Predictive but Not Plannable: RC-aux for Latent World Models

Wenyuan Li
Hokkaido University
wenyuan@lmd.ist.hokudai.ac.jp

Guang Li*
Hokkaido University
guang@lmd.ist.hokudai.ac.jp

Keisuke Maeda
Hokkaido University
maeda@lmd.ist.hokudai.ac.jp

Takahiro Ogawa
Hokkaido University
ogawa@lmd.ist.hokudai.ac.jp

Miki Haseyama
Hokkaido University
mhaseyama@lmd.ist.hokudai.ac.jp

Abstract

A latent world model may achieve accurate short-horizon prediction while still inducing a latent space that is poorly aligned with planning. A key issue is spatiotemporal mismatch: these models are often trained with local predictive supervision, but deployed for long-horizon goal-directed search in latent spaces where Euclidean distance may not reflect what is reachable within a finite action budget. We present the **R**eachability-**C**orrection **aux**iliary objective (RC-aux), a lightweight correction for this mismatch in reconstruction-free latent world models. RC-aux keeps the world-model backbone unchanged and adds planning-aligned supervision along two axes. Along the *time* axis, multi-horizon open-loop prediction trains the model beyond one-step consistency. Along the *space* axis, budget-conditioned reachability supervision, together with temporal hard negatives, encourages the latent space to distinguish states that are eventually reachable from those reachable within the current planning horizon. At test time, the learned reachability signal can also be used by a reachability-aware planner to favor trajectories that are both goal-directed and attainable under the available budget. We instantiate RC-aux on LeWorldModel and evaluate it under both continuation-training and matched-from-scratch settings. Across goal-conditioned pixel-control tasks and a LIBERO-Goal extension, RC-aux improves LeWM-style planning with modest additional cost. These results suggest that planning with latent world models depends not only on predictive accuracy, but also on whether the learned representation encodes the temporal and geometric structure required by downstream search. The code is available at https://github.com/Guang000/RC-aux.

1 Introduction

A latent world model can be predictive without being plannable. In goal-conditioned control from pixels, planning requires more than accurate latent prediction. The planner also relies on the latent space to rank candidate trajectories: terminal latent distance should reflect feasible progress toward the goal, and the planning horizon should reflect what can be achieved with a finite action budget. A candidate rollout may end near the goal in Euclidean latent distance while following a shortcut

*Correspondence to: Guang Li <guang@lmd.ist.hokudai.ac.jp>

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

that is not supported by feasible finite-horizon transitions. When the latent geometry permits such shortcuts, a planner can optimize a plausible latent objective while selecting trajectories that are short in representation space but poor in the environment. Figure 1 illustrates this failure mode and the correction introduced by our method.

Latent world models have become a central tool for control from pixels, enabling agents to plan or learn behavior through compact latent rollouts [17, 19, 20]. Recent reconstruction-free and JEPA-style models make this especially attractive for offline and goal-conditioned control, since they avoid pixel-level decoding while still supporting planning in learned latent spaces [47, 38, 29]. However, many such models are trained under objectives that are only indirectly aligned with their test-time use. A model trained primarily for short-horizon latent prediction is later embedded in a long-horizon planner that optimizes terminal latent distance to a goal. This train-test gap makes prediction accuracy alone insufficient for reliable planning.

The gap creates two coupled mismatches. The temporal mismatch arises because training supervision is concentrated on short-horizon prediction, whereas the planner must evaluate many-step open-loop rollouts at test time. The spatial mismatch arises because Euclidean proximity in latent space does not necessarily reflect the finite-horizon reachability structure supported by the data. As a result, planning can fail even when one-step prediction is accurate: the planner may optimize a latent objective whose geometry does not distinguish feasible progress from unsupported latent-space shortcuts. Recent work on value-guided representations, temporal straightening, plannable continuous latents, and quasimetric reachability similarly shows that planning depends on representation geometry, not only prediction error [10, 44, 24, 43, 34, 3]. These approaches motivate planning-aware geometry, but they do not directly align the open-loop rollout distribution, finite-horizon reachability structure, and terminal objective used by latent planners.

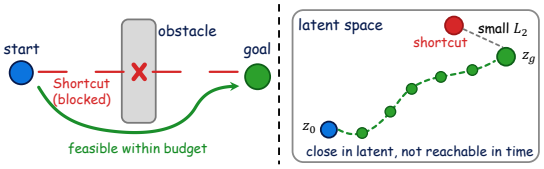

Figure 1: Conceptual illustration of the latent-shortcut failure mode. A terminal latent-distance planner may favor a shortcut that is close in latent space but unreachable within the finite action budget. RC-aux encourages distance to align with finite-horizon reachability, making feasible trajectories more consistent with planning.

We propose a Reachability-Correction auxiliary objective (RC-aux) for latent world models, together with a reachability-aware planner that uses the learned signal during search. RC-aux makes the training queries more closely match the queries made by the planner. It corrects the temporal mismatch by training the dynamics model on multi-horizon open-loop rollouts. It corrects the spatial mismatch by learning a budget-conditioned reachability predicate that estimates whether a target latent is attainable from a source latent within a finite horizon. The reachability objective uses reachable positives, batch negatives, and same-trajectory *temporal hard negatives*: future states that are eventually attainable but not reachable under the current budget. These hard negatives make the budget dependence identifiable by forcing the model to distinguish "reachable eventually" from "reachable within this horizon." We also impose the same reachability structure on predicted rollout latents, so states generated during planning inherit the finite-horizon semantics used by the planner.

At test time, our planner couples this learned reachability signal with terminal latent planning. Standard terminal latent L2 planning is recovered as the $\lambda_{\text{plan}} = 0$ member of this planner family. For $\lambda_{\text{plan}} > 0$, the planner favors candidate rollouts that are close to the goal under the base latent cost and contain intermediate predicted states from which the goal is estimated reachable within the remaining horizon. Thus, reachability becomes an explicit search signal rather than only a training-time regularizer.

We instantiate RC-aux on LeWorldModel (LeWM), a compact reconstruction-free JEPA world model for goal-conditioned planning from pixels [29]. RC-aux preserves the LeWM backbone, allowing us to isolate the effect of correcting the training horizon, shaping finite-horizon reachability geometry, and using the learned reachability signal during search. Although our experiments use LeWM, the objective itself is backbone-agnostic: it requires only latent rollouts, goal latents, and finite-horizon transition structure.

Our experiments evaluate RC-aux on the original LeWM task family and benchmark extensions under matched local protocols, including continuation controls, paired fixed-episode outcomes, ablations, and rollout visualizations. Across these settings, RC-aux improves four out of five matched LeWM-family comparisons, with the clearest gain on Wall, where finite-budget reachability makes latent Euclidean distance a particularly weak proxy for planning progress. These results support the central claim of this paper: strong latent planning requires not only accurate prediction, but a representation whose notions of distance and horizon are aligned with data-supported finite-horizon reachability.

Our contributions are:

- We introduce RC-aux, a backbone-agnostic auxiliary objective that makes latent world models more plannable by jointly correcting rollout-horizon mismatch and finite-horizon reachability geometry.
- We introduce a reachability-aware planner family that uses the learned reachability signal during test-time search, with standard terminal latent L2 planning recovered when $\lambda_{\text{plan}} = 0$.
- We provide a formal analysis showing how multi-horizon open-loop prediction, budget-conditioned reachability, and temporal hard negatives align training with finite-horizon planning queries.
- We provide controlled empirical validation on LeWorldModel, including matched comparisons, continuation controls, ablations, rollout visualizations, and a larger LIBERO-Goal extension with OFT-style action heads.

2 Related Work

RC-aux builds on latent world models for pixel-based control, including recurrent latent planners, imagined-rollout agents, and decoder-free control-centric models [17, 19, 20]. It is closest to reconstruction-free and JEPA-style models such as DINO-WM, PLDM, and LeWorldModel, which avoid pixel reconstruction and plan directly in learned or pretrained visual feature spaces [47, 38, 29]. Rather than introducing a new backbone, RC-aux studies how to train the latent space so that it better exposes the finite-horizon structure needed by planning.

RC-aux is also related to planning-aware representation learning, including value-guided latent distances, temporally structured representations, plannable latent spaces, reachability learning, and quasimetric objectives [10, 44, 24, 43, 34, 3, 41]. These methods show that latent proximity or short-step prediction error alone may be insufficient for control. RC-aux targets this mismatch with budget-conditioned reachability supervision, temporal hard negatives, and multi-horizon open-loop prediction. Additional discussion is provided in Appendix A.

3 Method

We consider goal-conditioned control from pixels with a latent world model. The offline dataset consists of trajectories

$$\mathcal{D} = \{\tau^{(n)}\}_{n=1}^{N}, \qquad \tau = (o_{1:T}, a_{1:T-1}), \tag{1}$$

where o_t is an image observation and a_t is the action between o_t and o_{t+1}. A latent world model consists of an encoder e_θ and an action-conditioned latent dynamics model. The encoder maps observations to latent states,

$$z_t = e_\theta(o_t), \tag{2}$$

and the dynamics model predicts future latents under candidate action sequences. Given a current observation o_t and a goal observation o_g, a latent planner encodes

$$z_t = e_\theta(o_t), \qquad z_g = e_\theta(o_g), \tag{3}$$

rolls out candidate action sequences in latent space, and selects the sequence with the lowest goal-matching cost.

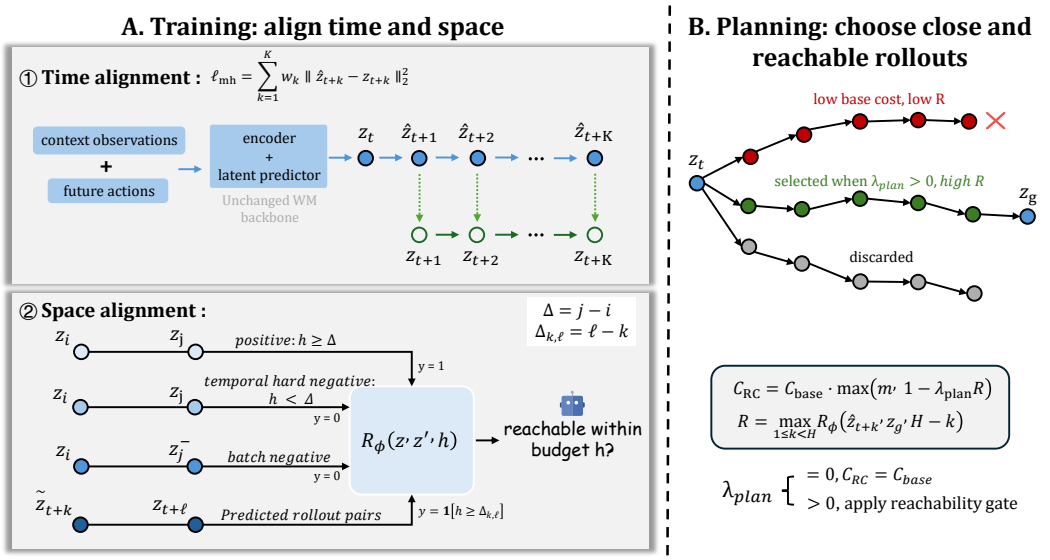

Figure 2: The framework of RC-aux. RC-aux keeps the latent world-model backbone unchanged and aligns it with planning through two training signals: multi-horizon open-loop prediction for time alignment, and budget-conditioned reachability $R_\phi(z, z', h)$ for space alignment. Reachability is trained with positives, trajectory-induced hard negatives, batch negatives, and predicted rollout pairs. At test time, $\lambda_{\text{plan}} = 0$ recovers the base latent planner, while $\lambda_{\text{plan}} > 0$ gates the base goal cost by trajectory-level reachability to favor rollouts that are both close to the goal and empirically attainable.

3.1 Time alignment: open-loop multi-horizon prediction

A planner evaluates a world model by feeding predicted latents back into the dynamics model. RC-aux therefore trains the model in the same open-loop regime. For a sampled trajectory segment, let $z_{t-L+1:t}$ be the context latents and let $a_{t:t+K-1}$ be the future actions used for rollout. Any action history used by the underlying backbone is absorbed into the context notation. The model produces an open-loop prediction

$$\hat{z}_{t+1:t+K} = F_\theta^{(K)}\left(z_{t-L+1:t}, a_{t:t+K-1}\right), \tag{4}$$

where predicted latents are fed back into the dynamics model for later steps. The encoded future latents

$$z_{t+1}, z_{t+2}, \ldots, z_{t+K} \tag{5}$$

are used as supervision targets, not as rollout inputs. For this segment, the multi-horizon open-loop loss is

$$\ell_{\text{mh}} = \sum_{k=1}^{K} w_k \left\|\hat{z}_{t+k} - z_{t+k}\right\|_2^2, \tag{6}$$

where w_k are horizon weights. The context latent z_t is the rollout start; the first supervised prediction is $\hat{z}_{t+1}$, which is matched to z_{t+1}. Thus, the loss directly trains the model on the predicted latents that will later be scored by the planner.

3.2 Space alignment: budget-conditioned reachability

Multi-step prediction alone does not guarantee that latent distance reflects attainability. RC-aux therefore learns a budget-conditioned reachability predicate

$$R_\phi(z, z', h) = \sigma(r_\phi(z, z', h)) \in [0, 1], \tag{7}$$

where $R_\phi(z, z', h)$ estimates whether target latent z' is reachable from source latent z within a finite budget h. The reachability head is conditioned on an ordered source–target pair and a horizon budget. The order matters because finite-horizon reachability is generally asymmetric.

For same-trajectory pairs (z_i, z_j) with $i < j$, define the observed trajectory offset

$$\Delta = j - i. \tag{8}$$

RC-aux uses trajectory offsets to construct empirical finite-budget reachability labels. These labels should be interpreted as *trajectory-induced supervision* rather than ground-truth shortest-path reachability in the underlying MDP: a target that is observed after Δ steps along one trajectory may in principle be reachable by a shorter unobserved path. Nevertheless, such offsets provide a useful proxy for the finite-horizon attainability structure that a planner needs.

For each ordered pair, budgets are sampled from

$$h \in \{0, 1, \ldots, H_{\max}\}. \tag{9}$$

The trajectory-induced label is positive when the target is observed within the sampled budget along the trajectory, and negative otherwise:

$$y_{ijh} = \mathbf{1}[h \geq \Delta]. \tag{10}$$

Reachable positives. If $h \geq \Delta$, then z_j is observed within the available budget after z_i along the sampled trajectory. The pair (z_i, z_j, h) is labeled reachable.

Trajectory-induced temporal hard negatives. If $h < \Delta$, then z_j is not observed to be reached from z_i within budget h along the sampled trajectory. The pair (z_i, z_j, h) is therefore used as a trajectory-induced temporal hard negative. These examples force the reachability head to distinguish eventual co-occurrence along a trajectory from empirical finite-budget attainability.

Batch negatives. Targets sampled from other trajectories in the minibatch are used as cross-trajectory negatives. These negatives prevent the reachability head from declaring arbitrary latent pairs attainable. For a batch negative (z_i, z_j^-, h), we assign label

$$y_{ijh}^- = 0. \tag{11}$$

Predicted rollout pairs. The planner queries reachability on predicted latents, not only on encoded dataset latents. RC-aux therefore also trains the reachability head on predicted-latent inputs produced by open-loop rollouts. For the open-loop rollout in Eq. (4), predicted latents used in reachability supervision are treated as stop-gradient inputs unless otherwise stated:

$$\tilde{z}_{t+k} = \mathrm{sg}(\hat{z}_{t+k}). \tag{12}$$

For predicted-source pairs, RC-aux pairs a predicted intermediate latent $\tilde{z}_{t+k}$ with an encoded future target $z_{t+\ell}$, where $0 < k < \ell \leq K$. The empirical offset from the predicted source to the encoded target is

$$\Delta_{k,\ell} = \ell - k, \tag{13}$$

and the corresponding finite-budget label is

$$y_{k\ell h} = \mathbf{1}[h \geq \Delta_{k,\ell}]. \tag{14}$$

This construction matches the planner-time query: from a predicted intermediate latent, the model asks whether a future target can be reached within the remaining budget. The encoded-latent reachability terms provide representation-shaping gradients to the encoder, while the stop-gradient predicted-latent terms calibrate the reachability head on planner-induced latent distributions.

Avoiding budget-agnostic shortcuts. Trajectory-induced temporal hard negatives make the budget variable identifiable. Without them, all same-trajectory pairs observed by the reachability loss would be positive, while batch negatives would be negative. In that case, a classifier could fit the supervision by simply distinguishing same-trajectory pairs from cross-trajectory pairs, without using the budget h. Temporal hard negatives remove this degenerate solution. For the same ordered pair (z_i, z_j) with offset $\Delta = j - i$, the label is negative when $h < \Delta$ and positive when $h \geq \Delta$. Therefore, any classifier that fits these labels must assign different predictions to the same latent pair under different budgets. This forces $R_\phi(z, z', h)$ to represent empirical finite-horizon attainability rather than merely visual similarity or trajectory membership.

Planning-alignment view. Ideally, a finite-horizon planner would benefit from a predicate that indicates whether a goal is reachable within the remaining budget. Let $D^\star(s, g)$ denote the minimum number of environment steps required to reach g from s, with $D^\star(s, g) = \infty$ if g is unreachable. The ideal finite-budget reachability predicate is

$$R_h^\star(s, g) = \mathbf{1}[D^\star(s, g) \leq h]. \tag{15}$$

RC-aux does not assume access to $D^\star$. Instead, it uses observed trajectory offsets as an empirical proxy for this horizon-indexed predicate. This proxy is directed and budget-dependent, matching the structure of finite-horizon planning more closely than a symmetric, budget-free Euclidean latent distance. Appendix B further formalizes this view by analyzing open-loop cost distortion, budget identifiability, and the effect of reachability-aware planning as a soft feasibility gate.

Let $\mathcal{B}_{\text{enc}}$ be the set of reachability pairs constructed from encoded latents and $\mathcal{B}_{\text{pred}}$ the set of analogous pairs involving stop-gradient predicted rollout latents. For a pair (z, z', h, y) with label $y \in \{0, 1\}$, define

$$\ell_{\text{bce}}(z, z', h, y) = \text{BCE}\left(R_\phi(z, z', h), y\right). \tag{16}$$

The reachability loss for a minibatch is

$$\begin{aligned} \ell_{\text{reach}} = \frac{1}{|\mathcal{B}_{\text{enc}}|} \sum_{(z,z',h,y)\in\mathcal{B}_{\text{enc}}} \omega_y \, \ell_{\text{bce}}(z, z', h, y) \\ + \rho_{\text{pred}} \frac{1}{|\mathcal{B}_{\text{pred}}|} \sum_{(z,z',h,y)\in\mathcal{B}_{\text{pred}}} \omega_y \, \ell_{\text{bce}}(z, z', h, y), \end{aligned} \tag{17}$$

where ω_y are optional class-balancing weights and ρ_{pred} controls the weight on predicted-latent reachability supervision.

3.3 RC-aux training objective

RC-aux preserves the underlying world-model backbone and its latent regularization, but replaces local prediction supervision with planning-aligned open-loop supervision. For a sampled trajectory segment, the core RC-aux objective is

$$\ell_{\text{RC-aux}} = \ell_{\text{mh}} + \alpha \ell_{\text{reg}} + \beta \ell_{\text{reach}}, \tag{18}$$

where ℓ_{mh} is the multi-horizon open-loop prediction loss, ℓ_{reach} is the budget-conditioned reachability loss, and ℓ_{reg} denotes the latent regularizer of the underlying world model. For the LeWM instantiation, ℓ_{reg} is the original SIGReg regularizer, while ℓ_{mh} replaces the original one-step latent prediction loss rather than being added on top of it. During training, Eq. (18) is averaged over sampled trajectory segments. Thus, RC-aux is not a new world-model architecture and is not merely an additional test-time planner; it changes what the learned latent dynamics and latent geometry are trained to support: open-loop prediction over planning horizons and finite-budget reachability.

3.4 Planner family

At evaluation time, RC-aux uses the same latent rollout interface as the base planner. For a candidate action sequence

$$\tau = a_{t:t+H-1}, \tag{19}$$

the model predicts

$$\hat{z}_{t+1:t+H} = F_\theta^{(H)}\left(z_{t-L+1:t}, \tau\right). \tag{20}$$

Let $C_{\text{base}}(\tau)$ denote the base latent goal cost. For terminal latent planning,

$$C_{\text{base}}(\tau) = \|\hat{z}_{t+H} - z_g\|_2^2. \tag{21}$$

Other reductions over the predicted rollout, such as minimum, soft-minimum, or mean distance, can also be used.

RC-aux defines a planner family parameterized by a reachability coupling weight

$$\lambda_{\text{plan}} \in [0, 1]. \tag{22}$$

For each predicted intermediate latent $\hat{z}_{t+k}$, the reachability head asks whether the goal can be reached within the remaining budget $H - k$. We use intermediate latents with positive remaining budget:

$$1 \leq k < H. \tag{23}$$

The trajectory-level reachability score is

$$R(\tau, z_g) = \max_{1 \leq k < H} R_\phi \left(\hat{z}_{t+k}, z_g, H - k \right). \tag{24}$$

The reachability-aware planning cost is

$$C_{\mathrm{RC}}(\tau) = C_{\mathrm{base}}(\tau) \cdot \max \left(m,\, 1 - \lambda_{\mathrm{plan}} R(\tau, z_g) \right), \tag{25}$$

where $m > 0$ is a small floor for numerical stability.

When $\lambda_{\mathrm{plan}} = 0$, Eq. (25) recovers the base latent planner exactly. When $\lambda_{\mathrm{plan}} > 0$, the reachability score softly discounts the base cost for rollouts whose intermediate predicted latents are estimated to make the goal reachable within the remaining budget. Low reachability does not add a separate penalty; it simply removes this discount. Thus, the planner still respects the base latent goal cost while favoring trajectories that are both geometrically close and empirically attainable under the learned reachability predicate. This planner-side use is optional: $\lambda_{\mathrm{plan}} = 0$ isolates the training-side effect of RC-aux, while $\lambda_{\mathrm{plan}} > 0$ evaluates reachability-aware planning.

4 Experiments

4.1 Experimental Setup

We evaluate RC-aux on five pixel-based goal-conditioned control tasks: TwoRoom [39], Reacher [42], Push-T [47], Wall [47], and Cube [33]. Each episode provides an initial observation and a goal image, and we report success rate as the primary metric. We compare RC-aux with prior baselines and LeWM-family controls under the same evaluation protocol. For local LeWM-family runs, results are averaged over five fixed evaluation groups; when available, we include a continuation-trained LeWM control to account for additional training under the same backbone. We additionally evaluate representation transfer on LIBERO-Goal [28] by training an OFT-style action head on top of the learned representation. Full evaluation details, planner settings, and training hyperparameters are provided in Appendix C. All experiments were conducted on a single NVIDIA RTX 6000 Ada GPU.

4.2 Main Results

Table 1 reports success rates on the five pixel-based goal-conditioned control tasks. We include prior baselines for benchmark context and local LeWM-family runs for controlled comparison. RC-aux achieves strong performance across the suite, with the clearest improvement on Wall, where success increases from 52.8% for LeWM to 83.6%. It also improves over the continuation-trained LeWM control on TwoRoom, Reacher, and Cube, reaching 98.0%, 87.2%, and 76.0% success, respectively. On Push-T, all LeWM-family methods are already in a high-success regime, and RC-aux remains competitive with the strongest local control.

Table 1: Success rates on five pixel-based goal-conditioned control tasks. Local LeWM-family results are reported as mean±std over five fixed evaluation groups. The matched Δ row compares RC-aux against LeWM-cont.

Method	TwoRoom	Reacher	Push-T	Wall	Cube
DINO-WM [47]	100.0	79.0	74.0	96.0	86.0
PLDM [39]	97.0	78.0	78.0	–	65.0
DINO-WM+prop [47]	–	–	92.0	–	–
GCBC [13]	100.0	–	75.0	–	84.0
IQL [23]	100.0	–	20.0	–	64.0
IVL [33]	100.0	–	33.0	–	56.0
LeWM [29]	88.8±3.0	81.2±7.9	90.4±3.0	50.4±6.5	72.4±5.9
LeWM-cont [29]	88.8±3.0	82.8±7.2	**91.2±3.9**	52.8±4.8	72.8±5.2
RC-aux	**98.0±1.4**	**87.2±6.4**	90.8±3.3	**83.6±3.6**	**76.0±7.5**
Matched Δ	+9.2	+4.4	-0.4	+30.8	+3.2

The matched Δ row of Table 1 summarizes the within-family comparison against LeWM-cont under the same task protocol across all five tasks. RC-aux improves four out of five matched comparisons. The largest gain appears on Wall, consistent with the motivation of RC-aux: latent Euclidean proximity is a particularly poor proxy for finite-horizon reachability in obstacle-constrained planning.

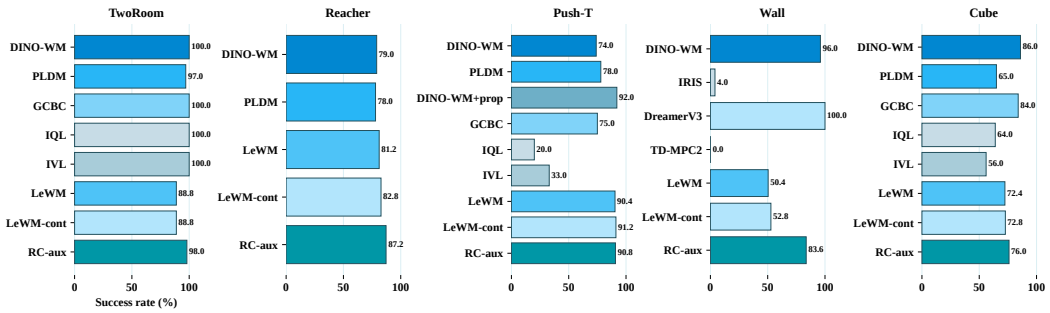

Figure 3: Success rates across the five pixel-based control tasks. RC-aux improves four of the five matched LeWM-family comparisons and gives the largest gain on Wall.

Figure 3 visualizes the same comparison. The bar chart highlights both the broader benchmark context and the local LeWM-family comparison. RC-aux is most effective on Wall and remains strong across the remaining tasks, supporting the view that reachability correction is most useful when the planner must reason about what can be reached within a finite action budget. Appendix E provides paired fixed-episode outcomes and additional local success visualizations.

Qualitative rollouts. Qualitative rollout visualizations are provided in Appendix F. Representative Cube and Wall examples illustrate how RC-aux maintains intermediate progress toward the goal, complementing the quantitative results in Table 1.

4.3 Ablation Study

We examine the contributions of planner-side scoring, time and space alignment, and explicit budget conditioning.

Reachability-aware planning. We compare the same RC-aux-trained model with and without reachability-aware scoring. Setting $\lambda_{\text{plan}} = 0$ recovers the original terminal latent-distance planner. For planner-on evaluation, we select λ_{plan} from $\{0, 0.35, 0.85\}$ on validation episodes disjoint from the fixed test groups, then freeze it for testing. All other planner settings remain unchanged within each task. As shown in Table 2, reachability-aware scoring improves success by 4.8 points on TwoRoom and 6.0 points on Reacher, demonstrating its utility beyond the base latent goal cost.

Table 2: Planner-side ablation using the same RC-aux model. Success rates are percentages.

Task	Planner off	Planner on
TwoRoom	93.2	**98.0**
Reacher	81.2	**87.2**

Time and space alignment. We compare a matched continued-training control, time alignment alone, space alignment alone, and full RC-aux. All variants retain the base latent regularizer. The control uses the original training objective; time alignment alone replaces one-step prediction with multi-horizon prediction; space alignment alone adds reachability supervision to the original one-step objective; and full RC-aux combines both modifications. Data, initialization procedures, and optimization budgets are matched, and all variants use the same evaluation episodes with $\lambda_{\text{plan}} = 0$.

Table 3 shows task-dependent contributions. Space alignment accounts for most of the TwoRoom improvement (84.8% to 92.4%), with full RC-aux reaching 93.2%. On Wall, time alignment increases success from 52.8% to 69.6%, while full RC-aux reaches 74.8%. The full objective achieves the highest observed success on both tasks, although its margin over space alignment alone is small on TwoRoom. Its 22.0-point gain over the Wall continuation control also indicates that additional training alone does not explain the improvement.

Explicit budget conditioning. We compare $R_\phi(z, z', h)$ with a budget-free directed-distance model $d_\psi(z, z')$ trained to regress trajectory offsets. The latter can represent a reachability decision

Table 3: Training-side ablations with $\lambda_{\text{plan}} = 0$. Success rates are percentages. MH denotes multi-horizon prediction.

Training variant	TwoRoom	Wall
Continued-training control	84.8	52.8
Time alignment only	86.4	69.6
Space alignment only	92.4	60.0
Full RC-aux	**93.2**	**74.8**
MH + directed distance	87.6	63.6

through $\widetilde{R}(z, z', h) = \mathbf{1}[d_\psi(z, z') \leq h]$. Both variants retain multi-horizon prediction and use matched backbones, initialization, training budgets, and evaluation protocols. Neither auxiliary head nor the threshold decision is used during planning; both variants use the original terminal latent-distance planner. The comparison therefore evaluates the effect of training supervision on downstream planning. Explicit budget conditioning outperforms distance regression by 5.6 points on TwoRoom and 11.2 points on Wall (Table 3), supporting the benefit of directly supervising finite-budget reachability. Additional paired outcomes and local success visualizations are provided in Appendix E.

4.4 Model Size and Computational Overhead

RC-aux preserves the LeWM backbone and adds only a lightweight reachability head. Table 4 reports the parameter footprint of the modules used during planner scoring. RC-aux increases the parameter count from 18.034M to 18.710M, corresponding to only a 3.74% overhead over LeWM.

For comparison with a foundation-model-based world model, we also include DINO-WM-S/14. Unlike RC-aux, DINO-WM uses a frozen DINOv2-S/14 visual encoder pretrained on external image data. Although this encoder is frozen, it is still used during planner scoring, so we include it in the scoring-time parameter footprint. Under this accounting, DINO-WM-S/14 has 42.179M active scoring-time parameters, about $2.25\times$ the size of RC-aux.

Table 4: Parameter footprint of modules used during planner scoring. For DINO-WM-S/14, the frozen DINOv2-S/14 encoder is included because it is active at scoring time.

Model	parameters	Relative to RC-aux
DINO-WM-S/14 [47]	42.179M	$2.25\times$
LeWM [29]	18.034M	$0.96\times$
RC-aux	18.710M	$1.00\times$

We also measure controlled planner cost-call overhead. This microbenchmark isolates model/planner scoring from environment simulation, rendering, video generation, and logging. All LeWM and RC-aux measurements use the horizon $H = 5$, 1024 candidate action sequences, 5 warmup calls, and 20 measured calls. As shown in Table 5, reachability-aware scoring adds less than 0.8 ms per cost call over LeWM across all tasks.

DINO-WM-S/14 is included as a foundation-model-based baseline. Its frozen DINOv2-S/14 [32] visual encoder substantially increases scoring-time computation. On Push-T, DINO-WM-S/14 takes 4527.141 ms per planner cost call with 512 candidate action sequences, while the 1024-candidate setting exceeds memory under the same hardware setup.

Table 5: Controlled planner cost-call timing. LeWM and RC-aux are measured with $H = 5$ and 1024 candidate action sequences. DINO-WM-S/14 is also included as a foundation-model-based baseline.

Task	Model	Candidates	ms / call
TwoRoom	LeWM [29]	1024	34.253
	RC-aux	1024	35.016
Reacher	LeWM [29]	1024	34.893
	RC-aux	1024	35.361
Push-T	DINO-WM-S/14 [47]	512	4527.141
	DINO-WM-S/14 [47]	1024	-
	LeWM [29]	1024	34.619
	RC-aux	1024	35.295

Table 6: LIBERO-Goal extension with an OFT-style action head. The main matched comparison is between trainable LeWM and trainable RC-aux under the no-repeat protocol.

Method	Setting	Mean success
LeWM + OFT-style head	Trainable, no-repeat	0.712
RC-aux + OFT-style head	Trainable, no-repeat	0.812
RC-aux + OFT-style head	Repeat-tuned	0.864
OpenVLA-OFT 7B [21, 22]	External reference	0.970

Overall, RC-aux retains the low-cost planning profile of LeWM while adding only a small reachability module. In contrast to DINO-WM, it does not rely on an externally pretrained visual foundation-model encoder.

4.5 LIBERO-Goal Extension

We further evaluate whether the RC-aux representation transfers beyond the five latent-planning tasks. For this experiment, we use LIBERO-Goal [28] with an OFT-style [21] action head trained on top of the learned representation. This setting differs from the CEM-style latent planner used above: the representation is used by an action-prediction head rather than directly searched by a planner. It therefore tests whether the reachability-corrected representation remains useful in a larger robot manipulation setting.

Table 6 reports all-task mean success on LIBERO-Goal. Under the aligned trainable no-repeat protocol, RC-aux improves over the trainable LeWM comparator from 0.712 to 0.812. With repeat tuning on weaker tasks, RC-aux further reaches 0.864 mean success. This shows that the representation learned with RC-aux is not only useful for terminal latent planning but also provides a stronger initialization for downstream action-head training. Per-task LIBERO-Goal results are provided in Appendix D.

5 Conclusion

We presented RC-aux, a lightweight auxiliary training objective for making reconstruction-free latent world models more useful for planning. Rather than modifying the LeWM backbone, RC-aux changes what the latent space is encouraged to represent: not only predictive consistency over future observations, but also whether a candidate state is attainable under a finite action budget. This distinction is crucial for goal-conditioned control, where latent proximity alone can make infeasible states appear deceptively promising to a planner. Our results suggest that a compact latent world model can become substantially more effective when its representation is trained around the operational questions asked at test time. In this sense, RC-aux is not simply an accuracy improvement, but a planning-alignment correction: it helps the model expose reachability structure that standard rollout losses tend to leave implicit. The current formulation still uses trajectory-derived reachability labels as a proxy for true environment-level attainability, and the test-time planner uses a simple reachability gate rather than a full decision-theoretic treatment of feasibility. Extending this idea to uncertainty-aware reachability, richer directed distances, and larger-scale manipulation settings is an important direction for future work.

6 Limitations and Future Work

RC-aux learns finite-budget reachability from observed trajectory offsets, which may overestimate the minimum number of steps needed to reach a goal when shorter routes are absent from the data. The learned signal therefore reflects dataset-supported reachability rather than an exact shortest-path oracle. Aggregating evidence across trajectories or using soft labels could improve this supervision. The planner applies a lightweight soft correction rather than enforcing feasibility constraints; uncertainty-aware scoring and analysis under stochastic dynamics are natural extensions. Our evaluation focuses on finite-budget planning, with the full objective primarily tested on LeWorldModel. The LIBERO-Goal extension provides complementary evidence of representation transfer through an action-prediction head. Further evaluation across backbones, data-coverage conditions, and longer horizons would help characterize its broader applicability.

Acknowledgments

This work was partly supported by Council for Science, Technology and Innovation(CSTI), Cross-ministerial Strategic Innovation Promotion Program (SIP), the 3rd period of SIP "Creation of new ways of learning and working in a post-COVID-19 era" Grant Number JPJ012347 (Funding agency: JST), and JSPS KAKENHI Grant Numbers JP23K11211, JP24K02942, and JP25K21218.

References

[1] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15619–15629, 2023.

[2] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

[3] Junik Bae, Kwanyoung Park, and Youngwoon Lee. Tldr: Unsupervised goal-conditioned rl via temporal distance-aware representations. *arXiv preprint arXiv:2407.08464*, 2024.

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

[5] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

[6] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

[7] Maxime Burchi and Radu Timofte. Mudreamer: Learning predictive world models without reconstruction. *arXiv preprint arXiv:2405.15083*, 2024.

[8] Pablo Samuel Castro, Tyler Kastner, Prakash Panangaden, and Mark Rowland. Mico: Improved representations via sampling-based state similarity for markov decision processes. *Advances in Neural Information Processing Systems*, 34:30113–30126, 2021.

[9] Fei Deng, Ingook Jang, and Sungjin Ahn. Dreamerpro: Reconstruction-free model-based reinforcement learning with prototypical representations. In *International conference on machine learning*, pages 4956–4975. PMLR, 2022.

[10] Matthieu Destrade, Oumayma Bounou, Quentin Le Lidec, Jean Ponce, and Yann LeCun. Value-guided action planning with jepa world models. *arXiv preprint arXiv:2601.00844*, 2026.

[11] Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2786–2793. IEEE, 2017.

[12] Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In *International conference on machine learning*, pages 2170–2179. PMLR, 2019.

[13] Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. *arXiv preprint arXiv:1912.06088*, 2019.

[14] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[15] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

[16] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[17] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[18] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[19] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[20] Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

[21] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

[22] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[23] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[24] Anurag Koul, Shivakanth Sujit, Shaoru Chen, Ben Evans, Lili Wu, Byron Xu, Rajan Chari, Riashat Islam, Raihan Seraj, Yonathan Efroni, et al. Pclast: Discovering plannable continuous latent states. *arXiv preprint arXiv:2311.03534*, 2023.

[25] Thanard Kurutach, Aviv Tamar, Ge Yang, Stuart J Russell, and Pieter Abbeel. Learning plannable representations with causal infogan. *Advances in Neural Information Processing Systems*, 31, 2018.

[26] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International conference on machine learning*, pages 5639–5650. PMLR, 2020.

[27] Yann LeCun. A path towards autonomous machine intelligence version 0.9.2, 2022-06-27. 2022.

[28] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023.

[29] Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

[30] Tung D Nguyen, Rui Shu, Tuan Pham, Hung Bui, and Stefano Ermon. Temporal predictive coding for model-based planning in latent space. In *International conference on machine learning*, pages 8130–8139. PMLR, 2021.

[31] Masashi Okada and Tadahiro Taniguchi. Dreamingv2: Reinforcement learning with discrete world models without reconstruction. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 985–991. IEEE, 2022.

[32] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[33] Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. *arXiv preprint arXiv:2410.20092*, 2024.

[34] Zhifeng Qian, Mingyu You, Hongjun Zhou, Xuanhui Xu, and Bin He. Goal-conditioned reinforcement learning with disentanglement-based reachability planning. *IEEE Robotics and Automation Letters*, 8(8):4721–4728, 2023.

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

[36] Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. *arXiv preprint arXiv:2007.05929*, 2020.

[37] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. Planning to explore via self-supervised world models. In *International conference on machine learning*, pages 8583–8592. PMLR, 2020.

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

[39] Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim GJ Rudner, and Yann LeCun. Stress-testing offline reward-free reinforcement learning: A case for planning with latent dynamics models. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

[40] Aravind Srinivas, Allan Jabri, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Universal planning networks: Learning generalizable representations for visuomotor control. In *International conference on machine learning*, pages 4732–4741. PMLR, 2018.

[41] Lorenzo Steccanella and Anders Jonsson. State representation learning for goal-conditioned reinforcement learning. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 84–99. Springer, 2022.

[42] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[43] Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*, pages 36411–36430. PMLR, 2023.

[44] Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim GJ Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. *arXiv preprint arXiv:2603.12231*, 2026.

[45] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. *Advances in neural information processing systems*, 28, 2015.

[46] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. *arXiv preprint arXiv:2006.10742*, 2020.

[47] 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 Extended Related Work

Latent world models for control from pixels. World models learn compact predictive representations that support decision-making by imagining the consequences of actions. This idea builds on earlier efforts to learn compact latent or predictive models for control from high-dimensional observations, including locally linear latent dynamics for raw-image control, generative world models, and action-conditioned visual foresight [45, 15, 11]. Early pixel-based model-based agents such as PlaNet showed that latent dynamics can enable online planning from images [17], while Dreamer-style agents learn behavior from imagined rollouts inside recurrent latent models [16, 18, 19]. Control-centric approaches such as TD-MPC and TD-MPC2 further demonstrate that decoder-free latent models can be highly effective for continuous control when coupled with trajectory optimization or value learning [20]. Related latent-planning agents also show that models need not reconstruct pixels to be useful for control: MuZero plans with a learned model that predicts planning-relevant quantities, while Plan2Explore uses self-supervised world models for exploration and fast downstream adaptation [35, 37]. More recent reconstruction-free world models move closer to our setting. Before DINO-WM and PLDM, several Dreamer-style variants already explored reducing or removing pixel reconstruction through temporal predictive coding, prototype-based objectives, contrastive objectives, or predictive task-relevant targets [30, 9, 31, 7]. DINO-WM performs test-time goal planning by predicting future DINOv2 feature embeddings from offline trajectories [32, 47], while PLDM studies reward-free offline planning with latent dynamics learned through JEPA-style self-supervision [38]. LeWorldModel (LeWM) shows that an end-to-end JEPA world model can be trained stably from pixels using next-embedding prediction together with a Gaussian latent regularizer [29]. RC-aux builds directly on this line of reconstruction-free latent planning, but targets a different failure mode: even when a latent model is predictive, its learned geometry may not encode which goals are reachable within the planner's finite horizon.

Joint-embedding predictive architectures. Joint-Embedding Predictive Architectures (JEPAs) avoid pixel reconstruction by predicting future or masked representations in a learned latent space [27]. Image and video JEPAs, such as I-JEPA and V-JEPA, use masked prediction and target-encoder mechanisms to learn semantic visual representations [1, 5, 2]. These designs are related to a broader family of self-supervised representation-learning methods that use target networks, temporal prediction, or contrastive objectives to obtain useful visual features for downstream control [14, 36, 26]. For action-conditioned world modeling, the central challenge is to learn representations that do not collapse while remaining useful for control. Some methods avoid collapse by freezing pretrained encoders [47]; others use multi-term regularizers inspired by VICReg [6, 38]. LeWM instead uses SIGReg to enforce an isotropic Gaussian latent distribution and achieves stable end-to-end JEPA training from pixels [4, 29]. RC-aux keeps this backbone unchanged. Rather than introducing a new anti-collapse mechanism, it adds planning-aligned supervision on top of the existing latent world model: multi-horizon open-loop prediction for temporal alignment and budget-conditioned reachability for geometric alignment.

Planning-aware representation geometry. A growing body of work shows that latent planning depends not only on prediction accuracy, but also on the geometry of the representation used by the planner. Earlier work on plannable representations similarly learned latent spaces that support goal specification, trajectory optimization, or symbolic-style planning from high-dimensional observations [40, 25]. Value-guided JEPA shapes representation distances so that they approximate a goal-conditioned value function or quasi-distance [10]. Temporal Straightening regularizes latent trajectories to reduce curvature, making Euclidean distance a better proxy for geodesic progress and improving the conditioning of latent planning [44]. PcLast learns plannable continuous representations by associating reachable states in ℓ_2 space [24]. Complementary representation-learning work studies MDP-aware, bisimulation-based, or behavioral metrics that align latent distances with decision-relevant state similarity [12, 46, 8]. Related work in goal-conditioned reinforcement learning studies temporal distance, reachability, and asymmetric quasimetric structure as planning-relevant notions of state similarity [43, 34, 3, 41]. RC-aux shares the view that planning requires a representation with operational geometry. However, instead of replacing terminal latent planning with a learned value metric or imposing an unconditional geometric prior, RC-aux learns a *budget-conditioned* reachability signal. This distinction is important: a target may be eventually reachable but not reachable under the current planning budget, and this finite-horizon distinction is exactly what temporal hard negatives teach the model.

Multi-step prediction and planning mismatch. Multi-step prediction has long been used to reduce the mismatch between learned dynamics and planning-time rollouts, for example, through overshooting objectives in latent dynamics models [17] and self-supervised future-representation prediction objectives [36, 30]. Reconstruction-free variants further show that predictive objectives can improve robustness by avoiding unnecessary pixel-level detail [9, 7]. However, multi-step prediction alone does not determine whether the latent distance used by a goal-conditioned planner reflects finite-horizon reachability. A model can roll out accurately and still assign low terminal distance to states separated by obstacles, irreversible dynamics, or manipulation bottlenecks. RC-aux therefore combines multi-horizon open-loop prediction with budget-conditioned reachability. The former aligns the learned dynamics with the open-loop rollouts used by the planner, while the latter shapes the latent space around horizon-indexed attainability.

B Planning-Alignment Analysis

This appendix formalizes the planning-alignment view of RC-aux. The goal is not to prove a complete performance guarantee for high-dimensional visual control. Instead, we isolate the two mismatches targeted by RC-aux: (i) the mismatch between local prediction training and open-loop planning, and (ii) the mismatch between Euclidean latent proximity and finite-budget reachability. We show that multi-horizon open-loop prediction controls planning-time cost distortion, that trajectory-induced temporal hard negatives identify the budget dependence of reachability, and that the reachability gate acts as a soft feasibility bias over the base latent planning cost.

B.1 Setup

Consider a deterministic controlled system

$$s_{t+1} = T(s_t, a_t), \qquad o_t = O(s_t), \tag{26}$$

with observation map O. A latent world model consists of an encoder e_θ and an action-conditioned latent predictor. We write

$$z_t = e_\theta(o_t). \tag{27}$$

For an action sequence $\tau = a_{t:t+H-1}$, let

$$\hat{z}_{t+1:t+H} \tag{28}$$

denote the open-loop latent rollout predicted by the model, and let

$$z^\star_{t+1:t+H} \tag{29}$$

denote the encoded latent sequence obtained by executing the same action sequence in the true environment and encoding the resulting observations. The base planner evaluates a latent goal cost

$$C_{\text{base}}(\hat{z}_{t+1:t+H}, z_g), \tag{30}$$

such as terminal latent distance, minimum distance over the rollout, soft-minimum distance, or mean distance.

B.2 Open-loop prediction controls planning-time cost distortion

The planner makes decisions using predicted rollouts. Therefore, the relevant prediction error is not only the one-step error, but the error in the open-loop latent sequence on which the planner evaluates costs.

Lemma 1 (Planning cost distortion from rollout error)**.** *Assume the base planning cost $C_{\text{base}}(\cdot, z_g)$ is L_C-Lipschitz with respect to the stacked rollout latent vector on the relevant latent region. Then, for any candidate action sequence τ,*

$$\left|C_{\text{base}}(\hat{z}_{t+1:t+H}, z_g) - C_{\text{base}}(z^\star_{t+1:t+H}, z_g)\right| \le L_C \left\|\hat{z}_{t+1:t+H} - z^\star_{t+1:t+H}\right\|_2 . \tag{31}$$

Proof. This follows directly from Lipschitz continuity of C_{base} in its rollout argument. □

On a sampled training segment, z_{t+k} is the encoded observation obtained after executing the sampled action sequence, and therefore corresponds to $z^\star_{t+k}$ for that segment. For that segment, the RC-aux multi-horizon loss is

$$\ell_{\mathrm{mh}} = \sum_{k=1}^{K} w_k \left\| \hat{z}_{t+k} - z_{t+k} \right\|_2^2 . \tag{32}$$

Assume $H \le K$ and define

$$w_{\min} = \min_{1 \le k \le H} w_k > 0. \tag{33}$$

Then

$$\sum_{k=1}^{H} \left\| \hat{z}_{t+k} - z^\star_{t+k} \right\|_2^2 \le \frac{1}{w_{\min}} \ell_{\mathrm{mh}}. \tag{34}$$

Combining this inequality with Lemma 1 yields

$$\left| C_{\mathrm{base}}(\hat{z}_{t+1:t+H}, z_g) - C_{\mathrm{base}}(z^\star_{t+1:t+H}, z_g) \right| \le \frac{L_C}{\sqrt{w_{\min}}} \sqrt{\ell_{\mathrm{mh}}}. \tag{35}$$

Thus, multi-horizon prediction directly controls an upper bound on the distortion of the cost that the planner actually evaluates.

Terminal-cost special case. If the planner uses the terminal squared latent distance,

$$C_{\mathrm{base}}(\hat{z}_{t+1:t+H}, z_g) = \| \hat{z}_{t+H} - z_g \|_2^2, \tag{36}$$

and all relevant latents satisfy $\|z\|_2 \le B$, then the terminal cost is $4B$-Lipschitz on this bounded region:

$$\begin{aligned} \left| \|x - z_g\|_2^2 - \|y - z_g\|_2^2 \right| &= |\langle x - y, x + y - 2z_g \rangle| \\ &\le \|x - y\|_2 \left(\|x\|_2 + \|y\|_2 + 2\|z_g\|_2 \right) \\ &\le 4B \|x - y\|_2 . \end{aligned} \tag{37}$$

B.3 Why one-step prediction is only an indirect control signal

A one-step prediction objective can be locally accurate while still leaving planning-time rollout errors uncontrolled. The following standard compounding argument makes this mismatch explicit.

Lemma 2 (Compounding of one-step error)**.** *Let $F^\star$ be the true latent dynamics and f_θ the learned one-step predictor. Assume f_θ is L_f-Lipschitz in the latent argument and the one-step prediction error is uniformly bounded on the rollout region:*

$$\| f_\theta(z, a) - F^\star(z, a) \|_2 \le \epsilon_1. \tag{38}$$

Then the K-step open-loop error satisfies

$$\| \hat{z}_{t+K} - z^\star_{t+K} \|_2 \le \epsilon_1 \sum_{i=0}^{K-1} L_f^i. \tag{39}$$

Proof. Let

$$e_k = \| \hat{z}_{t+k} - z^\star_{t+k} \|_2. \tag{40}$$

Then

$$\begin{aligned} e_{k+1} &= \left\| f_\theta(\hat{z}_{t+k}, a_{t+k}) - F^\star(z^\star_{t+k}, a_{t+k}) \right\|_2 \\ &\le \left\| f_\theta(\hat{z}_{t+k}, a_{t+k}) - f_\theta(z^\star_{t+k}, a_{t+k}) \right\|_2 + \left\| f_\theta(z^\star_{t+k}, a_{t+k}) - F^\star(z^\star_{t+k}, a_{t+k}) \right\|_2 \\ &\le L_f e_k + \epsilon_1. \end{aligned} \tag{41}$$

Since $e_0 = 0$, unrolling the recurrence gives the result. □

This lemma does not imply that one-step prediction is useless. Rather, it shows that one-step prediction controls planning only indirectly, while RC-aux trains directly on the open-loop quantities queried by the planner.

B.4 Finite-budget reachability as directed temporal geometry

Let

$$D^\star(s, g) = \inf_\tau \{h : T^h(s, \tau) = g\} \tag{42}$$

be the shortest hitting time from s to g, with $D^\star(s, g) = \infty$ if g is unreachable. In continuous environments, exact equality can be replaced by reaching an ϵ-ball around the goal. The ideal finite-budget reachability predicate is

$$R^\star_h(s, g) = \mathbf{1}[D^\star(s, g) \le h]. \tag{43}$$

This predicate is directed and budget-dependent. In general,

$$D^\star(s, g) \neq D^\star(g, s), \tag{44}$$

especially in systems with obstacles, irreversible dynamics, or contact constraints. Whenever the quantities are finite, $D^\star$ satisfies a directed triangle inequality:

$$D^\star(s, u) \le D^\star(s, g) + D^\star(g, u), \tag{45}$$

because a path from s to g can be concatenated with a path from g to u.

Thus, finite-horizon planning is naturally described by horizon-indexed sublevel sets of a directed temporal distance:

$$\{g : D^\star(s, g) \le h\}. \tag{46}$$

By contrast, Euclidean latent distance is symmetric and budget-free. RC-aux therefore learns a budget-conditioned predicate $R_\phi(z, z', h)$ rather than relying only on $\|z - z'\|_2$.

B.5 Trajectory-induced reachability labels

RC-aux does not assume access to $D^\star$. Instead, it uses observed trajectory offsets as empirical supervision. For a same-trajectory pair (z_i, z_j) with $i < j$, define the observed offset

$$\Delta = j - i. \tag{47}$$

The trajectory-induced finite-budget label is

$$y_{ijh} = \mathbf{1}[h \ge \Delta]. \tag{48}$$

These labels should be interpreted carefully. A positive label corresponds to a demonstrated feasible path within budget h along the sampled trajectory. A negative label with $h < \Delta$ should be interpreted as an empirical finite-budget negative under the observed trajectory distribution, rather than a certificate that the target is globally unreachable within h under the true MDP. A shorter unobserved path may exist, so the learned predicate is best understood as dataset-induced reachability unless additional coverage assumptions hold.

Let $D_{\mathcal{D}}(s, g)$ denote the shortest observed offset from s to g in the dataset, when such an offset exists. Then any observed segment is a feasible path, so

$$D^\star(s, g) \le D_{\mathcal{D}}(s, g). \tag{49}$$

If the behavior data is c-competitive in the sense that

$$D_{\mathcal{D}}(s, g) \le c\, D^\star(s, g), \tag{50}$$

then dataset reachability approximates true reachability up to a multiplicative slack. Specifically,

$$D_{\mathcal{D}}(s, g) \le h \Rightarrow D^\star(s, g) \le h, \tag{51}$$

while

$$D_{\mathcal{D}}(s, g) > h \Rightarrow D^\star(s, g) > h/c. \tag{52}$$

This formalizes the role of data coverage: trajectory-induced labels become closer to true finite-budget reachability as the offline trajectories become more complete and less suboptimal.

B.6 Bayes-optimal reachability under binary supervision

For a latent pair and budget, let $Y \in \{0, 1\}$ denote the trajectory-induced finite-budget label. The reachability head is trained with binary cross-entropy:

$$\mathcal{L}_{\text{bce}} = \mathbb{E}\left[-Y \log R_\phi(Z, Z', h) - (1 - Y) \log(1 - R_\phi(Z, Z', h))\right]. \tag{53}$$

Proposition 1 (Bayes-optimal reachability predictor)**.** *For any fixed representation, the minimizer of the binary cross-entropy risk is*

$$R^\star(z, z', h) = \mathbb{P}(Y = 1 \mid Z = z, Z' = z', h). \tag{54}$$

If the representation preserves the relevant temporal relation and the labels are deterministic, then for same-trajectory pairs with offset $\Delta = j - i$,

$$R^\star(z_i, z_j, h) = \mathbf{1}[h \geq \Delta]. \tag{55}$$

Proof. For a fixed input (z, z', h), let $p = \mathbb{P}(Y = 1 \mid z, z', h)$ and let q be the predicted probability. The conditional BCE risk is $-p \log q - (1 - p) \log(1 - q)$, whose pointwise minimizer is $q = p$. The deterministic-label case follows immediately. □

This proposition clarifies what the reachability head estimates: a conditional probability of empirical finite-budget reachability under the data-generating distribution.

B.7 Trajectory-induced temporal hard negatives identify the budget

The most important role of trajectory-induced temporal hard negatives is to make the budget variable identifiable.

Proposition 2 (Budget identifiability)**.** *Consider same-trajectory latent pairs (z_i, z_j) with temporal offset $\Delta = j - i$. If reachability supervision includes same-trajectory positives and batch negatives, but never includes same-trajectory pairs with insufficient budget, then there exists a classifier that fits all observed labels while ignoring the budget h on same-trajectory pairs. If trajectory-induced temporal hard negatives with $h < \Delta$ are included, then any classifier that fits both labels for the same ordered pair must depend on h.*

Proof. Without trajectory-induced temporal hard negatives, every observed same-trajectory pair used by the reachability loss is labeled positive, while batch negatives are labeled negative. A classifier can therefore predict positive for same-trajectory pairs and negative for cross-trajectory pairs, independent of h. This fits the labels without learning finite-budget reachability.

With trajectory-induced temporal hard negatives, the same ordered pair (z_i, z_j) can be labeled negative for budgets $h < \Delta$ and positive for budgets $h \geq \Delta$. Any classifier that fits both labels must assign different predictions to the same latent pair under different budgets. Therefore, it cannot ignore h. □

This result explains why hard negatives are not merely additional negative samples. They are the samples that identify the horizon dependence of the reachability predicate.

B.8 Reachability-aware planning as a soft feasibility bias

RC-aux optionally modulates the base latent planning cost with the trajectory-level reachability score

$$R(\tau, z_g) = \max_{1 \leq k < H} R_\phi(\hat{z}_{t+k}, z_g, H - k). \tag{56}$$

The planning cost is

$$C_{\text{RC}}(\tau) = C_{\text{base}}(\tau) \cdot \max(m, 1 - \lambda_{\text{plan}} R(\tau, z_g)), \tag{57}$$

where $m > 0$ is a small floor.

Ignoring the floor m for clarity, and assuming $0 \leq \lambda_{\text{plan}} < 1$, consider two trajectories τ_a and τ_b with base costs

$$d_a = C_{\text{base}}(\tau_a), \qquad d_b = C_{\text{base}}(\tau_b), \tag{58}$$

and reachability scores

$$R_a = R(\tau_a, z_g), \qquad R_b = R(\tau_b, z_g). \tag{59}$$

RC-aux prefers τ_a over τ_b when

$$d_a(1 - \lambda_{\text{plan}} R_a) < d_b(1 - \lambda_{\text{plan}} R_b). \tag{60}$$

Equivalently,

$$\frac{d_a}{d_b} < \frac{1 - \lambda_{\text{plan}} R_b}{1 - \lambda_{\text{plan}} R_a}. \tag{61}$$

If τ_a has high reachability $R_a \approx 1$ and τ_b has low reachability $R_b \approx 0$, then τ_a can be selected even if its base latent cost is larger, as long as

$$\frac{d_a}{d_b} < \frac{1}{1 - \lambda_{\text{plan}}}. \tag{62}$$

Thus, reachability-aware planning can prefer a slightly longer but empirically attainable trajectory over a deceptively short latent-space shortcut. When $\lambda_{\text{plan}} = 0$, the condition reduces to standard base-cost planning.

B.9 Predicted rollout pairs reduce planner-time distribution shift

The reachability head is queried during planning on predicted latents $\hat{z}$, not only on encoded dataset latents z. Training reachability only on encoded latents may leave the classifier poorly calibrated on the planner-induced rollout distribution. Predicted rollout pairs address this mismatch.

Let $r_\phi(z, z', h)$ be the reachability logit, and suppose that on encoded pairs it classifies with margin $\gamma > 0$:

$$y\, r_\phi(z, z', h) \geq \gamma, \qquad y \in \{-1, +1\}. \tag{63}$$

Assume r_ϕ is L_r-Lipschitz in its source latent. If a predicted latent satisfies

$$\|\tilde{z} - z\|_2 \leq \delta \tag{64}$$

and

$$L_r \delta < \gamma, \tag{65}$$

then replacing z by $\tilde{z}$ does not change the sign of the reachability prediction:

$$y\, r_\phi(\tilde{z}, z', h) > 0. \tag{66}$$

Thus, reliable reachability on planner-time rollouts requires the classifier to be robust on predicted latents. RC-aux trains this regime directly by applying reachability supervision to predicted rollout pairs. If stop-gradient is used on predicted latents, these terms calibrate the reachability head on planner-induced latents without directly backpropagating reachability labels into the rollout model. Without stop-gradient, they additionally propagate the reachability signal into the predicted rollout states.

B.10 Summary

The analysis supports the design of RC-aux in three ways. First, multi-horizon open-loop prediction controls the distortion of the latent costs evaluated by the planner. Second, budget-conditioned reachability learns empirical horizon-indexed reachable sets, and trajectory-induced temporal hard negatives are necessary to make the budget dependence identifiable. Third, the optional reachability-aware planner acts as a soft feasibility bias over the base latent cost, recovering the base planner when $\lambda_{\text{plan}} = 0$ and favoring empirically attainable rollouts when $\lambda_{\text{plan}} > 0$.

C Experimental Details

This appendix provides additional details for the experimental protocol used in Section 4, including task setup, baseline definitions, planner hyperparameters, and LIBERO-Goal training details.

Table 7: Planner and evaluation hyperparameters for the five-task pixel-control evaluation. Each local LeWM-family method is evaluated with five fixed groups of 50 episodes.

Task	CEM samples	CEM iters	Top-k	Budget	Goal offset / horizon	λ_{plan}
TwoRoom	300	30	30	50	25	0.35
Reacher	300	30	30	50	25	0.35
Push-T	300	30	30	50	25	0.35
Wall	600	20	60	60	$h = 8,\ rh = 4$	0.85
Cube	300	10	30	50	25	0.00

Task suite. The main evaluation uses five pixel-based goal-conditioned control tasks: TwoRoom, Reacher, Push-T, Wall, and Cube. Each episode provides an initial observation and a goal image. The agent executes actions from the initial observation, and success is determined by whether the resulting trajectory reaches the specified goal within the task horizon. TwoRoom evaluates navigation in a top-down room environment. Reacher evaluates goal-conditioned reaching. Push-T evaluates object pushing from image observations. Wall evaluates obstacle-constrained planning, where Euclidean closeness in latent space can be misleading if the goal is not reachable through a direct path. Cube evaluates goal-conditioned 3D manipulation.

Evaluation protocol. We report the success rate as the primary metric. For local LeWM-family runs, each method is evaluated on five fixed evaluation groups with 50 episodes per group, and we report the mean and group-level standard deviation. The same fixed groups are used across local LeWM-family comparisons, enabling paired episode-level analyses in Appendix E. Previously reported baseline results are included as benchmark references in the main comparison table, while local LeWM-family rows are reported with group-level standard deviations.

Baselines. We compare RC-aux with prior baselines and LeWM-family controls. The prior baselines include reconstruction-free visual world models and latent-dynamics methods such as DINO-WM [47] and PLDM [39], goal-conditioned imitation learning such as GCBC [13], and offline value-learning methods such as IQL [23] and IVL [33]. When available, we also include task-specific references such as DINO-WM+prop on Push-T. The LeWM-family rows provide the closest controlled comparison: LeWM is the original backbone baseline, LeWM-cont is a continuation-trained control using the same backbone, and RC-aux augments LeWM [29] with the proposed reachability-correction objective and reachability-aware planner.

LeWM-family controls. For all five tasks, LeWM-cont serves as the primary comparator, accounting for additional training under the same backbone. The matched Δ row in Table 1 is computed relative to LeWM-cont.

Planner and evaluation hyperparameters. Table 7 lists the main planner and evaluation hyperparameters. All local LeWM-family results use the same fixed evaluation grouping described above. For Wall, we use the tuned planner horizon configuration used for that environment. For Cube, the planner reachability weight is set to zero in the reported configuration.

LIBERO-Goal OFT setup. For the LIBERO-Goal extension, we evaluate all 10 LIBERO-Goal tasks using the official success checker with 50 evaluation episodes per task and a maximum horizon of 600 environment steps. The policy uses two RGB views, `agentview_rgb` and `eye_in_hand_rgb`, and trains an OFT-style action chunk head on top of the learned representation. The main matched comparison uses trainable encoders and the no-repeat all-task setting for both LeWM and RC-aux. We additionally report a repeat-tuned RC-aux result as the best observed RC-aux setting.

D LIBERO-Goal Extension Details

This appendix reports per-task LIBERO-Goal results for the OFT-style action-head extension. The training and evaluation setup is described in Appendix C. The mean-level results are summarized in Section 4.5; here we provide the task-level breakdown. Table 9 reports per-task success rates. Under

Table 8: Training and evaluation settings for the LIBERO-Goal OFT-style extension.

Setting	Value
Tasks	LIBERO-Goal, tasks 0–9
Evaluation episodes	50 per task
Maximum steps	600
Observation views	`agentview_rgb`, `eye_in_hand_rgb`
Action head	OFT-style chunk head
Chunk length	8
Action horizon	8
Batch size	32
Learning rate	5×10^{-5}
Weight decay	1×10^{-4}
Precision	bfloat16
Loss	ℓ_1 action loss
Main comparator	Trainable, no-repeat

the matched no-repeat protocol, RC-aux improves or matches LeWM on all 10 tasks, with the largest gains on tasks 0, 8, and 9. The repeat-tuned setting further improves weaker tasks, especially task 5.

Table 9: Per-task LIBERO-Goal success rates. The matched comparison is between LeWM trainable no-repeat and RC-aux trainable no-repeat. Repeat-tuned RC-aux and OpenVLA-OFT 7B [21, 22] are included as additional references.

Method	T0	T1	T2	T3	T4	T5	T6	T7	T8	T9	Mean
LeWM, trainable no-repeat	0.64	0.78	0.70	0.76	0.90	0.44	0.60	0.94	0.70	0.66	0.712
RC-aux, trainable no-repeat	0.92	0.86	0.80	0.78	0.96	0.48	0.70	0.96	0.86	0.80	0.812
RC-aux, repeat-tuned	0.94	0.86	0.76	0.88	0.92	0.84	0.74	1.00	0.88	0.82	0.864
OpenVLA-OFT 7B	0.98	0.92	0.96	0.86	1.00	1.00	1.00	1.00	1.00	0.98	0.970

E Additional Quantitative Results

This appendix provides additional quantitative evidence for the results in Section 4. We include paired fixed-episode outcomes and local LeWM-family success visualizations.

E.1 Paired Fixed-Episode Outcomes

To verify that the aggregate gains are not caused by differences in evaluation episodes, we compare LeWM and RC-aux on the same fixed evaluation episodes. Table 10 reports episode-level outcome counts. RC-aux solves many episodes missed by LeWM, especially on Wall and TwoRoom. On Wall, RC-aux-only successes outnumber LeWM-only successes by 85 to 2. Push-T is the most balanced task, consistent with the main results.

Table 10: Paired fixed-episode outcomes for LeWM and RC-aux. Each entry counts evaluation episodes in one of four categories: both methods fail, only LeWM succeeds, only RC-aux succeeds, or both methods succeed.

Task	Both fail	LeWM only	RC-aux only	Both succeed
TwoRoom	4	1	24	221
Reacher	15	17	32	186
Push-T	10	13	14	213
Wall	39	2	85	124
Cube	54	6	15	175

Figure 4 visualizes the same paired outcomes over the fixed evaluation groups. The visual pattern highlights that the largest aggregate gains correspond to many RC-aux-only successes on the same evaluation episodes.

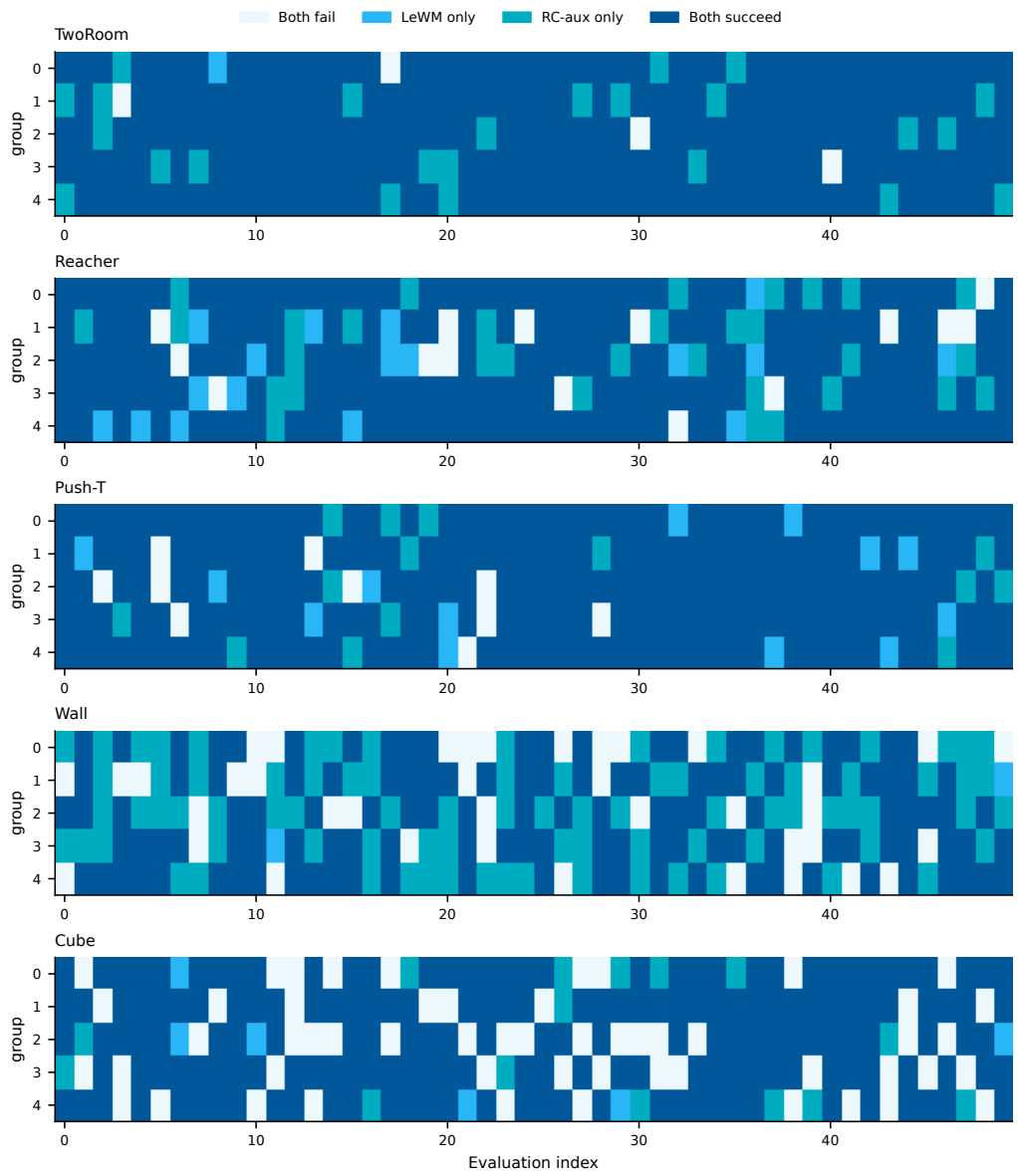

Figure 4: Paired outcomes on fixed evaluation episodes. Each cell corresponds to one evaluation episode. RC-aux-only successes are most frequent on Wall and TwoRoom, while Push-T remains nearly balanced between LeWM-only and RC-aux-only successes.

E.2 Local LeWM-Family Success Visualizations

Figure 5 shows the local LeWM-family success rates with group-level standard deviations. This visualization isolates the comparison among LeWM, continuation-trained LeWM, and RC-aux, complementing the broader baseline comparison in the main text. For all five tasks, the matched comparator is LeWM-cont, consistent with the matched deltas reported in Table 1.

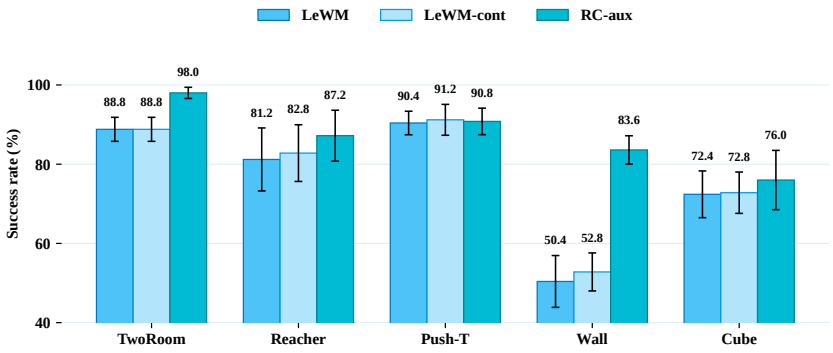

Figure 5: Local LeWM-family success rates. Bars show mean success over five fixed evaluation groups, and error bars show group-level standard deviation. For matched controlled comparisons, RC-aux is compared against LeWM-cont across all five tasks.

F Qualitative Results

F.1 Paired Wall and Cube Comparisons

Figure 6 shows paired Wall rollouts. Wall provides a visually clear example of obstacle-constrained reachability: latent proximity to the goal can be misleading when the direct path is blocked. RC-aux produces rollouts that more consistently follow attainable paths toward the goal.

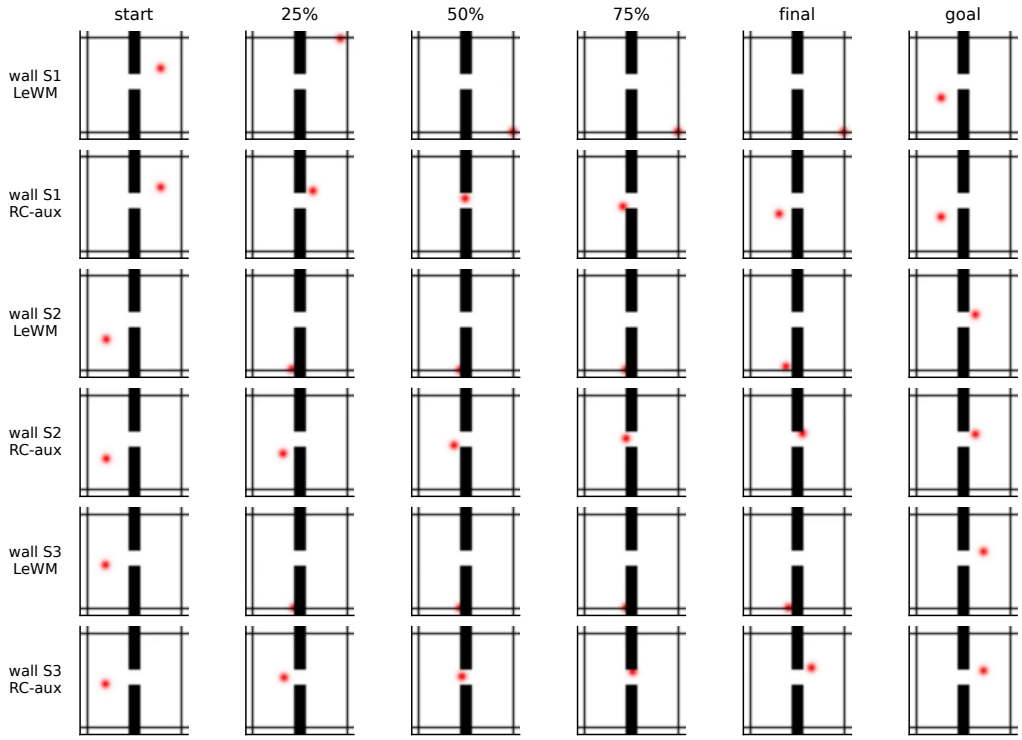

Figure 6: Paired Wall rollouts comparing LeWM and RC-aux. Wall highlights obstacle-constrained reachability, where visually or latently nearby states may not be attainable within the planning budget.

Figure 7 provides the full Cube comparison. This figure complements the main-text qualitative result by showing additional Cube rollouts under the same paired comparison format.

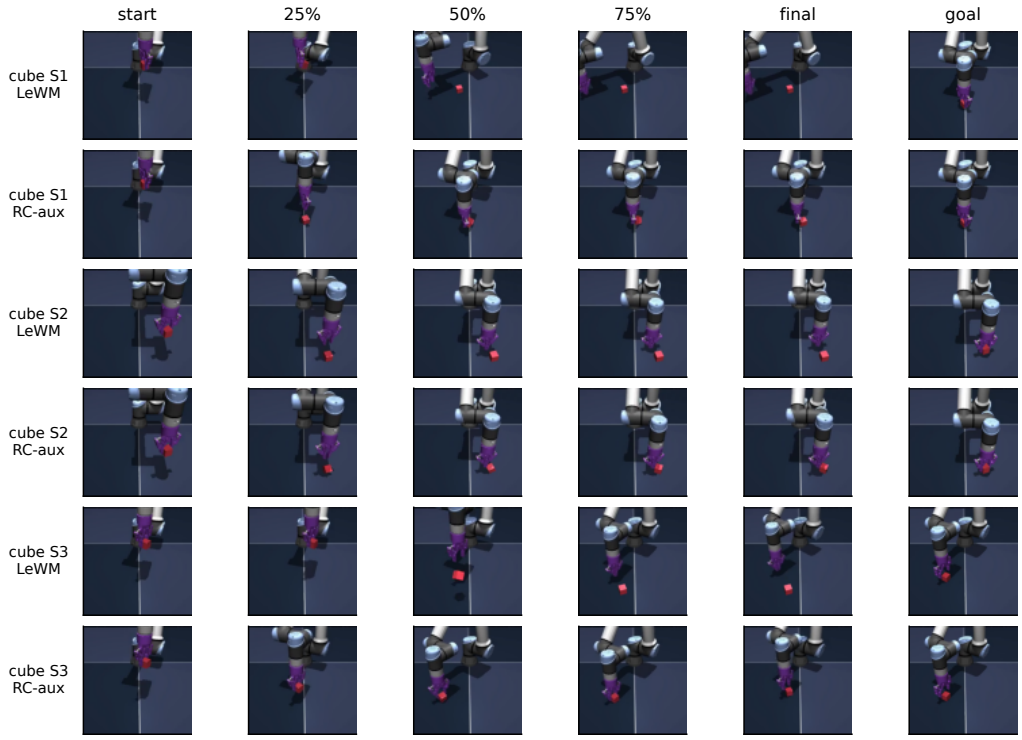

Figure 7: Full paired Cube rollouts comparing LeWM and RC-aux. The figure provides additional examples beyond the compact Cube visualization in the main text.

F.2 Additional RC-aux Success Rollouts

Figures 8–12 show additional successful RC-aux rollouts across the five tasks. These examples illustrate the visual diversity of the benchmark and the types of goal-conditioned behavior produced by the RC-aux latent planner.

F.3 Approximate Pixel-Space Trajectory Overlays

Figure 13 provides approximate pixel-space trajectory overlays for Wall and TwoRoom. The trajectories are extracted from rendered videos by tracking the visible object or agent center, and should therefore be interpreted as qualitative visualizations rather than simulator ground-truth state trajectories.

G Latent and Physical Diagnostics

This appendix provides additional diagnostics of the learned latent representation. These analyses complement the success-rate results by examining latent terminal distances, reachability scores, and physical probing on Push-T.

G.1 Latent Reachability Diagnostic

Figure 14 visualizes selected Wall and TwoRoom rollouts in a two-dimensional PCA projection of the corresponding model latents. The figure is diagnostic rather than a matched quantitative benchmark: each panel uses the corresponding model encoder and PCA projection. The high-dimensional terminal latent distances reported in the titles provide a more reliable comparison.

Figure 15 summarizes terminal latent distances on the selected diagnostic examples. The RC-aux rollouts end closer to the goal latents in these examples, consistent with the paired-outcome and qualitative rollout results.

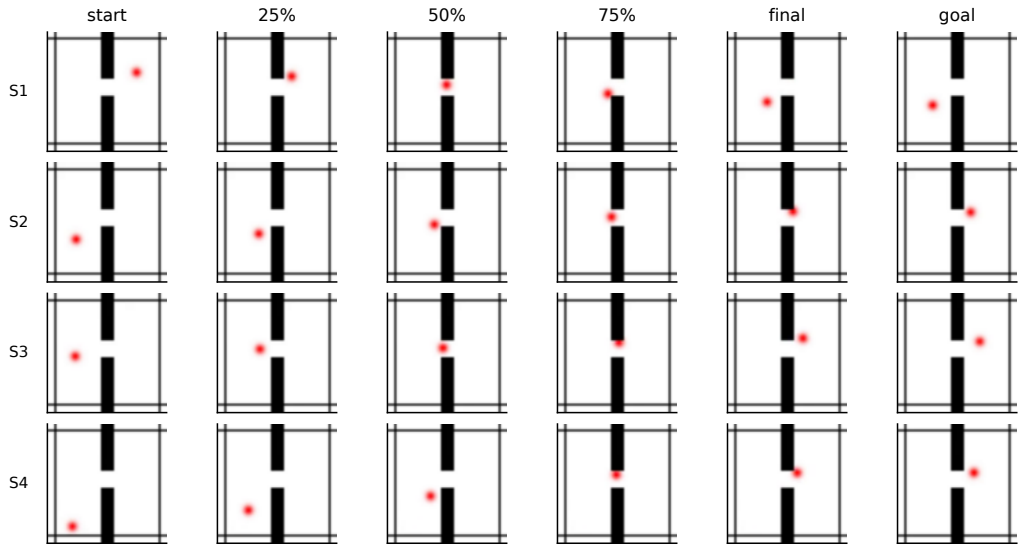

Figure 8: Additional successful RC-aux rollouts on Wall.

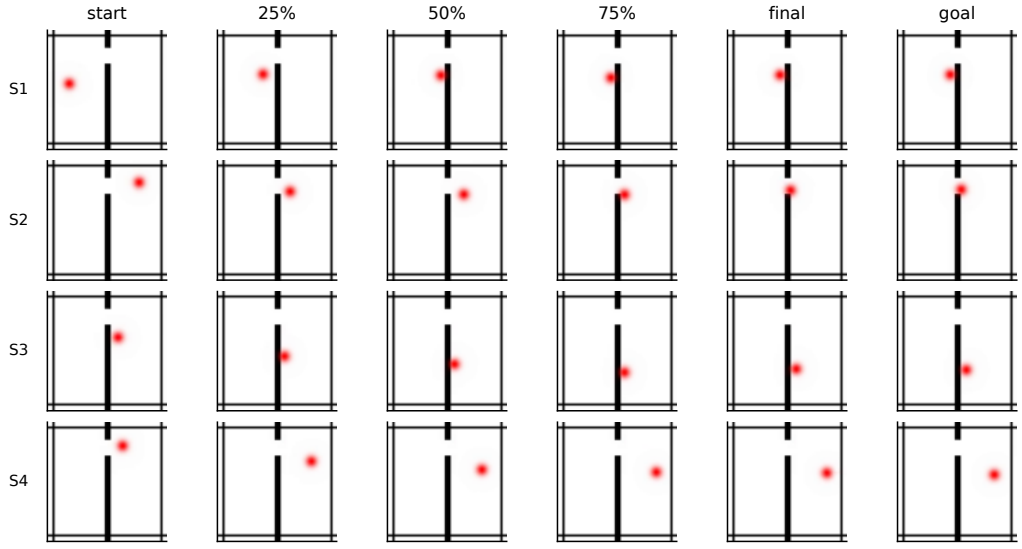

Figure 9: Additional successful RC-aux rollouts on TwoRoom.

G.2 Push-T Physical Probe

We further evaluate whether RC-aux preserves physical information in the latent representation on Push-T. Table 11 summarizes local probe results for representative physical quantities. RC-aux is comparable to LeWM across these probes, suggesting that the reachability correction does not discard basic physical information.

Figure 16 visualizes the Push-T linear-probe MSE comparison.

H Broader Impacts

The work is foundational research on latent world models for planning. Potential positive impacts include improving the reliability and efficiency of model-based planning from pixels, while potential risks include inappropriate deployment of learned planning systems in real-world robotic settings

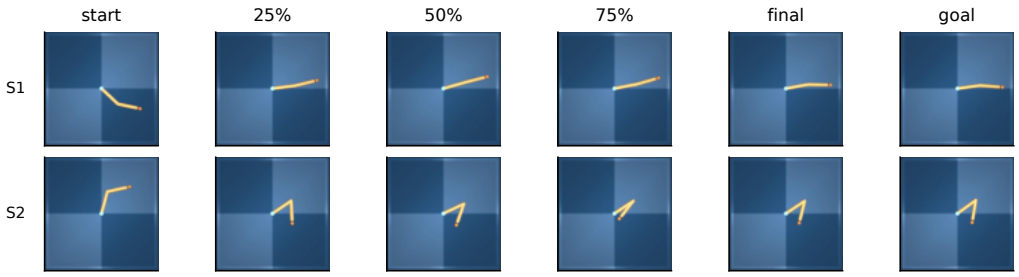

Figure 10: Additional successful RC-aux rollouts on Reacher.



Figure 11: Additional successful RC-aux rollouts on Push-T.

without adequate validation, safety constraints, or uncertainty handling. The paper frames the method as an offline benchmark study rather than a deployed autonomous system.

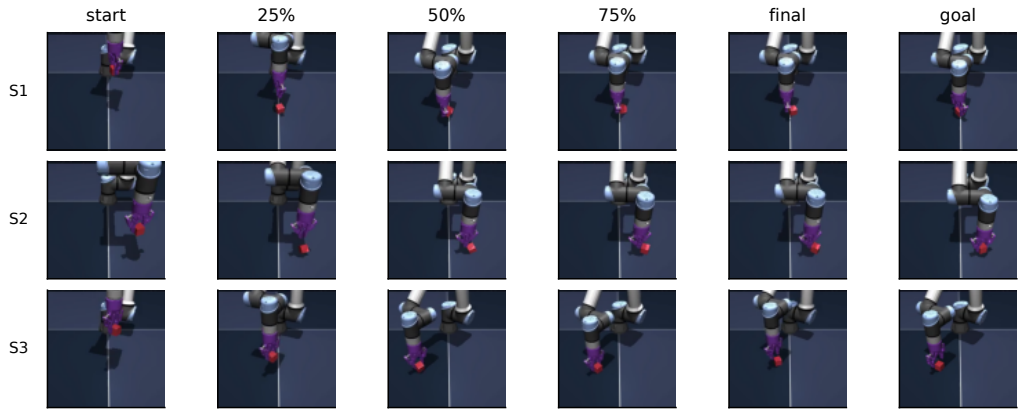

Figure 12: Additional successful RC-aux rollouts on Cube.

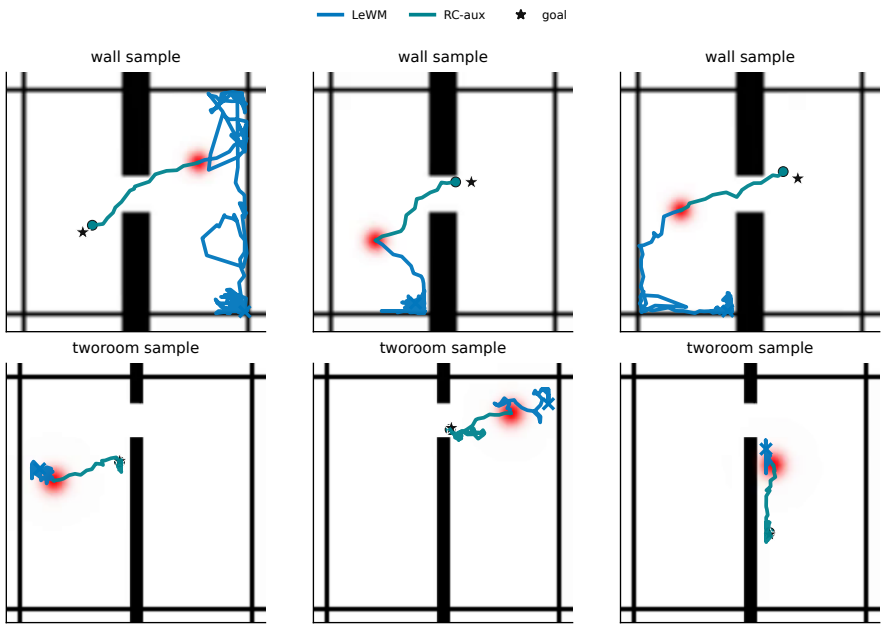

Figure 13: Approximate pixel-space trajectory overlays. Paths are extracted from rendered rollout videos by color tracking. The visualization is qualitative and is intended to illustrate the trajectory-level behavior of LeWM and RC-aux.

Table 11: Push-T physical probe summary. Lower MSE is better. The results indicate that RC-aux preserves physical information in the latent representation.

Target	Probe metric	LeWM	RC-aux
Agent location	Linear MSE	0.039	0.037
Block location	Linear MSE	0.021	0.020
Block angle	Linear MSE	0.170	0.157
Block angle	MLP MSE	0.017	0.020

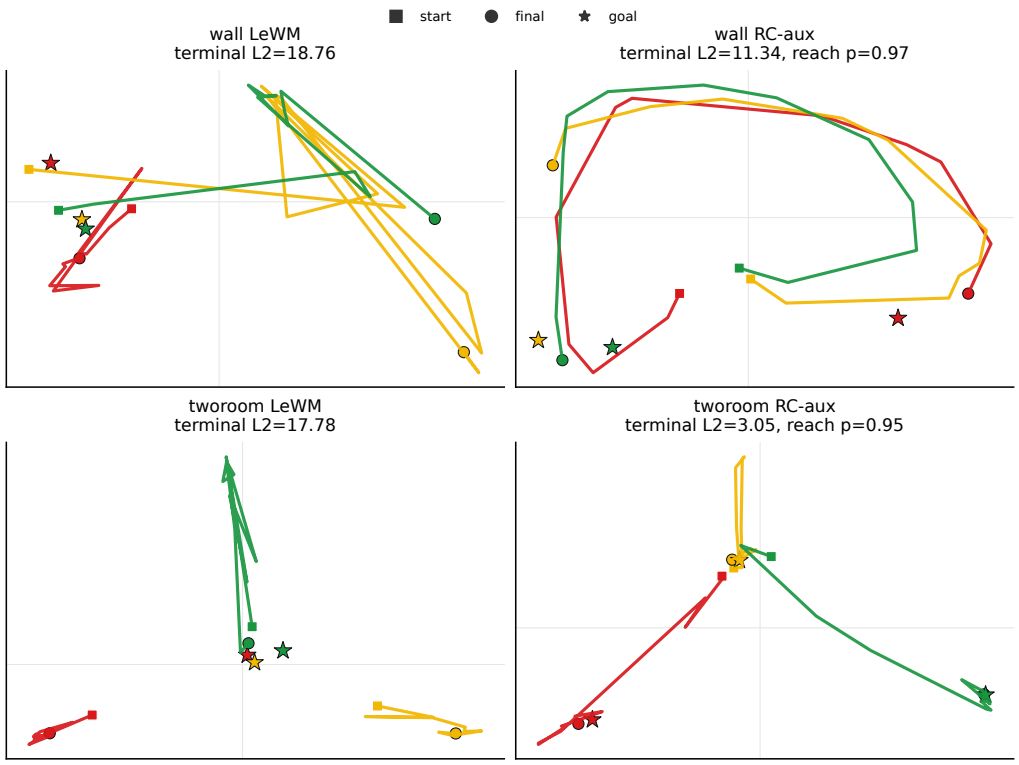

Figure 14: Latent reachability diagnostic on selected Wall and TwoRoom rollouts. Rollout frames and goals are encoded by the corresponding model and projected to two dimensions by PCA for visualization. Squares, circles, and stars denote start, final, and goal latents. The diagnostic suggests that RC-aux terminal latents are closer to the encoded goals and are assigned high reachability by the learned reachability head.

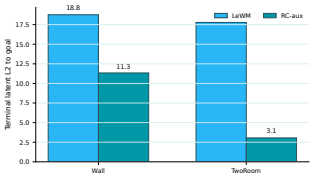

Figure 15: Terminal latent distance summary for selected diagnostic rollouts. Lower distance indicates that the final rollout latent is closer to the encoded goal latent.

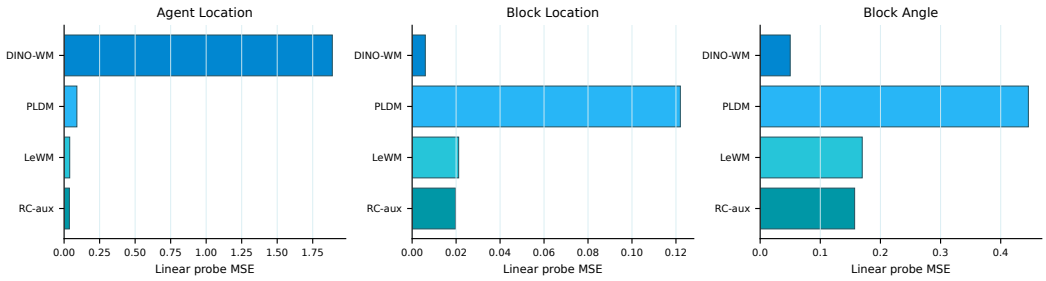

Figure 16: Push-T physical probe with linear predictors. RC-aux remains comparable to LeWM on physical quantities, indicating that reachability supervision does not remove basic physical structure from the learned latents.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state that RC-aux improves the plannability of reconstruction-free latent world models by aligning open-loop prediction and finite-horizon reachability with downstream planning. These claims are supported by the method formulation in Section 3, the planning-alignment analysis in Appendix B, and the controlled empirical results in Section 4 and Appendices D–F.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 discusses trajectory-derived supervision, soft feasibility correction, and the scope of the empirical evaluation.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper provides the formal planning-alignment analysis in Appendix B. The assumptions for the rollout-error and cost-distortion results are stated explicitly, and the corresponding lemmas and derivations are provided to justify how multi-horizon prediction, budget-conditioned reachability, and temporal hard negatives align the training objective with finite-horizon planning queries.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The method is fully specified in Section 3, including the multi-horizon loss, reachability supervision, training objective, and planner scoring rule. Section 4 describes the evaluation tasks and main protocols, while Appendix C provides training, evaluation, planner, and LIBERO-Goal implementation details needed to reproduce the main experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code is not included in the initial anonymous submission. The paper provides the full algorithmic formulation and detailed experimental settings in the main text and appendices, and we plan to release an anonymized implementation and reproduction instructions upon acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 specifies the evaluated goal-conditioned control tasks, baselines, matched LeWM-family comparisons, success-rate metric, and compute setting. Appendix C further provides the training and evaluation configurations, planner settings, and LIBERO-Goal OFT-style action-head details.

   Guidelines:

   - The answer [Yes] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

Justification: The main LeWM-family results are reported as mean and standard deviation over five fixed evaluation groups. The paper also includes paired fixed-episode comparisons in Appendix E to verify that the aggregate improvements are not due to differences in evaluation episodes.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section 4 states that all experiments were conducted on a single NVIDIA RTX A6000 Ada GPU. The paper further reports parameter footprint and controlled planner cost-call timing in Section 4.4 and Appendix E, including the overhead of RC-aux relative to LeWM and the scoring-time comparison with DINO-WM-S/14.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The study uses existing simulated control and robot-manipulation benchmarks and does not involve human-subject research, sensitive personal data, or high-risk data collection. The authors have reviewed the NeurIPS Code of Ethics and made efforts to preserve anonymity throughout the submission.

   Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The work is foundational research on latent world models for planning. Potential positive impacts include improving the reliability and efficiency of model-based planning from pixels, while potential risks include inappropriate deployment of learned planning systems in real-world robotic settings without adequate validation, safety constraints, or uncertainty handling. The paper frames the method as an offline benchmark study rather than a deployed autonomous system.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a high-risk pretrained language model, image generator, scraped dataset, or similarly dual-use asset. The proposed method is an auxiliary objective and planner modification for latent world models evaluated on established control benchmarks, so special release safeguards are not applicable.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper cites the existing benchmarks, models, and baselines used in the experiments, including LeWorldModel, DINO-WM, PLDM, and LIBERO-Goal. The experiments are based on established research assets rather than newly scraped or repackaged datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset or benchmark asset. The proposed contribution is a training objective and planner family for latent world models; the implementation will be documented when released.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

Justification: This study does not involve crowdsourcing experiments, human-subject experiments, or human participant data collection.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: This study does not involve human-subject research or crowdsourcing and therefore does not require IRB approval or equivalent review.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core methodology does not involve large language models. RC-aux is based on latent world-model training, multi-horizon open-loop prediction, budget-conditioned reachability supervision, and reachability-aware planning; therefore, no LLM usage declaration is required for the scientific method.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.