-Free World Models. *arXiv preprint arXiv:2607.26056*, 2026.

Erik Talvitie. Self-correcting models for model-based reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2017.

Lucas Thil, Jesse Read, Rim Kaddah, and Guillaume Doquet. Subspace-Decomposed JEPAs: Disentangling Progression and Content in Latent World Models. *arXiv preprint arXiv:2605.31111*, 2026.

Arun Venkatraman, Martial Hebert, and J. Andrew Bagnell. Improving multi-step prediction of learned time series models. In *AAAI Conference on Artificial Intelligence*, 2015.

Jiawei Wang, Ke Rui, Yushen Zuo, Yichun Feng, and Minglei Li. Decision-Metric Alignment in Latent World Models: Diagnostics and Action-Conditioned Objectives for MPC Planning. *arXiv preprint arXiv:2608.18746*, 2026.

Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. *International Conference on Machine Learning*, 2023.

Manuel Watter, Jost Tobias Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in Neural Information Processing Systems*, 2015.

Yilun Wu, Yunjian Zhang, Aobo Li, Mujiangshan Wang, Haitao Wu, and Aqiang Zhang. FIRM-WM: State-factorized factual-interventional recurrent modeling for reward-free visual planning. *arXiv preprint arXiv:2609.22816*, 2026.

Zhoushun Yu, Xiaoyu Hu, and Xiangyu Xu. QQWorld: Quantile-Quantile Matching for World Model Regularization. *arXiv preprint arXiv:2607.28415*, 2026.

Xi Zeng, Haojie Ren, and Ziying Song. PhyLatent: Learning Dynamics-Relevant Representations for JEPA World Models. *arXiv preprint arXiv:2608.05720*, 2026.

Yi-Ge Zhang, Tianqi Du, Qi Zhang, and Yisen Wang. DWM: Separating World Effects from Actions in Latent World Models. *arXiv preprint arXiv:2607.18715*, 2026a.

Zhenghao Zhang, Yuanxiang Wang, Zhenyu Guan, Yujia Yang, Bingkang Shi, Tianyu Zong, Hongzhu Yi, Guoqing Chao, Xingchen Chen, Tiankun Yang, Chenxi Bao, Tao Yu, Jingjing Zhou, and Jungang Xu. Delta-JEPA: Learning Action-Sensitive World Models via Latent Difference Decoding. *arXiv preprint arXiv:2606.31232*, 2026b.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World Models on Pre-trained Visual Features enable Zero-shot Planning. *arXiv preprint arXiv:2411.04983*, 2024.

A Survey of Evaluation Protocols

A.1 Citation count

On 24 September 2026, Semantic Scholar listed 195 papers citing LeWM (Maes et al., 2026a).

A.2 Protocol inheritance

Table 2 lists the 36 papers we found that report goal-conditioned planning in LeWM's environments: LeWM, two stable-worldmodel papers (Maes et al., 2026c;b), and 33 successors. We screened citing papers whose abstract mentions planning, excluding those that construct goals differently or do not say how. Papers that only state they follow LeWM are coded with its settings. All 36 take each goal from the same stored episode as its start. Of the 33 successors, 9 evaluate beyond $\Delta = 25$, and 25 plan with Euclidean distance (6 with added terms).

The authors hope that the great accessibility of LeWM and the insights provided by this paper may motivate exploration of more diverse environments. Of interest besides long-horizon tasks may be partial observability, multi-agent, and embodiment and background agnosticism.

Table 2: LeWM-line planning papers.

Paper	Work	Envs.	Changes	Cost	Δ evaluated	Held-out
stable-worldmodel-v1	Maes et al. (2026c)	P	Infra	L2	n.s.	n.s.
LeWM	Maes et al. (2026a)	T P R C	Repr	L2	25	no
RC-aux	Li et al. (2026b)	T P R C, Wall, LIBERO	Repr, Cost	L2+	25	n.s.
GC-IDM	Nguyen et al. (2026)	T P R C	Search	none	5–50	partly
stable-worldmodel	Maes et al. (2026b)	P C	Infra	L2	25	partly
Trajectory Reachability Metrics	Li et al. (2026a)	T P	Cost	other	25, 50, 75	no
Subspace-Decomposed JEPAs	Thil et al. (2026)	T P R C	Repr, Cost	other	25	n.s.
IMWM	Gao et al. (2026)	T P R C	Search, Cost	L2+	25	n.s.
ATM	Chen (2026)	T P C	Repr	L2	25	n.s.
FF-JEPA	Masip et al. (2026)	P, real	Search	L2†	25, 75	n.s.
Pixels to Proofs	Nath et al. (2026)	P R C, Rope, real	Search	L2+†	full episode	yes
Sensorimotor World Models	Ivashkov et al. (2026)	T P R C	Repr	L2	25^{+}	yes
Variable-Length Latent WMs	Du et al. (2026)	T P C	Pred	L2	25–100	yes
Fast LeWorldModel	Gao & Xu (2026)	T P R C	Pred	L2	25	n.s.
Delta-JEPA	Zhang et al. (2026b)	T P R C	Repr	L2	25	n.s.
ACID	Seo et al. (2026)	P R C	Cost	L2+	25	n.s.
Qantara	Rakhimov et al. (2026)	T P R C	Pred	L2	n.s.	n.s.
SAGE	Cheng et al. (2026)	P C	Search	L2	25–150	yes
DWM	Zhang et al. (2026a)	T P R, real	Pred	L2	10, 25	n.s.
VisualPatchWorld	Bai & Xiong (2026b)	P R C, Scene	Pred	other	25	n.s.
Temporal-Distance JEPA	Bai & Xiong (2026a)	T P R C	Repr, Cost	other	25	n.s.
INTACT	Sun et al. (2026)	T P R C	Search	none	25	n.s.
QQWorld	Yu et al. (2026)	T P R C	Repr	L2	25	n.s.
PhyLatent	Zeng et al. (2026)	T P R C	Repr	L2	25	n.s.
Traj-LeWM	Huang et al. (2026b)	T P R C, real	Cost	L2+	25	n.s.
SCALE	Hu et al. (2026)	T P R C, PointMaze	Repr	L2	25	n.s.
No Gaussian Required	Boylan & Hokamp (2026)	T P R C, Scene	Repr	L2	25, 100	n.s.
Reinforced Planning	Sommer & Schilling (2026)	T R C	Search	other	25, 100	n.s.
Decision-Metric Alignment	Wang et al. (2026)	T P R C	Repr	L2	n.s.	yes
LpWM	Kuang et al. (2026)	T P C, Wall	Repr	L2	25	yes
LeFlow	Huang et al. (2026a)	T P R C	Search	L2	25	partly
Latent Energy Action Planning	Pham & Bera (2026)	T P R C	Search	L2+	25	n.s.
Physically Grounded JEPA WMs	Liu et al. (2026)	T P R C	Repr	L2	25	n.s.
LePlanner	Bansal et al. (2026)	T P R C	Search	none	25	n.s.
Sandwich-Residuals	Soni et al. (2026)	P C, real	Pred	L2†	25	yes
FIRM-WM	Wu et al. (2026)	T P R C	Pred	L2	25	n.s.

Envs.: T/P/R/C = Two-Room, Push-T, Reacher, Cube. Changes: component modified (Pred, Repr, Search, Cost, Infra). Cost: planning cost (L2; L2+: L2 plus extra terms; other; none). Δ: goal offsets in environment steps. n.s.: not stated; †: inferred.

B Environment Properties

Table 3: Task properties under the same-episode construction at $\Delta = 25/75/125$ environment steps, from 4,000 sampled windows per offset over all episodes.

Environment	Success test	Eligible (%)	Solved at start (%)	Demo steps (median)
Two-Room	< 16 px	100 / 84.4 / 0	8.8 / 0.5 / –	17 / 65 / –
Push-T	agent, block < 20; angle $< \pi/9$	100 / 90.3 / 47.6	0.2 / 0.0 / 0.0	23 / 73 / 123
Reacher	joints < 0.05 rad	100 / 100 / 100	0.4 / 0.2 / 0.1	25 / 75 / 125
Cube (expert)	cube ≤ 4 cm (arm ignored)	100 / 100 / 100	38.0 / 37.3 / 16.2	21 / 45 / 32
Cube-play	cube ≤ 4 cm (arm ignored)	100 / 100 / 100	38.3 / 2.6 / 0.5	22 / 53 / 103
Cube-double-play	cubes ≤ 4 cm (arm ignored)	100 / 100 / 100	38.6 / 2.6 / 0.7	22 / 56.5 / 107
PointMaze-giant	≤ 1.0 unit	100 / 100 / 100	6.2 / 5.9 / 5.6	19 / 69 / 119

Eligible: episodes long enough for the offset. Demo steps: steps until the demonstration first satisfies the success test against the goal (an upper bound on the horizon). –: not applicable.

Table 4: Evaluation protocol: LeWM's and ours.

Property	LeWM	Ours
Evaluation episodes	training data	held out (20% of initial-state groups)
Eval tasks sharing an initial-state group with training (%)	100	0
Tasks per cell	50	200
Goal offsets Δ	25	25, 75, 125
Budget (steps)	50	2Δ
Starts already solved	included	excluded
Uncertainty	none	bootstrap CI or seed SD

Table 5: Does a larger offset make a longer task? Per-dataset properties (left) and per-task properties at $\Delta = 25/75/125$ (right), from 4,000 sampled windows per offset. Init.: distinct initial states. Tail: frames at the end of an episode already within tolerance of its final state. Lag-1: correlation between consecutive demonstration actions (0 for random actions). Layouts: distinct held-out starting states among the 200 evaluation tasks. Dist.: median start-to-goal distance relative to $\Delta = 25$. k^* and h_{lb}: median upper and lower bounds on the steps a task needs (Eq. 5). Known short: $k^* \leq 25$. Known long: $h_{\mathrm{lb}} > 25$. Tail, Eligible, Solved and Known are percentages.

	Dataset								Bounds (steps)		Known	
	Init.	Length	Tail	Lag-1	Eligible	Layouts	Solved	Dist.	k^*	h_{lb}	Short	Long
LeWM datasets												
Two-Room	7,000	101	7.6	0.06	100/84/0	191/187/–	8.8/0.5/–	1/2.5/–	17/65/–	2/6/–	100/5/–	0/0/–
Push-T	185	123	4.1	0.87	100/90/48	36/32/**15**	0.2/0.0/0.0	1/3.7/4.8	23/73/123	2/3/3	100/1/0	0/0/0
Reacher	9,970	201	0.7	**0.00**	100/100/100	191/190/186	0.4/0.2/0.1	1/1.7/2.2	25/75/125	3/5/6	100/4/2	0/0/0
Cube (expert)	6,157	201	24.9	0.88	100/100/100	191/190/186	38.0/37.3/16.2	1/1.0/2.2	21/45/32	1/1/2	100/50/51	0/0/0
OGBench datasets												
Cube-play	941	1,001	5.1	0.91	100/100/100	125/125/125	38.3/2.6/0.5	1/2.0/2.4	22/53/103	2/5/6	100/19/3	0/0/0
Cube-double-play	1,000	1,001	5.1	0.91	100/100/100	125/125/125	38.6/2.6/0.7	1/2.1/2.4	22/57/107	1/2/3	100/15/2	0/0/0
PointMaze-giant	496	2,001	4.9	0.58	100/100/100	85/85/85	6.2/5.9/5.6	1/3.0/6.0	19/69/119	–/**34/56**	100/8/7	0/**81/85**

C LEARNED COST DETAILS

The learned cost-to-go is a small head trained on frozen LeWM latents and used only as the planner's terminal cost. It is a diagnostic probe of how much a better terminal cost changes long-horizon success, not a proposed method, and we did not tune it beyond the single choice of update count below.

Parametrization. The head is a learned Euclidean distance between embedded latents,

$$V_\psi(z, z_g) = \left\| \phi_\psi(z) - \phi_\psi(z_g) \right\|_2, \tag{6}$$

where ϕ_ψ is an MLP (Linear $192 \to 816$, ReLU, Linear $816 \to 128$; 262,064 parameters) applied to the 192-dimensional LeWM embedding (encoder CLS token followed by LeWM's projector). This form makes $V_\psi(z_g, z_g) = 0$ hold by construction, so the boundary condition in Eq. (3) needs no loss term. It also makes V_ψ symmetric and satisfy the triangle inequality.

Training data. Before training, every frame of the dataset is encoded once by the frozen backbone with the evaluator's image preprocessing, and the cache is checked by re-encoding four randomly chosen blocks of 256 frames (maximum absolute difference 0.0 on the headline caches). The encoder and predictor are never updated. Training pairs are drawn by hindsight relabeling within single episodes: an anchor t is drawn uniformly from frames that have a later frame in the same episode, and the goal g uniformly from the strictly later frames, so $g > t$ always. Across an episode of N frames this gives offsets $k = g - t$ with probability proportional to $H_{N-1} - H_{k-1}$ (H the harmonic numbers), which favors short offsets, and the final frame of each episode is drawn as a goal about $8\times$ more often than a uniform frame. Anchors are restricted to the backbone's training episodes, so no held-out episode is seen by either the backbone or its head. Each head is stored with the SHA-256 of its backbone checkpoint and held-out split, and the planner refuses a head whose hashes do not match.

Objective. We fit V_ψ by undiscounted one-step temporal-difference learning with a Polyak-averaged target network $\bar{\psi}$ and the Huber loss ℓ_H (threshold 1),

$$\mathcal{L}(\psi) = \ell_H\Big(V_\psi(z_t, z_g) - y\Big), \qquad y = \begin{cases} 1 & g = t+1, \\ 1 + V_{\bar{\psi}}(z_{t+1}, z_g) & g > t+1, \end{cases} \tag{7}$$

where one step is one dataset frame, that is one environment step. The target starts as a copy of ψ and is updated as $\bar{\psi} \leftarrow (1-\tau)\bar{\psi} + \tau\psi$ after every gradient step. Table 6 lists the remaining settings.

Table 6: Learned cost-to-go training settings, identical for every environment and backbone seed.

Setting	Value
Input	frozen LeWM embedding (192-d, float32)
Architecture	$\|\phi(z) - \phi(z_g)\|_2$, ϕ: $192 \to 816 \to 128$, ReLU
Backup	one-step TD, undiscounted
Loss	Huber, threshold 1
Target network	Polyak, $\tau = 0.005$ per update
Optimizer	Adam, learning rate 3×10^{-4}, no gradient clipping
Batch size	4,096 pairs
Updates U	25,000 (about 1.0×10^8 pairs)
Head seed	0 for every backbone
Training episodes	the backbone's training split only

Use in planning. The head replaces the squared L2 cost of Eq. (2) as the only term of the planner's objective: each CEM candidate is scored by $V_\psi(\hat{z}_H, z_g)$ at its predicted endpoint, or at the re-encoded simulator endpoint in the true-dynamics conditions. Everything else in the planner is unchanged (Table 9). Because CEM only ranks candidates to select elites, the different scale of V_ψ and squared L2 does not matter.

Choice of update count. We trained heads for $U \in \{5{,}000, 10{,}000, 25{,}000\}$ with head seed 0 and evaluated them on training-episode tasks of one backbone (Table 15). Our predeclared rule, most pooled successes with ties to the smaller U, selected $U = 5{,}000$ (692 of 800, against 677 for 25,000 and 672 for 10,000). We instead fixed $U = 25{,}000$ for every head. This choice was made before any held-out cost-to-go evaluation existed, the three counts differ by at most 20 of 800 training-episode tasks, and no other setting was varied.

What the head estimates. Each backup follows a trajectory that goes on to reach g, so the fixed point of Eq. (7) is approximately the sampler-weighted mean number of steps the behavior policy β took from z to g in trajectories that visited both, rather than the shortest distance. This biases it in both directions. It overestimates wherever the data detour or wander, most strongly on Reacher, whose data come from uniformly random actions. It underestimates relative to β's true hitting time because failures and separations longer than an episode are never observed. Its symmetric form averages forward and reverse separations and cannot express one-way costs such as irreversible pushes. The head is also trained on encoder latents of real frames but queried on predictor outputs, and it is unconstrained on states the data never visit. Quasimetric and contrastive objectives address several of these issues (Section 5) and are natural replacements.

D Full Results

Table 7: Change in success (percentage points) from swapping one component, starting from L2 with imagined dynamics, and the cost×dynamics interaction $[\text{CtG}_{\text{sim}} - \text{CtG}_{\text{learned}}] - [\text{L2}_{\text{sim}} - \text{L2}_{\text{learned}}]$. Computed from the rounded means in Table 8. CtG: learned cost-to-go.

Environment	Protocol	Swap to true dynamics	Swap to CtG	Interaction
Two-Room	SHORT	−6.6	+12.0	+6.6
	LONG	−5.1	+68.2	+6.3
Push-T	SHORT	+13.6	−3.3	+7.1
	LONG	+6.5	+18.1	+11.0
	XLONG	+4.5	+10.4	+10.1
Cube-play	SHORT	+13.5	+1.8	+11.4
	LONG	+0.7	+5.2	+9.4
	XLONG	+6.0	+4.5	+12.2
Cube-double-play	SHORT	+20.0	+13.0	+0.6
	LONG	+5.1	+2.4	+8.0
	XLONG	+5.7	+4.4	+3.4
PointMaze-giant	SHORT	+0.2	+74.3	+4.3
	LONG	0.0	+39.9	−1.5
	XLONG	0.0	+15.1	−2.9

Table 8: Planning success (%) for every environment and protocol, comparing the L2 and Cost-to-Go planning costs under imagined (LeWM) and true dynamics. Entries are mean ± SD over three held-out LeWM backbone seeds (3072/3073/3074; SD with $n-1$). The SD measures variation between independently trained backbones; every seed plans on the same tasks, so it excludes task-sampling error. Tasks already solved at the start are excluded; n is the number of tasks per seed after exclusion (out of 200), and all four columns in a row use the same tasks. In each row, the highest mean among the first three result columns (excluding true-dynamics Cost-to-Go) is in bold.

			Imagined dynamics		True dynamics	
Environment	Protocol	n	L2	Cost-to-Go	L2	Cost-to-Go
Headline environments						
Two-Room	SHORT	186	88.0 ±2.2	**100.0** ±0.0	81.4 ±1.9	100.0 ±0.0
	LONG	197	30.6 ±1.8	**98.8** ±1.1	25.5 ±1.5	100.0 ±0.0
	XLONG	–	–	–	–	–
Push-T	SHORT	199	79.2 ±3.0	75.9 ±3.1	**92.8** ±0.3	96.6 ±0.3
	LONG	200	16.7 ±0.8	**34.8** ±3.1	23.2 ±1.5	52.3 ±2.6
	XLONG	200	4.8 ±2.6	**15.2** ±3.2	9.3 ±1.3	29.8 ±4.1
Cube-play	SHORT	114	28.9 ±1.5	30.7 ±1.5	**42.4** ±1.3	55.6 ±1.8
	LONG	199	21.6 ±1.5	**26.8** ±4.4	22.3 ±2.0	36.9 ±0.3
	XLONG	200	10.8 ±0.8	15.3 ±0.8	**16.8** ±1.8	33.5 ±3.3
Cube-double-play	SHORT	115	40.0 ±13.5	53.0 ±10.0	**60.0** ±10.9	73.6 ±4.5
	LONG	196	17.5 ±2.8	19.9 ±4.7	**22.6** ±4.4	33.0 ±6.6
	XLONG	198	8.6 ±4.5	13.0 ±2.5	**14.3** ±6.4	22.1 ±3.0
PointMaze-giant	SHORT	185	2.5 ±0.3	**76.8** ±16.0	2.7 ±0.0	81.3 ±14.3
	LONG	183	2.2 ±0.0	**42.1** ±16.7	2.2 ±0.0	40.6 ±15.7
	XLONG	184	1.6 ±0.0	**16.7** ±4.6	1.6 ±0.0	13.8 ±4.4
Reacher	SHORT	199	82.6 ±3.0	79.6 ±1.3	**98.8** ±0.3	99.3 ±0.8
	LONG	200	90.3 ±1.3	**99.7** ±0.6	88.5 ±0.9	100.0 ±0.0
	XLONG	200	74.8 ±1.0	**97.0** ±0.0	68.3 ±1.2	97.5 ±0.0
Appendix environment						
Cube (expert)	SHORT	117	40.2 ±1.7	**57.0** ±3.9	55.6 ±3.7	77.8 ±0.9
	LONG	131	40.7 ±0.9	**51.4** ±3.2	50.6 ±1.2	59.5 ±2.3
	XLONG	165	31.7 ±2.1	**41.4** ±2.9	41.2 ±0.6	67.5 ±2.4

E LeWM PLANNER SETTINGS

Every evaluation in the paper uses the planner below. We keep LeWM's released settings and change only what the table marks, so that differences in success come from the protocol, the cost and the dynamics, not the optimizer.

Table 9: Planner and evaluation settings. Changed: differs from LeWM's released evaluation.

Setting	Value	Changed
Optimizer	cross-entropy method (CEM)	
Samples per iteration	300	
Iterations	30	
Elites kept	30	
Initial sampling scale	1.0	
Action block	5 environment steps per predictor step	
Lookahead L	25 environment steps (5 predictor calls)	
Replanning	after executing the full 5-step plan, warm-started	
Budget B	2Δ environment steps	
Success	success test met at any step $k \leq B$	
Goal offset Δ	25, 75, 125	yes (LeWM: 25)
Tasks per cell	200, sampling seed 42	yes (LeWM: 50)
Evaluation episodes	held-out 20% of initial-state groups (split seed 42042)	yes (LeWM: training)
Tasks solved at start	excluded	yes (LeWM: counted)
Action clipping	none on LeWM's four datasets; clipped to the action box on OGBench's	OGBench only
CEM seed	42 per task	

LeWM's paper lists 10 CEM iterations outside Push-T, but its released configurations use 30 on every environment; we follow the release. The subsections below vary the planning horizon, the goal offset on a fixed set of episodes, and the budget. We also report how clipping the planner's actions changes LeWM's own reported numbers.

E.1 Planning horizon

Table 10: L2 success (%) with imagined dynamics as the lookahead L (environment steps) grows. Mean $\pm$ SD over three backbone seeds; the SD measures variation between training runs, since every seed plans on the same tasks. n: tasks per seed not solved at the start. $^{\uparrow}$/$^{\downarrow}$: significantly better/worse than $L = 25$ on the same tasks ($p < 0.05$, task-clustered sign-flip test across seeds). CEM keeps 300 samples and 30 iterations at every L, and the plan is replanned every 5 steps.

Environment	Protocol	n	$L = 25$	$L = 50$	$L = 75$	$L = 100$	$L = 125$
Two-Room	SHORT	186	88.0 $\pm$2.2	57.9 $\pm$3.5$^{\downarrow}$	–	–	–
	LONG	197	30.6 $\pm$1.8	39.4 $\pm$3.9$^{\uparrow}$	29.3 $\pm$2.1	–	–
Push-T	SHORT	199	79.2 $\pm$3.0	18.1 $\pm$4.8$^{\downarrow}$	–	–	–
	LONG	200	16.7 $\pm$0.8	9.3 $\pm$0.8$^{\downarrow}$	5.0 $\pm$1.0$^{\downarrow}$	–	–
	XLONG	200	4.8 $\pm$2.6	5.8 $\pm$2.4	3.8 $\pm$0.6	3.7 $\pm$0.8	1.5 $\pm$1.3$^{\downarrow}$
Cube-play	SHORT	114	28.9 $\pm$1.5	20.8 $\pm$1.0$^{\downarrow}$	–	–	–
	LONG	199	21.6 $\pm$1.5	9.2 $\pm$0.6$^{\downarrow}$	6.5 $\pm$1.7$^{\downarrow}$	–	–
	XLONG	200	10.8 $\pm$0.8	9.5 $\pm$1.5	8.3 $\pm$1.4	6.8 $\pm$1.6$^{\downarrow}$	6.5 $\pm$2.0$^{\downarrow}$
Cube-double-play	SHORT	115	40.0 $\pm$13.5	20.6 $\pm$2.0$^{\downarrow}$	–	–	–
	LONG	196	17.5 $\pm$2.8	7.3 $\pm$2.9$^{\downarrow}$	3.6 $\pm$0.9$^{\downarrow}$	–	–
	XLONG	198	8.6 $\pm$4.5	6.4 $\pm$2.0	4.7 $\pm$0.8$^{\downarrow}$	3.9 $\pm$2.1$^{\downarrow}$	3.0 $\pm$1.5$^{\downarrow}$
PointMaze-giant	SHORT	185	2.5 $\pm$0.3	2.7 $\pm$0.0	–	–	–
	LONG	183	2.2 $\pm$0.0	2.2 $\pm$0.0	2.0 $\pm$0.3	–	–
	XLONG	184	1.6 $\pm$0.0	1.6 $\pm$0.0	1.4 $\pm$0.6	1.4 $\pm$0.8	2.2 $\pm$0.9

E.2 Success vs offset on a fixed episode population

Longer offsets admit only longer episodes, so the task populations of SHORT and XLONG differ. Using the existing per-task records, we restrict the shorter protocols to episodes that are also eligible at the longest offset (Push-T: length ≥ 126; Two-Room: length ≥ 76) and recompute success.

Table 11: LeWM success (%) on all eligible tasks and on the fixed episode population. Entries are means over three backbone seeds with 95% bootstrap CIs clustered on source episode, which accounts for tasks sharing an episode and for the seeds sharing one task draw. n: tasks per seed not solved at the start. Cube-play, Cube-double-play and PointMaze-giant have fixed-length episodes, so their populations are identical by construction.

				Imagined dynamics		True dynamics	
Environment	Protocol	Population	n	L2	Cost-to-go	L2	Cost-to-go
Push-T	SHORT	All eligible	199	79.2 [74.5, 83.6]	75.9 [71.3, 80.3]	92.8 [89.4, 95.8]	96.6 [94.6, 98.5]
		Fixed	115	81.2 [75.1, 86.5]	73.9 [67.5, 80.1]	94.2 [90.1, 97.7]	97.1 [94.4, 99.2]
	LONG	All eligible	200	16.7 [12.5, 21.2]	34.8 [30.2, 39.5]	23.2 [17.7, 28.8]	52.3 [46.9, 57.7]
		Fixed	143	19.8 [14.2, 25.5]	35.7 [29.8, 41.5]	28.2 [21.5, 35.3]	52.0 [45.5, 58.5]
	XLONG	Both	200	4.8 [2.8, 7.2]	15.2 [11.7, 18.9]	9.3 [5.4, 13.7]	29.8 [24.4, 35.7]
Two-Room	SHORT	All eligible	186	88.0 [83.2, 92.3]	100.0 [100.0, 100.0]	81.4 [76.1, 86.4]	100.0 [100.0, 100.0]
		Fixed	173	87.9 [83.0, 92.3]	100.0 [100.0, 100.0]	80.7 [75.2, 86.1]	100.0 [100.0, 100.0]
	LONG	Both	197	30.6 [24.6, 36.9]	98.8 [97.6, 99.8]	25.5 [20.0, 31.3]	100.0 [100.0, 100.0]

E.3 Budget sensitivity

Every protocol uses a budget of twice the goal offset. On LONG ($\Delta = 75$) and one backbone seed, we rerun each cell with budgets of $1\times$ and $3\times$ the offset (75 and 225 steps), changing nothing else; the $2\times$ column is the main-table run of that seed.

Table 12: LeWM success (%) on LONG at three planning budgets, backbone seed 3072, tasks not solved at the start. Brackets: 95% bootstrap CIs clustered on source episode, the right interval for a single seed where the only sampling error is the task draw. All budgets use identical tasks and CEM seeds.

Environment	Cost	n	$1\times$ offset	$2\times$ offset (default)	$3\times$ offset
Imagined dynamics					
Two-Room	L2	197	27.4 [20.9, 34.0]	30.5 [23.6, 37.2]	33.5 [26.6, 40.3]
	Cost-to-go	197	95.9 [92.9, 98.5]	100.0 [100.0, 100.0]	100.0 [100.0, 100.0]
Push-T	L2	200	13.5 [8.6, 18.4]	16.0 [11.1, 21.1]	16.5 [11.4, 21.7]
	Cost-to-go	200	23.5 [17.8, 29.4]	31.5 [25.3, 37.8]	34.5 [27.9, 41.1]
Cube-play	L2	199	21.6 [15.9, 27.4]	21.6 [15.9, 27.4]	21.6 [15.9, 27.4]
	Cost-to-go	199	31.2 [24.7, 37.8]	31.7 [25.0, 38.4]	31.7 [25.0, 38.4]
Cube-double-play	L2	196	18.9 [13.6, 24.7]	18.9 [13.6, 24.7]	18.9 [13.6, 24.7]
	Cost-to-go	196	20.9 [15.5, 27.0]	20.9 [15.5, 27.0]	20.9 [15.5, 27.0]
PointMaze-giant	L2	183	2.2 [0.5, 4.4]	2.2 [0.5, 4.4]	2.2 [0.5, 4.4]
	Cost-to-go	183	27.9 [22.3, 33.7]	43.7 [37.8, 50.0]	46.4 [40.2, 53.0]
True dynamics					
Two-Room	L2	197	23.9 [17.9, 30.3]	26.4 [20.2, 33.0]	26.9 [20.5, 33.7]
	Cost-to-go	197	100.0 [100.0, 100.0]	100.0 [100.0, 100.0]	100.0 [100.0, 100.0]
Push-T	L2	200	21.5 [15.8, 27.2]	21.5 [15.8, 27.2]	21.5 [15.8, 27.2]
	Cost-to-go	200	44.0 [37.0, 51.0]	49.5 [42.6, 56.5]	51.0 [44.2, 58.0]
Cube-play	L2	199	23.6 [17.8, 29.7]	24.1 [18.1, 30.2]	24.1 [18.1, 30.2]
	Cost-to-go	199	36.2 [29.2, 43.2]	36.7 [29.8, 43.6]	36.7 [29.8, 43.6]
Cube-double-play	L2	196	23.0 [16.7, 29.5]	23.5 [17.2, 30.0]	23.5 [17.2, 30.0]
	Cost-to-go	196	32.7 [26.3, 39.4]	32.7 [26.3, 39.4]	32.7 [26.3, 39.4]
PointMaze-giant	L2	183	2.2 [0.5, 4.4]	2.2 [0.5, 4.4]	2.2 [0.5, 4.4]
	Cost-to-go	183	33.9 [27.5, 40.8]	36.6 [30.4, 43.1]	36.6 [30.4, 43.1]

E.4 Effect of action clipping under the authors' evaluator

The authors' evaluator executes CEM actions outside the action box. Using LeWM's released checkpoints and evaluator (offset 25, budget 50), we reran Cube (expert) and Push-T with every CEM candidate clamped to the action box, on identical tasks and CEM seeds.

Table 13: Success (%) of LeWM's released checkpoints with unclipped and clipped CEM actions, all tasks (including those solved at the start), with successes/tasks. Clip-only / unclip-only: tasks solved under one condition only. Difference is clipped minus unclipped with a 95% paired (Newcombe) CI, and p is the exact McNemar test; both use the pairing of identical tasks. The 50-task draw is the one the authors' released evaluator makes by default (seed 42), which we take to be Figure 6's; on it the released checkpoints solve 46 and 34 of 50 tasks, not the reported 48 and 37. Table 1's Official column uses our evaluator on nearly the same 200 tasks (196 of 200 shared on Push-T, 199 on Cube) and gives 1.5 points less; each difference between the two evaluators moves success by at most 2 points on identical tasks.

Environment	Tasks	Unclipped	Clipped	Clip-only / unclip-only	Difference (pp)	p
Cube (expert)	50	68.0 (34/50)	66.0 (33/50)	2 / 3	-2.0 $[-10.8, +6.8]$	1.00
	200	63.5 (127/200)	62.5 (125/200)	8 / 10	-1.0 $[-5.1, +3.1]$	0.81
Push-T	50	92.0 (46/50)	94.0 (47/50)	2 / 1	$+2.0$ $[-6.6, +11.3]$	1.00
	200	83.0 (166/200)	85.5 (171/200)	9 / 4	$+2.5$ $[-1.1, +6.3]$	0.27

Table 14: Out-of-range actions under the unclipped authors' evaluator, 200 tasks, over executed steps up to each task's first success. Exact counts on one run, so no interval is given. Displacement ratio: largest per-step effector (Cube) or agent (Push-T) displacement divided by the dataset's largest. With clipping, no step exceeds $|a| = 1$.

Environment	Steps with $\lvert a\rvert > 1$ (%)	Max $\lvert a\rvert$	Displacement ratio	Successes using such steps (%)
Cube (expert)	27.2 (of 4,347)	15.67	1.85×	42.5 (54/127)
Push-T	0.93 (of 5,933)	2.21	1.26×	2.4 (4/166)

F Additional results

F.1 Head update count

Before any held-out evaluation, we ran LeWM heads trained for 5k, 10k and 25k updates on training-episode tasks of one backbone. The paper uses 25k updates for every head.

Table 15: Successes out of 200 training-episode tasks (SHORT, $H = 5$, backbone seed 3072, all tasks including those solved at the start) for L2 and LeWM cost-to-go heads at three update counts U. Counts only, with no interval: the cells come from one seed on training episodes and were used only to fix U, not to compare heads.

Environment	L2	$U = 5000$	$U = 10000$	$U = 25000$
Two-Room	174	200	200	200
Push-T	174	175	173	167
Cube (expert)	137	147	143	141
Reacher	165	170	156	169
Total (of 800)	650	692	672	677

F.2 Paired significance of every headline cell

For each LeWM cell of Table 8 in the five headline environments, we test three paired comparisons on the per-task outcomes: cost-to-go vs L2 under imagined dynamics, cost-to-go vs L2 under true dynamics, and true vs imagined dynamics under L2.

F.3 PointMaze-giant audit

We replayed all 7,200 PointMaze-giant LeWM episodes in the simulator and found no wall violations and no action outside $[-1, 1]$. We also measured task difficulty against OGBench's own giant-maze tasks, and ran an oracle planner that uses the same CEM, horizon, budget and tasks with simulator rollouts and the true shortest-path distance to the goal as its cost.

Table 16: Paired tests for each headline cell (LeWM). Discordants: per seed (3072 / 3073 / 3074), tasks solved only by the first-named condition : tasks solved only by the second, on identical tasks. p: task-clustered sign-flip test pooled over the three seeds (200,000 flips), which respects the pairing and treats a task's three seed outcomes as one unit. †: $p \geq 0.05$. n: tasks per seed not solved at the start.

			Cost-to-go vs L2, learned		Cost-to-go vs L2, simulator		L2: simulator vs learned	
Environment	Protocol	n	Discordants	p	Discordants	p	Discordants	p
Two-Room	SHORT	186	23:0/18:0/26:0	$< 10^{-5}$	38:0/31:0/35:0	$< 10^{-5}$	1:16/2:15/2:11	8×10^{-5}
	LONG	197	137:0/129:0/137:0	$< 10^{-5}$	145:0/150:0/145:0	$< 10^{-5}$	5:13/2:19/2:7	8.5×10^{-5}
Push-T	SHORT	199	19:23/15:21/12:22	$0.07^\dagger$	11:4/10:3/11:2	0.0068	39:6/26:5/35:8	$< 10^{-5}$
	LONG	200	46:15/54:14/49:11	$< 10^{-5}$	70:14/72:12/69:10	$< 10^{-5}$	25:14/24:10/24:10	0.0062
	XLONG	200	32:1/17:8/29:7	$< 10^{-5}$	55:7/43:3/41:6	$< 10^{-5}$	18:1/10:8/12:4	0.0074
Cube-play	SHORT	114	14:15/16:11/15:13	$0.67^\dagger$	26:15/29:11/26:10	0.0083	23:7/25:9/21:7	0.00078
	LONG	199	31:11/15:15/27:16	0.031	38:13/42:14/44:10	2×10^{-5}	14:9/11:12/10:10	$0.79^\dagger$
	XLONG	200	16:7/14:8/16:4	0.0089	34:11/47:4/40:6	$< 10^{-5}$	22:7/17:10/20:6	0.0071
Cube-double-play	SHORT	115	30:5/14:11/22:5	$< 10^{-5}$	20:2/11:4/27:5	1.5×10^{-5}	28:5/28:8/29:3	$< 10^{-5}$
	LONG	196	12:8/15:6/9:8	$0.13^\dagger$	23:5/34:8/20:3	$< 10^{-5}$	12:3/19:5/11:4	0.00058
	XLONG	198	18:4/14:12/13:3	0.0039	21:5/15:7/26:4	7.5×10^{-5}	18:2/20:7/10:5	0.00065
PointMaze-giant	SHORT	185	138:0/166:0/108:0	$< 10^{-5}$	147:0/171:0/118:0	$< 10^{-5}$	0:0/0:0/1:0	$1^\dagger$
	LONG	183	76:0/102:0/41:0	$< 10^{-5}$	63:0/102:0/46:0	$< 10^{-5}$	0:0/0:0/0:0	$1^\dagger$
	XLONG	184	28:0/36:0/19:0	$< 10^{-5}$	23:0/30:0/14:0	$< 10^{-5}$	0:0/0:0/0:0	$1^\dagger$

Table 17: PointMaze-giant task difficulty and solvability. Distances are medians over the 200 tasks per protocol, in maze units; the geodesic goes around walls. These describe a fixed task set, so no interval is given. Oracle and demonstration success are over all 200 tasks per protocol (100% also with start-solved tasks excluded); the oracle ran CEM on CPU, so its random draws differ from the GPU cells despite equal seeds. Steps are medians over successful tasks.

		Start→goal			Success (%)		Median steps	
Protocol	Δ	Geodesic	Straight-line	Behind wall (%)	Oracle	Demo	Oracle	Demo
SHORT	25	3.8	3.8	0	100.0	100.0	19	19
LONG	75	10.7	10.0	60	100.0	100.0	46	69
XLONG	125	17.6	14.0	77	100.0	100.0	80.5	119
OGBench giant (5 tasks)	–	51.5–91.5	–	100	–	–	–	–

F.4 Evaluator checks: Two-Room door corners and Reacher angle wrap

Two-Room tests door passage only at each step's endpoint, so a diagonal step can cut a door corner (by at most 4.9 px past the door tolerance; the dataset's expert trajectories cut to the same depth). We reran every Two-Room cell with a collision test that also blocks such cuts, and recomputed Reacher's success test with the shoulder-angle difference wrapped to $(-\pi, \pi]$.

Table 18: PointMaze-giant LeWM success (%) per backbone seed, as start-solved-excluded rate / rate on all 200 tasks. Per-seed rates are shown because the cost-to-go SD across seeds reaches 16.7 points; they are exact counts on a fixed task set, so no interval is attached. n: tasks per seed not solved at the start (identical across seeds and conditions).

Dynamics	Cost	Protocol	n	Seed 3072	Seed 3073	Seed 3074
Learned	L2	SHORT	185	2.7 / 10.0	2.7 / 10.0	2.2 / 9.5
		LONG	183	2.2 / 10.5	2.2 / 10.5	2.2 / 10.5
		XLONG	184	1.6 / 9.5	1.6 / 9.5	1.6 / 9.5
	Cost-to-go	SHORT	185	77.3 / 79.0	92.4 / 93.0	60.5 / 63.5
		LONG	183	43.7 / 48.5	57.9 / 61.5	24.6 / 31.0
		XLONG	184	16.8 / 23.5	21.2 / 27.5	12.0 / 19.0
Simulator	L2	SHORT	185	2.7 / 10.0	2.7 / 10.0	2.7 / 10.0
		LONG	183	2.2 / 10.5	2.2 / 10.5	2.2 / 10.5
		XLONG	184	1.6 / 9.5	1.6 / 9.5	1.6 / 9.5
	Cost-to-go	SHORT	185	82.2 / 83.5	95.1 / 95.5	66.5 / 69.0
		LONG	183	36.6 / 42.0	57.9 / 61.5	27.3 / 33.5
		XLONG	184	14.1 / 21.0	17.9 / 24.5	9.2 / 16.5

Table 19: Two-Room success (%) with the stock collision test and with door-corner cuts blocked, LeWM, tasks not solved at the start. Rates are mean ± SD over three backbone seeds, as in Table 8. Cut: episodes (of 600, three seeds) with at least one door-corner cut. Flips: pooled tasks turning success→failure / failure→success when cuts are blocked; p is the task-clustered sign-flip test over seeds, which suits paired reruns of identical tasks.

Dynamics	Cost	Protocol	Cut	Flips	p	Stock test	Cuts blocked
Learned	Cost-to-go	SHORT	89	1 / 0	1	100.0 ±0.0	99.8 ±0.3
		LONG	278	10 / 0	0.016	98.8 ±1.1	97.1 ±2.1
	L2	SHORT	115	2 / 0	0.5	88.0 ±2.2	87.6 ±2.5
		LONG	206	3 / 5	0.69	30.6 ±1.8	31.0 ±2.2
Simulator	Cost-to-go	SHORT	53	0 / 0	1	100.0 ±0.0	100.0 ±0.0
		LONG	251	0 / 0	1	100.0 ±0.0	100.0 ±0.0
	L2	SHORT	105	1 / 3	0.75	81.4 ±1.9	81.7 ±2.2
		LONG	190	6 / 4	0.77	25.5 ±1.5	25.2 ±0.3

Table 20: Reacher decisions changed by wrapping the shoulder-angle difference, over all 7,200 LeWM episodes (both costs, both dynamics, three protocols, three seeds). Rates are means over three seeds; the SD is 0.0 in every listed cell, so only the mean is shown. No start-solved flag changes.

Dynamics	Cost	Protocol	Tasks changed	Stock test	Wrapped
Learned	Cost-to-go	XLONG	6	97.0	100.0
Simulator	Cost-to-go	XLONG	5	97.5	100.0
All other conditions and protocols			0	unchanged	

G PER-SEED RESULTS

Tables 21–24 give the per-seed numbers behind Table 8, one table per planning condition. Every seed and condition in a row plans on the same n tasks (out of 200), after excluding tasks already solved at the start.

Table 21: Per-seed planning success (%) for L2 cost with imagined dynamics. Each seed cell gives the number of tasks solved k out of the row's n tasks not solved at the start, the success rate k/n, and in brackets a 95% percentile bootstrap confidence interval (10,000 resamples) that resamples source episodes with replacement, because tasks drawn from the same episode are correlated. The interval covers task-sampling error for that one backbone seed only. Mean is the unweighted mean of the three seed rates and matches Table 8.

			Seed 3072		Seed 3073		Seed 3074		
Environment	Protocol	n	k	% 95% CI	k	% 95% CI	k	% 95% CI	Mean
Two-Room	SHORT	186	163	87.6 [82.5, 92.3]	168	90.3 [85.6, 94.6]	160	86.0 [80.7, 90.8]	88.0
	LONG	197	60	30.5 [23.8, 37.4]	64	32.5 [25.9, 39.2]	57	28.9 [22.5, 35.6]	30.6
	XLONG	–		–		–		–	–
Push-T	SHORT	199	152	76.4 [70.4, 82.2]	164	82.4 [77.0, 87.4]	157	78.9 [73.0, 84.4]	79.2
	LONG	200	32	16.0 [11.1, 21.3]	35	17.5 [12.4, 22.9]	33	16.5 [11.6, 21.6]	16.7
	XLONG	200	4	2.0 [0.5, 4.1]	14	7.0 [3.5, 10.9]	11	5.5 [2.5, 8.9]	4.8
Cube-play	SHORT	114	34	29.8 [21.2, 39.0]	31	27.2 [19.1, 36.0]	34	29.8 [21.1, 38.7]	28.9
	LONG	199	43	21.6 [15.9, 27.4]	46	23.1 [17.5, 29.3]	40	20.1 [14.9, 25.6]	21.6
	XLONG	200	22	11.0 [6.7, 15.5]	23	11.5 [7.2, 16.2]	20	10.0 [5.8, 14.4]	10.8
Cube-double-play	SHORT	115	45	39.1 [30.5, 48.1]	62	53.9 [44.8, 62.9]	31	27.0 [18.6, 35.5]	40.0
	LONG	196	37	18.9 [13.4, 24.5]	38	19.4 [13.6, 25.4]	28	14.3 [9.3, 19.7]	17.5
	XLONG	198	14	7.1 [3.6, 11.2]	27	13.6 [8.8, 19.0]	10	5.1 [2.2, 8.2]	8.6
PointMaze-giant	SHORT	185	5	2.7 [0.6, 5.3]	5	2.7 [0.6, 5.2]	4	2.2 [0.5, 4.4]	2.5
	LONG	183	4	2.2 [0.5, 4.5]	4	2.2 [0.5, 4.5]	4	2.2 [0.5, 4.5]	2.2
	XLONG	184	3	1.6 [0.0, 3.6]	3	1.6 [0.0, 3.6]	3	1.6 [0.0, 3.6]	1.6
Reacher	SHORT	199	158	79.4 [73.5, 84.7]	165	82.9 [77.5, 88.0]	170	85.4 [80.4, 90.0]	82.6
	LONG	200	183	91.5 [87.4, 95.1]	178	89.0 [84.7, 93.1]	181	90.5 [86.3, 94.4]	90.3
	XLONG	200	149	74.5 [68.2, 80.6]	152	76.0 [69.6, 82.0]	148	74.0 [67.7, 80.2]	74.8
Cube (expert)	SHORT	117	49	41.9 [33.3, 50.4]	45	38.5 [29.9, 47.0]	47	40.2 [31.6, 48.7]	40.2
	LONG	131	54	41.2 [32.8, 50.0]	52	39.7 [31.6, 48.1]	54	41.2 [32.6, 50.0]	40.7
	XLONG	165	52	31.5 [24.2, 38.9]	56	33.9 [26.9, 41.0]	49	29.7 [23.2, 36.4]	31.7

Table 22: Per-seed planning success (%) for Cost-to-Go with imagined dynamics. Each seed cell gives the number of tasks solved k out of the row's n tasks not solved at the start, the success rate k/n, and in brackets a 95% percentile bootstrap confidence interval (10,000 resamples) that resamples source episodes with replacement, because tasks drawn from the same episode are correlated. The interval covers task-sampling error for that one backbone seed only. Mean is the unweighted mean of the three seed rates and matches Table 8.

			Seed 3072		Seed 3073		Seed 3074		
Environment	Protocol	n	k	% 95% CI	k	% 95% CI	k	% 95% CI	Mean
Two-Room	SHORT	186	186	100.0 [100.0, 100.0]	186	100.0 [100.0, 100.0]	186	100.0 [100.0, 100.0]	100.0
	LONG	197	197	100.0 [100.0, 100.0]	193	98.0 [95.9, 99.5]	194	98.5 [96.5, 100.0]	98.8
	XLONG	–	–		–		–		–
Push-T	SHORT	199	148	74.4 [68.2, 80.4]	158	79.4 [73.7, 84.8]	147	73.9 [67.8, 79.9]	75.9
	LONG	200	63	31.5 [25.2, 38.1]	75	37.5 [30.8, 44.4]	71	35.5 [29.1, 42.2]	34.8
	XLONG	200	35	17.5 [12.4, 23.0]	23	11.5 [7.1, 16.1]	33	16.5 [11.4, 21.9]	15.2
Cube-play	SHORT	114	33	28.9 [20.9, 37.1]	36	31.6 [23.0, 40.4]	36	31.6 [23.2, 40.0]	30.7
	LONG	199	63	31.7 [25.0, 38.6]	46	23.1 [17.2, 29.1]	51	25.6 [19.4, 32.0]	26.8
	XLONG	200	31	15.5 [10.2, 21.2]	29	14.5 [9.9, 19.3]	32	16.0 [11.3, 21.1]	15.3
Cube-double-play	SHORT	115	70	60.9 [52.2, 69.2]	65	56.5 [48.1, 65.0]	48	41.7 [33.0, 50.8]	53.0
	LONG	196	41	20.9 [15.3, 26.8]	47	24.0 [17.5, 31.1]	29	14.8 [9.9, 20.1]	19.9
	XLONG	198	28	14.1 [9.6, 19.0]	29	14.6 [9.6, 19.9]	20	10.1 [6.1, 14.4]	13.0
PointMaze-giant	SHORT	185	143	77.3 [70.1, 83.9]	171	92.4 [88.1, 96.2]	112	60.5 [52.7, 68.0]	76.8
	LONG	183	80	43.7 [37.6, 50.3]	106	57.9 [51.0, 65.4]	45	24.6 [18.4, 31.2]	42.1
	XLONG	184	31	16.8 [11.7, 22.3]	39	21.2 [15.8, 26.9]	22	12.0 [7.0, 17.2]	16.7
Reacher	SHORT	199	158	79.4 [73.8, 84.7]	156	78.4 [72.2, 84.1]	161	80.9 [75.3, 86.2]	79.6
	LONG	200	198	99.0 [97.5, 100.0]	200	100.0 [100.0, 100.0]	200	100.0 [100.0, 100.0]	99.7
	XLONG	200	194	97.0 [94.4, 99.0]	194	97.0 [94.5, 99.0]	194	97.0 [94.4, 99.0]	97.0
Cube (expert)	SHORT	117	71	60.7 [52.1, 69.2]	62	53.0 [43.6, 62.4]	67	57.3 [47.9, 66.7]	57.0
	LONG	131	72	55.0 [46.6, 63.4]	64	48.9 [40.2, 57.5]	66	50.4 [41.9, 58.8]	51.4
	XLONG	165	72	43.6 [36.0, 51.5]	70	42.4 [34.8, 50.0]	63	38.2 [30.9, 45.4]	41.4

Table 23: Per-seed planning success (%) for L2 cost with true dynamics. Each seed cell gives the number of tasks solved k out of the row's n tasks not solved at the start, the success rate k/n, and in brackets a 95% percentile bootstrap confidence interval (10,000 resamples) that resamples source episodes with replacement, because tasks drawn from the same episode are correlated. The interval covers task-sampling error for that one backbone seed only. Mean is the unweighted mean of the three seed rates and matches Table 8.

			Seed 3072		Seed 3073		Seed 3074		
Environment	Protocol	n	k	% 95% CI	k	% 95% CI	k	% 95% CI	Mean
Two-Room	SHORT	186	148	79.6 [73.6, 85.3]	155	83.3 [77.8, 88.6]	151	81.2 [75.4, 86.8]	81.4
	LONG	197	52	26.4 [20.2, 32.8]	47	23.9 [17.9, 30.0]	52	26.4 [20.2, 32.7]	25.5
	XLONG	–	–		–		–		–
Push-T	SHORT	199	185	93.0 [89.2, 96.4]	185	93.0 [89.3, 96.4]	184	92.5 [88.6, 95.9]	92.8
	LONG	200	43	21.5 [15.9, 27.3]	49	24.5 [18.4, 30.8]	47	23.5 [17.6, 29.9]	23.2
	XLONG	200	21	10.5 [5.8, 15.9]	16	8.0 [4.0, 12.7]	19	9.5 [5.5, 13.9]	9.3
Cube-play	SHORT	114	50	43.9 [34.7, 52.8]	47	41.2 [32.1, 50.0]	48	42.1 [32.7, 51.4]	42.4
	LONG	199	48	24.1 [18.2, 30.4]	45	22.6 [16.8, 28.8]	40	20.1 [14.7, 25.6]	22.3
	XLONG	200	37	18.5 [13.2, 23.8]	30	15.0 [10.3, 20.2]	34	17.0 [11.9, 22.5]	16.8
Cube-double-play	SHORT	115	68	59.1 [50.0, 67.6]	82	71.3 [63.0, 79.1]	57	49.6 [40.9, 58.1]	60.0
	LONG	196	46	23.5 [17.2, 29.9]	52	26.5 [20.3, 33.0]	35	17.9 [12.4, 23.6]	22.6
	XLONG	198	30	15.2 [10.4, 20.3]	40	20.2 [14.5, 26.1]	15	7.6 [3.9, 11.6]	14.3
PointMaze-giant	SHORT	185	5	2.7 [0.6, 5.3]	5	2.7 [0.6, 5.3]	5	2.7 [0.6, 5.3]	2.7
	LONG	183	4	2.2 [0.5, 4.5]	4	2.2 [0.5, 4.5]	4	2.2 [0.5, 4.4]	2.2
	XLONG	184	3	1.6 [0.0, 3.6]	3	1.6 [0.0, 3.6]	3	1.6 [0.0, 3.6]	1.6
Reacher	SHORT	199	196	98.5 [96.5, 100.0]	197	99.0 [97.5, 100.0]	197	99.0 [97.5, 100.0]	98.8
	LONG	200	178	89.0 [84.5, 93.1]	178	89.0 [84.5, 93.0]	175	87.5 [82.8, 91.8]	88.5
	XLONG	200	138	69.0 [62.3, 75.4]	138	69.0 [62.3, 75.4]	134	67.0 [60.2, 73.6]	68.3
Cube (expert)	SHORT	117	70	59.8 [50.4, 68.4]	63	53.8 [44.4, 63.2]	62	53.0 [44.4, 62.4]	55.6
	LONG	131	66	50.4 [41.9, 58.9]	68	51.9 [43.5, 60.5]	65	49.6 [41.1, 58.3]	50.6
	XLONG	165	69	41.8 [34.5, 49.4]	68	41.2 [33.8, 48.8]	67	40.6 [33.5, 47.9]	41.2

Table 24: Per-seed planning success (%) for Cost-to-Go with true dynamics. Each seed cell gives the number of tasks solved k out of the row's n tasks not solved at the start, the success rate k/n, and in brackets a 95% percentile bootstrap confidence interval (10,000 resamples) that resamples source episodes with replacement, because tasks drawn from the same episode are correlated. The interval covers task-sampling error for that one backbone seed only. Mean is the unweighted mean of the three seed rates and matches Table 8.

			Seed 3072		Seed 3073		Seed 3074		
Environment	Protocol	n	k	% 95% CI	k	% 95% CI	k	% 95% CI	Mean
Two-Room	SHORT	186	186	100.0 [100.0, 100.0]	186	100.0 [100.0, 100.0]	186	100.0 [100.0, 100.0]	100.0
	LONG	197	197	100.0 [100.0, 100.0]	197	100.0 [100.0, 100.0]	197	100.0 [100.0, 100.0]	100.0
	XLONG	–		–		–		–	–
Push-T	SHORT	199	192	96.5 [93.8, 99.0]	192	96.5 [93.9, 99.0]	193	97.0 [94.4, 99.0]	96.6
	LONG	200	99	49.5 [42.7, 56.4]	109	54.5 [47.5, 61.4]	106	53.0 [45.9, 60.0]	52.3
	XLONG	200	69	34.5 [27.7, 41.7]	56	28.0 [21.7, 34.9]	54	27.0 [20.1, 34.0]	29.8
Cube-play	SHORT	114	61	53.5 [44.1, 62.6]	65	57.0 [47.7, 66.4]	64	56.1 [47.1, 65.0]	55.6
	LONG	199	73	36.7 [29.7, 43.9]	73	36.7 [29.9, 43.5]	74	37.2 [30.5, 44.0]	36.9
	XLONG	200	60	30.0 [23.9, 36.1]	73	36.5 [29.7, 43.2]	68	34.0 [27.5, 40.6]	33.5
Cube-double-play	SHORT	115	86	74.8 [66.4, 82.6]	89	77.4 [69.2, 85.1]	79	68.7 [59.5, 77.4]	73.6
	LONG	196	64	32.7 [26.1, 39.2]	78	39.8 [33.2, 46.4]	52	26.5 [20.5, 32.8]	33.0
	XLONG	198	46	23.2 [17.7, 29.2]	48	24.2 [18.5, 30.3]	37	18.7 [13.5, 24.1]	22.1
PointMaze-giant	SHORT	185	152	82.2 [76.0, 88.0]	176	95.1 [91.9, 97.9]	123	66.5 [57.9, 75.0]	81.3
	LONG	183	67	36.6 [30.5, 43.3]	106	57.9 [50.0, 65.9]	50	27.3 [21.7, 33.3]	40.6
	XLONG	184	26	14.1 [9.4, 19.1]	33	17.9 [12.6, 23.6]	17	9.2 [5.1, 14.0]	13.8
Reacher	SHORT	199	196	98.5 [96.5, 100.0]	198	99.5 [98.5, 100.0]	199	100.0 [100.0, 100.0]	99.3
	LONG	200	200	100.0 [100.0, 100.0]	200	100.0 [100.0, 100.0]	200	100.0 [100.0, 100.0]	100.0
	XLONG	200	195	97.5 [95.1, 99.5]	195	97.5 [95.3, 99.5]	195	97.5 [95.0, 99.5]	97.5
Cube (expert)	SHORT	117	92	78.6 [70.9, 85.5]	91	77.8 [70.1, 84.6]	90	76.9 [69.2, 84.6]	77.8
	LONG	131	78	59.5 [51.2, 67.7]	75	57.3 [48.8, 65.4]	81	61.8 [53.4, 70.0]	59.5
	XLONG	165	109	66.1 [58.9, 73.2]	109	66.1 [58.9, 73.2]	116	70.3 [63.3, 77.1]	67.5