BEYOND SINGLE GOAL IMAGES: LEARNING GOAL METRICS FOR LEWORLDMODEL PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

A single goal image shows both task-relevant information and incidental visual details, although many visually different outcomes may solve the same task. We extend LeWorldModel (LEWM) to plan toward sets of successful outcomes by separating predictive representations from goal-specific cost evaluation. We freeze the encoder and action-conditioned latent dynamics and learn a goal projector, used as either a linear map or a nonlinear network. The distance between a projected latent vector and the mean of projected goal examples defines the planning cost. Training combines compactness of successful examples with distribution regularization, temporal ordering along goal-reaching trajectories, and preferences for demonstrated action sequences over corrupted alternatives. The resulting cost integrates naturally into the existing latent model-predictive-control procedure without modifying the world model. Across three world-model seeds, the nonlinear projector achieves mean success rates of 90.8%, 100.0%, and 52.3% on Push-T, Two-Rooms, and OGBench-Cube, respectively. It surpasses both single-goal-image LeWorldModel planning and latent 1-nearest-neighbor goal-set planning, improving over the latter by 15.2, 26.7, and 10.0 percentage points under the same evaluation protocol. These results support learning task-specific planning costs while preserving a shared predictive representation. Source code will be made available.

1 INTRODUCTION

World models compress observations into a latent state and predict how actions change that state, allowing control by planning in imagination (Ha & Schmidhuber, 2018; Hafner et al., 2023; Robine et al., 2023). A particularly simple approach for visual control is goal-image conditioning. Given a current observation x_t and an image x_g of the desired future, the planner searches for an action sequence where the final predicted latent should resemble the encoding of $z_g = f_\phi(x_g)$. LeWorldModel (LEWM) follows this approach while learning its encoder f_ϕ and latent dynamics g_ϕ jointly from reward-free offline trajectories (Maes et al., 2026). Consequently, the learned model can be reused for many start and goal pairs. Visual control has previously combined action-conditioned prediction with MPC in pixel or latent space, while other methods learn task-relevant distances for image-goal planning (Finn & Levine, 2017; Hafner et al., 2019; Srinivas et al., 2018; Tian et al., 2020).

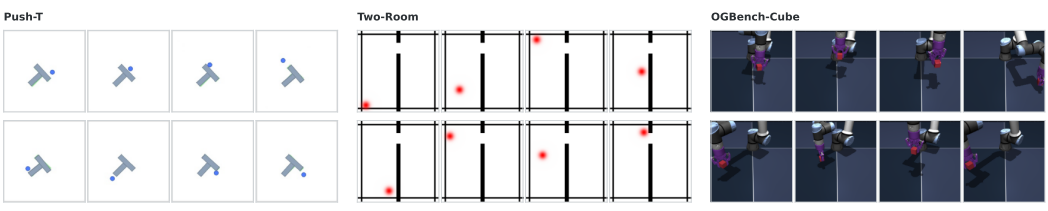

Figure 1: **Many Images can Satisfy a Goal.** In Push-T, agent position varies after the T-object is aligned. In Two-Room, the agent may occupy different locations in the left room. In OGBench-Cube, object and arm poses vary while the cube is lifted.

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

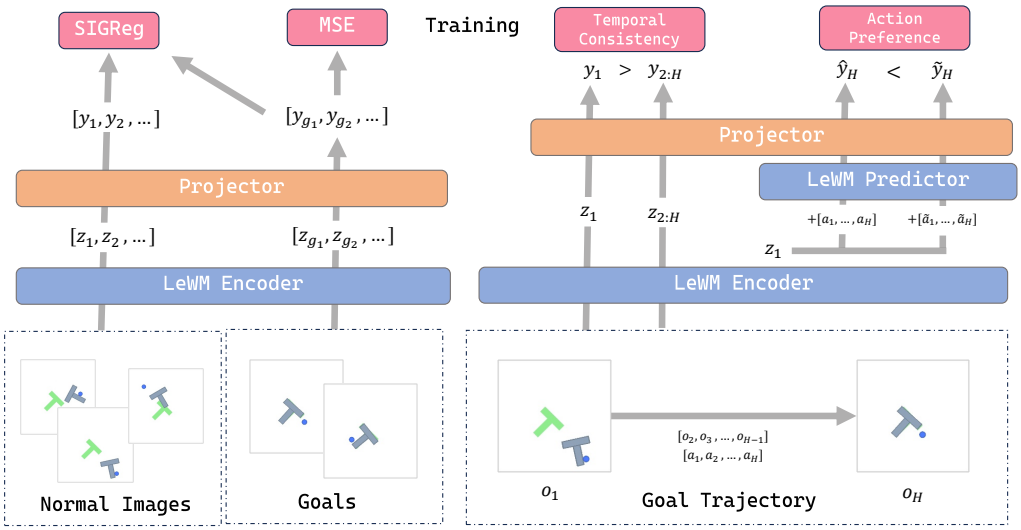

Figure 2: **Training Objectives.** We train the goal projector with four objectives. MSE brings goal representations toward their mean $\frac{1}{N}\sum_{i=1}^{N} y_{g_i}$ which becomes the goal vector. Normal images prevent representation collapse via SIGReg. Then, goal trajectories are used to structure the goal space. Temporal consistency encourages lower cost (distance to goal vector) for later states in goal-reaching trajectories and action-preference learning favors predicted outcomes of demonstrated action sequences over perturbed alternatives. The notation $y_1 > y_2$ denotes that y_1 should have a higher distance to the goal than y_2. LEWM components remain frozen during optimization.

An image, however, is usually more specific than a task. "*Push an object to the target location*" constrains the object's position and angle but ultimately not the agent's final location. "*Pick up the object*" constrains the object's height but allows many joint configurations and object positions. In each case, many visually distinct states are equally successful. Related approaches represent tasks through goal sets or distributions, or through collections of successful outcomes, rather than using a single target state (Conkey & Hermans, 2020; Nasiriany et al., 2021; Pavlasek et al., 2023; Eysenbach et al., 2021). Matching the future exactly to one goal image asks the planner to reproduce both the task-relevant variables but also incidental details of that given example.

This motivates our core design principle: *Plan in latent space but evaluate cost in goal space*. Prior task-aware representation methods often modify the predictive or control representation itself (Nair et al., 2020; Hansen-Estruch et al., 2022). Our separation instead preserves the predictive latent space and learns only a goal-specific geometry. Because for planning, environment details still matter for accurately predicting action consequences. A goal space, however, should instead discard details that are not relevant for the specific task. We therefore freeze the LEWM encoder and latent dynamics.

Contributions. We make three contributions:

- We formulate goal-tasks as a separation between latent and goal space representations. A shared latent world model predicts future latents, while a goal space identifies equivalent outcomes and tracks progress toward success.
- We introduce a LEWM adapter that evaluates cost using goal images and environment trajectories. In total, the extensions include a learned projector network, an MSE loss objective, SIGReg regularization, two dynamics-based losses and one goal vector.
- We provide a paired evaluation over three environments and three independently trained world models, comparing the learned representation with vanilla image-conditioned LEWM, latent 1-, 3-, and 5-NN, a latent mean estimator, and a randomly initialized projector. We extend success rates with per-episode outcomes and report additional ablations.

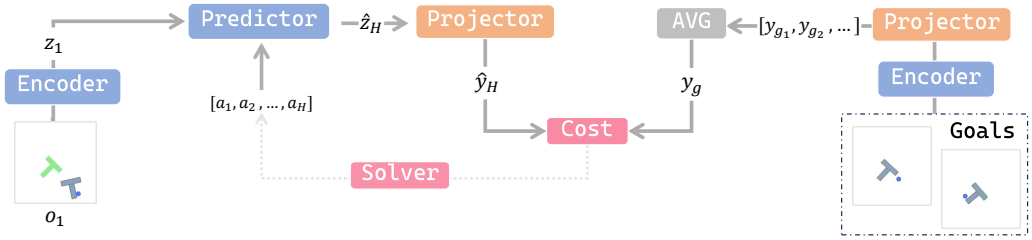

Figure 3: **Latent Planning and Cost Estimation.** LEWM gets an initial image o_1 and action sequence $[a_1, \dots, a_H]$ and then predict the future's latent outcome $\hat{z}_H$. This vector is projected to the goal space as $\hat{y}_H$ and compared to the mean vector y_g of projected goal images.

2 Related Work

World models and latent planning. World models learn predictive environment dynamics and use the model for policy learning or online trajectory optimization (Ha & Schmidhuber, 2018; Hafner et al., 2023). Transformer-based latent models can learn effective dynamics from comparatively small datasets (Micheli et al., 2022; Robine et al., 2023). Visual planning methods have additionally learned representations or distances suited to reaching image goals, including Universal Planning Networks and MBOLD (Srinivas et al., 2018; Tian et al., 2020). LEWM is a reward-free joint-embedding predictive architecture whose encoder and action-conditioned predictor are trained end-to-end with SIGReg (Maes et al., 2026). In contrast to methods that adapt the predictive representation, we freeze LEWM completely and learn only how predicted states should be evaluated.

Goal-conditioned control and goal sets. Goal-conditioned policies and value functions commonly condition on a state, image, or learned goal embedding (Schaul et al., 2015; Andrychowicz et al., 2017; Nasiriany et al., 2019). Related work has generalized this approach to goal distributions or sets (Conkey & Hermans, 2020; Nasiriany et al., 2021; Pavlasek et al., 2023), while successful-outcome examples have been used to infer rewards or control objectives (Xie et al., 2018; Eysenbach et al., 2021). We likewise assume that many observations can satisfy one goal, but instead learn which variation within the success set should be ignored. The resulting goal set is compressed to a task-specific metric and one center, while the original MPC procedure remains unchanged.

Task-relevant metric learning. Task-aware representations can suppress distinctions irrelevant to a goal, for example through goal-aware prediction or goal-conditioned abstractions (Nair et al., 2020; Hansen-Estruch et al., 2022). Our linear projectors are closely related to metric learning from equivalence constraints. Its terminal cost is a positive-semidefinite Mahalanobis distance centered at the latent goal mean (Bar-Hillel et al., 2005; Globerson & Roweis, 2005). Compacting successful examples alone admits a trivial constant representation, so we employ SIGReg, following the broader invariance/anti-collapse principle used in VICReg-style representation learning (Bardes et al., 2021; Balestriero & LeCun, 2025). The key distinction is that the predictive latent space remains shared and task-agnostic, while only the evaluation space is goal-specific.

3 Goal-Set Invariance

Let $s \in \mathcal{S}$ denote the environment state, $x = o(s)$ its image observation, and $i : \mathcal{S} \to \{0, 1\}$ the task's binary success indicator. The goal is the set of all successful states,

$$\mathcal{G} = \{s \in \mathcal{S} : i(s) = 1\}. \tag{1}$$

A goal image $x_g = o(s_g)$ represents only one member $s_g \in \mathcal{G}$. It shows one successful outcome, but does not specify which of its visual details are required by the task. Within $\mathcal{G}$, any variation that leaves the success indicator i unchanged is irrelevant to success. All elements of $\mathcal{G}$ should therefore be treated as interchangeable outcomes even when their images look different.

Desired goal space geometry. Let $z = f_\phi(o(s))$ denote the latent of state s under a frozen world-model encoder. Image-conditioned planning targets the single point z_g. For a set-valued goal, we

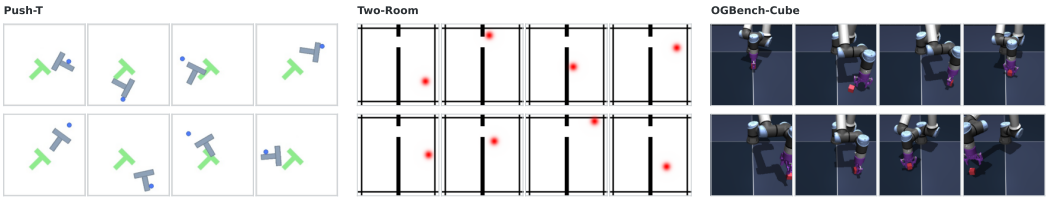

Figure 4: **Evaluation.** Episode starting positions from the evaluation suites are selected for diversity. Push-T varies agent and block position. Two-Room starts lie in the right room, and OGBench-Cube starts contain a resting, ungrasped object.

instead seek a task-specific mapping h_θ for which

$$s_i, s_j \in \mathcal{G} \quad \Longrightarrow \quad h_\theta(z(s_i)) \approx h_\theta(z(s_j)). \tag{2}$$

Which variations should disappear is itself goal-dependent. A difference that is irrelevant for one goal set can determine membership in another. Throughout, h_θ denotes the learned projector specifically for the current set $\mathcal{G}$. A constant function would satisfy Eq. 2 trivially, however, so representations of ordinary states must also keep variation.

4 Method

First, we start with a short recap on the underlying world model. LeWM is trained from reward-free sequences of images and actions. An encoder f_ϕ maps an RGB observation to a latent $z_t = f_\phi(x_t)$, and an action-conditioned predictor models the next embedding,

$$\hat{z}_{t+1} = g_\phi(z_{\leq t}, a_{\leq t}). \tag{3}$$

Encoder and predictor are jointly trained with next-latent mean-squared error and SIGReg regularization on encoder embeddings (Maes et al., 2026). At control time, LeWM performs latent model-predictive control with the Cross-Entropy Method (CEM) (Rubinstein & Kroese, 2004). Given a planning horizon H, CEM samples action sequences $a_{0:H-1}$. The predictor rolls each sequence forward from the current latent to the final prediction $\hat{z}_H$, which is scored against the encoding of a provided goal image. We refer to this score as the *cost*, defined as

$$C_{\text{LeWM}}(z, x_g) = \frac{1}{d_z}\|z - f_\phi(x_g)\|_2^2. \tag{4}$$

Therefore, its latent space predicts action-conditioned futures and also measures their distance to the goal image. Architecture and planning details of the original LeWM pipeline, that we also use, are given in Appendix A. We introduce the goal-space extension next.

4.1 Learning a goal-space projector

We freeze the LeWM encoder f_ϕ and action-conditioned predictor g_ϕ. Let $\mathcal{D}_g$ and $\mathcal{D}_x$ denote the distributions of goal latents and ordinary image latents, respectively. We learn a goal-specific projector $h_\theta : \mathbb{R}^{d_z} \to \mathbb{R}^{d_g}$, implemented as either a linear projector or an MLP. Its goal center and planning cost are

$$y_g = \mathbb{E}_{z^+ \sim \mathcal{D}_g}[h_\theta(z^+)], \qquad C_{\text{goal}}(z) = \frac{1}{d_g}\|h_\theta(z) - y_g\|_2^2. \tag{5}$$

In the following, we distinguish image-based objectives, which learn similarity among outcomes, from trajectory-based objectives, which consider progress and action preferences towards the goal.

Image-based Objectives. We contract the representations of successful outcomes by

$$\mathcal{L}_{\text{compact}} = \frac{1}{d_g}\mathbb{E}_{z^+ \sim \mathcal{D}_g}\left[\|h_\theta(z^+) - y_g\|_2^2\right]. \tag{6}$$

Compactness alone allows a constant representation. We therefore apply SIGReg to the distribution of projected ordinary image latents,

$$\mathcal{L}_{\text{SIGReg}}(\theta) = \mathbb{E}_{Z\sim\mathcal{D}_x}[\text{SIGReg}(h_\theta(Z))]\,. \tag{7}$$

Following the anti-collapse principle (Bardes et al., 2021; Balestriero & LeCun, 2025), these objectives contract successful outcomes while encouraging variation over the broader training distribution. Both objectives use unordered images through the frozen encoder. They require neither actions nor dynamics predictions. Ordinary frames are not labeled as failures and may include successful states.

Temporal Ordering. Goal compactness does not necessarily specify how cost should behave before successful states are reached. We additionally use trajectories $\tau = (z_{0:H}, a_{0:H-1})$ from training episodes that contain goal examples at the end. In addition, rolling out the frozen predictor under the provided actions generates latents $\hat{z}_{1:H}$, with $\hat{z}_0 = z_0$.

For trajectory- and dynamics-based losses, we use the cost function C_{goal} itself as the learning signal. First normalize it by its median value on ordinary states:

$$\widetilde{C}_{\text{goal}}(z) = \frac{C_{\text{goal}}(z)}{\max\left(\text{median}_{u\sim\mathcal{D}_x} C_{\text{goal}}(u),\, \epsilon\right)}, \qquad \epsilon > 0. \tag{8}$$

The goal center and normalization factor are treated as constants when differentiating the trajectory-based losses. Let $\mathcal{P}$ contain temporal pairs (i, j) with $i < j$. For a distance margin $m_t > 0$ and $[u]_+ = \max(0, u)$, we encourage later states in a trajectory to get lower cost:

$$\mathcal{L}_{\text{time}} = \mathbb{E}_\tau \sum_{(i,j)\in\mathcal{P}} \Big([m_t + \widetilde{C}_{\text{goal}}(z_j) - \widetilde{C}_{\text{goal}}(z_i)]_+ + \; [m_t + \widetilde{C}_{\text{goal}}(\hat{z}_j) - \widetilde{C}_{\text{goal}}(\hat{z}_i)]_+ \Big). \tag{9}$$

The first term considers the actual observed temporal order. The second additionally constrains the dynamics predictions scored during planning. This ordering is a progress assumption about recorded behavior, rather than a guarantee of progress. Therefore, data quality is important.

Action-preference Ordering. Planning evaluates the quality difference between different action sequences from the same state. For each given trajectory, we construct K alternative action sequences $\tilde{a}^{(k)}_{0:H-1}$ and roll them out from z_0. Let $\hat{z}^{\text{demo}}_H$ be the predicted endpoint under the real action sequence and $\hat{z}^{(k)}_H$ the endpoint under alternative k. For a distance margin $m_p > 0$, we encourage a lower cost for the real action sequence

$$\mathcal{L}_{\text{pref}} = \mathbb{E}_{\tau,\tilde{a}^{(1:K)}} \frac{1}{K} \sum_{k=1}^{K} [m_p + \widetilde{C}_{\text{goal}}(\hat{z}^{\text{demo}}_H) - \widetilde{C}_{\text{goal}}(\hat{z}^{(k)}_H)]_+. \tag{10}$$

Alternatives are action sequences sampled from other training clips or noise-perturbed versions of the actual sequence. The loss assumes that real action sequences are preferable to these alternatives and the loss does not require externally annotated preferences or verified failure labels.

Combined objective. Together, the projector parameters θ are optimized as

$$\mathcal{L}_{\text{goal}} = \underbrace{\mathcal{L}_{\text{compact}} + \lambda\mathcal{L}_{\text{SIGReg}}}_{\text{image-based}} + \underbrace{\lambda_{\text{time}}\mathcal{L}_{\text{time}} + \lambda_{\text{pref}}\mathcal{L}_{\text{pref}}}_{\text{trajectory- and dynamics-based}}. \tag{11}$$

Setting $\lambda_{\text{time}} = \lambda_{\text{pref}} = 0$ recovers projector training from unordered images alone. After training, we compute y_g over the given goal set. The frozen LEWM planner rolls candidate actions forward in its native latent space and evaluates predictions using Eq. 5. Inference still requires only the projector network and one goal center y_g.

4.2 Planning and Comparison Costs

Starting from $z_t = f_\phi(x_t)$, LEWM autoregressively predicts the final latent for each candidate action sequence. Every trajectory remains in the native latent space, and only its prediction $\hat{z}_H$ is mapped to goal space for scoring. We optimize the cost from Eq. 5 with the Cross-Entropy Method (CEM) (Rubinstein & Kroese, 2004) inside the original LEWM latent MPC pipeline. All goal representations share the same frozen model, episode starts, solver settings, and random seeds. The details are reported in Appendix A. We compare the learned projector with four cost alternatives:

1. **Vanilla LEWM** uses the native cost C_{LeWM} from Eq. 4 and the goal image x_g supplied by each evaluation episode, like in the original LEWM paper. This image is a concrete successful future observation and therefore a goal $s \in \mathcal{G}$.
2. **Latent k-NN** has access to the complete goal dataset $\mathcal{D}_g$. For cost evaluation, let $\mathcal{N}_k(z) \subset \mathcal{D}_g$ be the set of its k closest goal latents in Euclidean distance. We average these neighbors into a local latent center and measure the distance to it as

$$\mu_k(z) = \frac{1}{k} \sum_{z^+ \in \mathcal{N}_k(z)} z^+, \qquad C_{k\text{NN}}(z) = \frac{1}{d_z} \|z - \mu_k(z)\|_2^2. \tag{12}$$

   We evaluate $k \in \{1, 3, 5\}$. For $k = 1$, the cost is just the minimum MSE to any latent goal.
3. **Latent mean** also uses D_g and computes the center over all goal latents directly

$$\bar{z}_g = \mathbb{E}_{\mathcal{D}_g}[z^+], \qquad C_{\text{z-mean}}(z) = \frac{1}{d_z} \|z - \bar{z}_g\|_2^2. \tag{13}$$

   This cost method checks whether averaging in latent and goal space makes a difference.
4. **Random projector** evaluates an untrained goal projector. Cost follows Equation 5.

5 EVALUATION

We use three continuous-control datasets from the original LEWM study[1]. Figure 4 shows representative evaluation starts, while Figure 1 shows some goal images. Simulator information is only used to identify goal images and their respective episodes and later for measuring success during evaluation. Planning still uses RGB images only. The following success criteria apply to every method:

Push-T. The goal is to cover the green T-shaped target with the movable gray T-shaped object. Goal images are training frames with at most 100 visible green pixels, yielding approximately 1,500 images/episodes. An evaluation succeeds when the object position error is below 20 pixels and the circular angle error is below $\pi/9$ (20 degrees). These thresholds are taken from the original LEWM eval, but success now no longer depends on the agent's final position which is simply ignored.

Two-Room. The goal is to enter the left room. A frame is considered a goal image when the agent's horizontal coordinate satisfies $x < 112$. We sample 1,000 diverse goal frames. An evaluation succeeds at any position with $x < 112$ after starting in the right room, where $x \geq 112$ and y coordinate is arbitrary.

OGBench-Cube. The goal is to pick up the red cube. A goal image requires object height above 0.06 m, gripper contact above 0.5, and object–end-effector distance below 0.06 m. We use 1,000 goal frames that vary in object and robot-arm position, while all images are showing the red cube lifted above the ground. Evaluation succeeds when the same height and distance thresholds are met and a relaxed contact threshold of 0.1 is reached. However, all three conditions need to hold for fifteen consecutive environment steps (or three from the perspective of LeWM).

5.1 TRAINING AND EVALUATION PROTOCOL

For every dataset, we train three LEWM models with different seeds and learn a linear and MLP projector for each model. At first, the linear model only uses compactness and SIGReg loss to demonstrate the performance when no dynamics information is available. Ablations will introduce the dynamics loss for it later again. Each projector network has a dataset-specific hyperparameter configuration which is shared over all seeds. Appendix A lists these settings and Appendix C compares our LEWM models to the original paper results. In addition, we tune the projector's hyperparameters on 50 training episodes (these trajectories are part of the projector's training data) per dataset that are distinct from the evaluation episodes. All reported planning runs use the same evaluation episodes (not in the projector's dataset, but previously used for LEWM training) and are exactly paired across methods. They have the same start positions, goal offsets, action budget, and CEM randomness. We select start positions algorithmically as good as possible to get diverse positions and orientations, as shown in Figure 4, while no start satisfies the goal conditions in advance.

[1]The official datasets are available at https://huggingface.co/collections/quentinll/lewm.

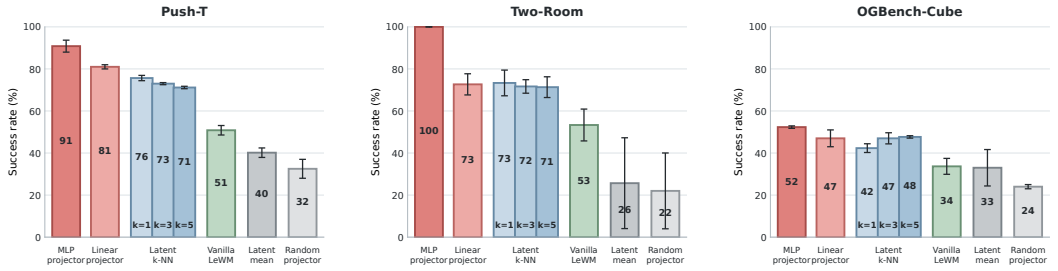

Figure 5: **Goal Success Rates.** Success rates averaged over three seeds with standard deviation intervals. All methods utilize the goal-set for cost estimation except Vanilla LEWM that uses the particular goal image given by the evaluation episode (as in the original paper). In this figure, the linear projector uses only compactness and SIGReg loss.

Push-T uses 200 diverse starts selected by block position, block angle and agent location. Two-Room uses 100 right-room locations whose paired future enters the left room. OGB-Cube uses 100 different starts with a resting object (height at most 0.03 m and contact below 0.1), and initial gripper to object distance at most 0.10 m. We average successes across the three seeds and report standard deviation.

6 EXPERIMENTS

Figure 5 presents the main comparison. At first, it shows that the learned projectors do more than averaging goal latents and the transformation to a learned goal space is genuinely useful. The latent mean cost reaches 40.2%, 25.7%, and 33.0% on Push-T, Two-Room, and OGB-Cube, while an untrained projector reaches 32.5%, 22.0%, and 24.0%. Both trail the trained linear projector at 81.0%, 72.7%, and 47.0% respectively and the difference is even more significant for the MLP variant. The improvement therefore comes from learning dynamics and which latent directions encode task-irrelevant variation, not merely from taking a global mean or using a projection.

Overall, the MLP projector reaches 90.8%, 100% and 52.3% on Push-T, Two-Room, and OGB-Cube, respectively. The increase in success rate for Push-T and Two-Room is noticeable while being slightly better in OGB-Cube. The remaining OGB-Cube failures may reflect limitations in both predictive accuracy of LEWM and the learned cost. Our present experiments do not isolate these factors.

The linear projector still reaches 81.0%, 72.7% and 47.0% success rates. It surpasses all k-NN baselines by at least 5.3 percentage points in Push-T while being tied in performance for Two-Room and OGB-Cube. Push-T is the clearest evidence for the intended invariance. Training collapses agent positions of the aligned block into a representation that remains useful for planning. The cost landscape remains intact without additional dynamics losses and operates solely on compactness and SIGReg.

Vanilla image-conditioned LEWM reaches 50.8% on Push-T, 53.3% on Two-Room, and 33.7% on OGB-Cube. These success rates are objectively lower than those reported in the original LEWM evaluation, because our eval has a different specification. First, a goal image is 50 environment steps after the start position rather than 25, because the goal criteria are often not reached after 25 steps. And more importantly, we select only evaluation trajectories whose future observation satisfies the abstract goal. The original evaluation asks the controller to reach a later observation from the same trajectory, without requiring that this final observation solves a particular abstract goal (Maes et al., 2026). Our protocol is therefore both longer-horizon and goal-constrained. However, considering the original

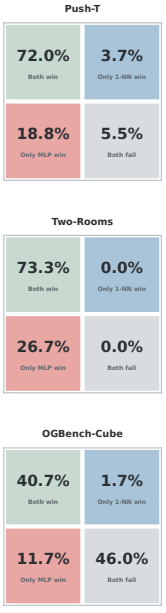

Figure 6: **MLP vs. 1-NN.** Overlapping and non-overlapping success rates.

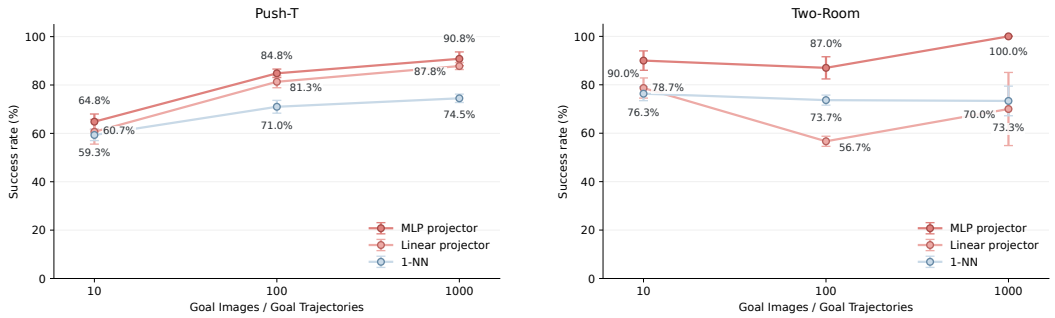

Figure 8: **Effect of Goal-data Budget on Planning Success in Push-T (left) and Two-Room (right).** We vary the number of goal images / trajectories available to each training run. The MLP and the linear projector use the full objective. Points show mean success rates across three world-model seeds, with error bars indicating standard deviation.

LEWM eval, our trained models reach very comparable results in their setup. Please see Appendix C for reproduction details. Under our shared protocol, the learned projector improves on vanilla LEWM by $+40.0$, $+46.7$, and $+18.6$ percentage points, respectively. Thus, even though the given image shows a goal state for Vanilla LEWM, reaching that particular image to fulfill the goal requirements is less effective.

7 Further Ablations and Analysis

Paired Outcomes. The bar chart in Figure 5 can hide whether two costs solve the same episodes. Therefore, Figure 6 compares paired binary outcomes between 1-NN and the MLP projector. On Push-T, 72.0% of episodes are solved by both methods, 18.8% only by the projector, 3.7% only by 1-NN, and 5.5% by neither. Also, Two-Room shows a noticeable advantage while OGB-Cube is slightly better.

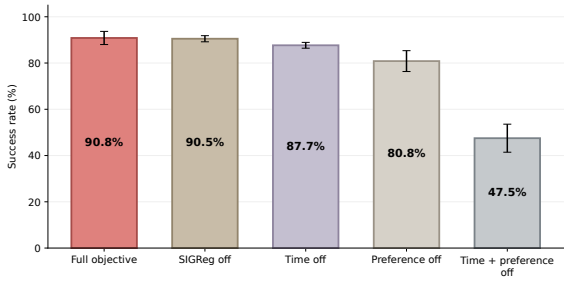

Figure 7: **Effect of Individual Loss Terms on Push-T Planning Success**. We compare the MLP projector's full objective with variants that disable SIGReg, temporal ordering, action-preference learning, or both temporal ordering and preference learning. Compactness remains enabled throughout and the eval stays identical.

Ablations on Loss Terms. Further, Figure 7 characterizes the loss objective benefits. It indicates that action-preference learning makes the largest contribution among the individual auxiliary losses. Removing it reduces mean success from 90.8% to 80.8%, whereas removing temporal ordering results in a smaller decrease to 87.7%. Removing SIGReg has little effect on mean Push-T success under the full objective, suggesting that the trajectory-based losses provide sufficient structure in this setting. In contrast, removing both trajectory losses reduces success to 47.5%. Although, Figure 5 shows that linear projectors work with only compactness and SIGReg.

Covariance Ablation. To test whether SIGReg provides benefits beyond preserving second-order variation, we replace it with a closed-form covariance constraint for the linear projector:

$$\min_{W} \operatorname{tr}(W\Sigma_g W^\top) \quad \text{s.t.} \quad W(\Sigma_x + \lambda_{\text{ridge}} I)W^\top = I, \tag{14}$$

whose solution uses the d_g smallest eigenvectors of $\Sigma_g v = \gamma(\Sigma_x + \lambda_{\text{ridge}} I)v$. Sweeping across the three world-model seeds results in $\lambda_{\text{ridge}} = 10^{-4}$. This covariance-only variant reaches 71.3% ± 4.73

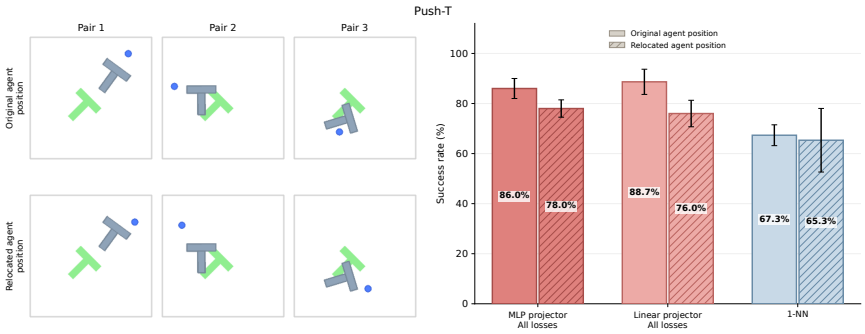

Figure 10: **Robustness to Agent-position Interventions in Push-T.** Left: three example pairs with original (top) and relocated (bottom) agent positions. Right: success rates for 1-NN and linear and MLP projectors trained with the full objective. Bars show means across three world-model seeds. Each seed is evaluated on 50 pairs.

on Push-T, compared with 81.0% for SIGReg, indicating that second-order preservation alone does not recover the same representation as hypothesized by Balestriero & LeCun (2025).

Robustness to Agent-Position Interventions. To evaluate robustness to agent-position changes, we directly modify the Push-T simulator to construct 50 paired initial states, preserving the block pose, goal configuration, and agent-to-block distance while resetting velocities to zero. Figure 10 shows three example pairs and planning success across three world-model seeds. Relocation reduces success from 86.0% to 78.0% for the full-objective MLP and from 88.7% to 76.0% for the full-objective linear projector. Although the MLP shows a smaller decrease, the difference is modest. The 1-NN baseline changes less (67.3% to 65.3%) but performs worse in both conditions. These results support robustness under the tested relocations, while acknowledging that agent position can affect control difficulty even when the goal-relevant object configuration remains unchanged.

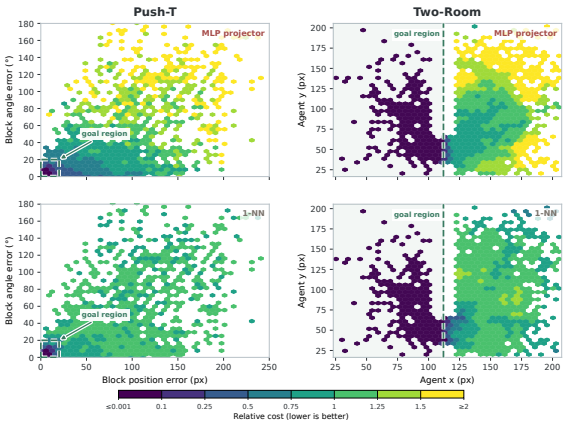

Figure 9: **Cost Landscapes for Push-T (left) and Two-Room (right).** Hexagons show median costs, normalized by each method's median non-goal cost. Discrete color bands indicate increasing relative cost, while dashed boundaries mark the goal region. The MLP shows a more smooth decrease in cost.

Planning-cost Landscape. Figure 9 compares normalized MLP and 1-NN costs over task-relevant state coordinates. Both assign low costs within the goal region, while the MLP shows a more pronounced increase farther from it. This suggests stronger task-relevant structure in the learned representation.

8 Conclusion

We extended LeWM to plan toward sets of successful outcomes by learning a goal-specific projector while keeping the encoder and dynamics frozen. Across three environments and three world-model seeds, the full-objective MLP goal projector surpasses single-goal-image and latent k-NN planning under a shared protocol. Push-T ablations highlight the importance of trajectory-based objective functions for learning useful planning costs. These results support separating predictive representations from task-specific goal metrics. This opens a path toward reusing a single world model across diverse tasks, each defined by examples of what success can look like.

AI USE STATEMENT

In this work, we used generative AI tools for aid and polish writing, for retrieval and discovery, for research execution (implement methods) and to draft sections of the paper.

We have not used generative AI tools for generating synthetic datasets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, design or provide feedback on research methodology or experiments, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis or interpret results.

Additionally, we used generative AI tools for some figure generation. We have reviewed all AI-assisted work. We checked LLM-generated figures for their correct data usage. LLM-generated code was verified and tested (with LLM-generated tests) for correctness by 1 author. Human-generated research ideas were executed by LLM agents. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

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

Aharon Bar-Hillel, Tomer Hertz, Noam Shental, Daphna Weinshall, and Greg Ridgeway. Learning a mahalanobis metric from equivalence constraints. *Journal of machine learning research*, 6(6):937, 2005.

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

Adam Conkey and Tucker Hermans. Planning under uncertainty to goal distributions. *arXiv preprint arXiv:2011.04782*, 2020.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Ben Eysenbach, Sergey Levine, and Russ R Salakhutdinov. Replacing rewards with examples: Example-based policy search via recursive classification. *Advances in Neural Information Processing Systems*, 34:11541–11552, 2021.

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

Amir Globerson and Sam Roweis. Metric learning by collapsing classes. *Advances in neural information processing systems*, 18, 2005.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3):440, 2018.

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

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

Philippe Hansen-Estruch, Amy Zhang, Ashvin Nair, Patrick Yin, and Sergey Levine. Bisimulation makes analogies in goal-conditioned reinforcement learning. In *International Conference on Machine Learning*, pp. 8407–8426. PMLR, 2022.

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.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. *arXiv preprint arXiv:2209.00588*, 2022.

Suraj Nair, Silvio Savarese, and Chelsea Finn. Goal-aware prediction: Learning to model what matters. In *International conference on machine learning*, pp. 7207–7219. PMLR, 2020.

Soroush Nasiriany, Vitchyr Pong, Steven Lin, and Sergey Levine. Planning with goal-conditioned policies. *Advances in neural information processing systems*, 32, 2019.

Soroush Nasiriany, Vitchyr H Pong, Ashvin Nair, Alexander Khazatsky, Glen Berseth, and Sergey Levine. Disco rl: Distribution-conditioned reinforcement learning for general-purpose policies. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6635–6641. IEEE, 2021.

Jana Pavlasek, Stanley Robert Lewis, Balakumar Sundaralingam, Fabio Ramos, and Tucker Hermans. Ready, set, plan! planning to goal sets using generalized bayesian inference. In *7th Annual Conference on Robot Learning*, 2023.

Jan Robine, Marc Höftmann, Tobias Uelwer, and Stefan Harmeling. Transformer-based world models are happy with 100k interactions. *arXiv preprint arXiv:2303.07109*, 2023.

Reuven Y Rubinstein and Dirk P Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning*, volume 133. Springer, 2004.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pp. 1312–1320. PMLR, 2015.

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*, pp. 4732–4741. PMLR, 2018.

Stephen Tian, Suraj Nair, Frederik Ebert, Sudeep Dasari, Benjamin Eysenbach, Chelsea Finn, and Sergey Levine. Model-based visual planning with self-supervised functional distances. *arXiv preprint arXiv:2012.15373*, 2020.

Annie Xie, Avi Singh, Sergey Levine, and Chelsea Finn. Few-shot goal inference for visuomotor learning and planning. In *Conference on Robot Learning*, pp. 40–52. PMLR, 2018.

A LeWM and Planning Implementation Details

Original LeWM Backbone. We rebuild the LeWM implementation used in the original paper (Maes et al., 2026). Observations are resized to 224×224 pixels and encoded by a 12-layer ViT with patch size 14, three attention heads, and hidden width 192 (Dosovitskiy et al., 2020). A linear BatchNorm projection maps the encoder's class token to the $d_z = 384$ latent used by the loss and planner. The action-conditioned predictor has six Transformer layers, 16 attention heads, and 10% MLP dropout. It is trained jointly with the encoder using next-latent mean-squared error and SIGReg ($\lambda = 0.1$), exactly as in LeWM. Our goal-set experiments freeze this complete model. No encoder, predictor, or world-model training objective is changed. For each environment, we use three independently trained world-model checkpoints.

Linear Projector without Dynamics Losses. The linear projector output dimension d_g is 256 for Push-T and 128 for Two-Room and OGBench-Cube, also summarized in Table 1. SIGReg uses 1,024 random unit directions and averages their Epps–Pulley statistics. Goal and ordinary frames are sampled independently, and the loss $\mathcal{L}_{\text{compact}}$ uses the empirical mean of the current goal minibatch. We optimize the projector with AdamW using batches of 48 ordinary frames and 48 goal frames, a learning rate of 5×10^{-5} with cosine decay, weight decay 0.01, and $\beta = (0.9, 0.95)$. The goal set D_g contains N_g images and after training, the goal center y_g is computed once over the complete goal dataset D_g.

Table 1: **Linear Goal-projector Configurations without Dynamics Losses**. Each row is fixed across all seeds.

Dataset	d_g	Steps	N_g	SIGReg λ
Push-T	256	4,000	1,000	0.015
Two-Room	128	1,000	1,000	0.14
OGBench-Cube	128	1,500	1,000	0.05

Goal Projectors with Full Objective. We implement h_θ as a linear or as a three-layer MLP with dimensions $384 \to 512 \to 512 \to 256$. The two hidden layers use $\tanh$ activations. All layers use biases, and we use neither normalization layers nor dropout. The projector is randomly initialized, while the LeWM encoder and predictor remain frozen.

Each update samples independently, 48 ordinary frames, 48 goal frames, and 48 trajectory clips. Clips are short windows within a goal trajectory and are sampled randomly. We train for 5,000 updates using AdamW with learning rate 5×10^{-5}, cosine decay, weight decay 0.01, and $\beta = (0.9, 0.95)$. After training, the goal center is recomputed as the mean projected representation of the complete selected goal set.

Table 2: **Projector Training Configurations**. All environments use the same architecture, output dimension $d_g = 256$, and 5,000 optimization updates. Each configuration is fixed across the three world-model seeds.

Dataset	λ_{SIGReg}	λ_{time}	λ_{pref}
Push-T	0.015	1	1
Two-Room	0.015	10^{-2}	10^{-2}
OGBench-Cube	0.015	10^{-4}	10^{-4}

The training objective is

$$\mathcal{L} = \mathcal{L}_{\text{compact}} + \lambda_{\text{SIGReg}}\mathcal{L}_{\text{SIGReg}} + \lambda_{\text{time}}\left(\mathcal{L}_{\text{time}}^{\text{obs}} + \mathcal{L}_{\text{time}}^{\text{pred}}\right) + \lambda_{\text{pref}}\mathcal{L}_{\text{pref}}. \tag{15}$$

Temporal ordering is applied to both observed and predicted six-state clips spanning five latent transitions. We compare state-index pairs $\{(0,3),(0,4),(0,5),(1,4),(1,5),(2,5)\}$, encouraging later states to have lower cost with a distance margin of 0.05. The action-preference loss uses the same margin to favor the predicted endpoint of the demonstrated action sequence over four alternatives

rolled out from the same initial latent. Specifically, two action sequences are randomly sampled from the training dataset and two noisy versions of the demonstrated sequence are generated. Gaussian perturbations use standard deviations 0.5 and 1.0 in normalized action space.

For both auxiliary losses, costs are measured relative to the current goal-minibatch center and divided by the median ordinary-minibatch cost, clamped below at 10^{-4}. The center and normalization factor are detached from the gradient computation. Table 2 reports the environment-specific loss weights.

Latent Planning. We keep the original LEWM CEM-based MPC pipeline and change only the cost function. Therefore in every comparison, CEM samples 300 action sequences per iteration, runs for 30 iterations, and keeps 30 elites. The planning horizon is five latent transitions. As in LEWM, frame skip groups five primitive actions into each latent transition. The complete five-block plan is executed before replanning, so the 50-step evaluation budget allows at most two planning rounds. All methods use these identical settings, while only the goal cost listed in Section 4 differs. Each start is paired with a qualifying dataset goal 50 primitive steps later, and every controller has a budget of 50 actions.

B SEED-LEVEL RESULTS

Table 3: Success rate (%) for every world-model seed. All methods within a dataset are evaluated on identical episodes. Vanilla LEWM uses the goal image supplied by the episode.

Dataset	Seed	MLP	Linear	Vanilla	1-NN	3-NN	5-NN	Mean	Random
Push-T	1	94.0	82.0	53.0	74.5	73.0	71.5	40.0	28.0
	2	90.0	81.0	51.0	77.0	73.5	71.5	42.5	32.5
	3	88.5	80.0	48.5	75.5	72.5	70.5	38.0	37.0
Two-Room	1	100.0	68.0	48.0	80.0	73.0	69.0	3.0	7.0
	2	100.0	78.0	50.0	72.0	74.0	77.0	28.0	17.0
	3	100.0	72.0	62.0	68.0	68.0	68.0	46.0	42.0
OGBench-Cube	1	52.0	51.0	32.0	43.0	45.0	47.0	28.0	24.0
	2	52.0	47.0	31.0	40.0	46.0	48.0	43.0	23.0
	3	53.0	43.0	38.0	44.0	50.0	48.0	28.0	25.0

C ADDITIONAL REPRODUCIBILITY DETAILS

Table 4 presents the reproduction quality of our own LEWM implementation compared to the reported performance of Maes et al. (2026). The table uses the same evaluation protocol that is applied for the original work and shows comparable results.

Image inputs are converted to floating point and normalized with ImageNet mean $(0.485, 0.456, 0.406)$ and standard deviation $(0.229, 0.224, 0.225)$. The k-NN, latent-mean, and random-projector comparisons are constrained to the projector's training examples. 1,000 Push-T, Two-Room and OGB-Cube latents from D_g. In particular, the Two-Room and Cube baselines do not use all hundreds of thousands of available positive frames that these datasets have to offer.

Table 4: **LEWM Reproduction.** Success rates (%) following the original LEWM evaluation protocol. This includes a goal offset of 25, action budget of 50 etc.

Dataset	Seed 1	Seed 2	Seed 3	LEWM (ours)	LEWM (Maes et al., 2026)
Push-T	93.0	93.0	94.0	93.3	96.0
Two-Room	84.0	85.0	85.0	84.7	87.0
OGB-Cube	69.0	70.0	73.0	70.7	74.0
Overall success rate				82.9	85.7

D WHAT CHANGES IN REPRESENTATION SPACE?

In Figure 11, we compare the latent and goal space. The LEWM latent preserves many physical and visual variables for accurate next time step prediction, so latent goals vectors (red) may be spread across the representation space. Now, the projector contracts goals while ensuring broad variation among the dataset using SIGReg and dynamics losses. This is visible most strongly on Push-T and OGB-Cube. The same behaviour can be observed on Two-Room.

The coloring of ordinary frames encodes their latent 1-NN distance and is used for both spaces. The projection reorganizes states that were close to different individual goal examples around a common goal cluster and maintains the general structure.

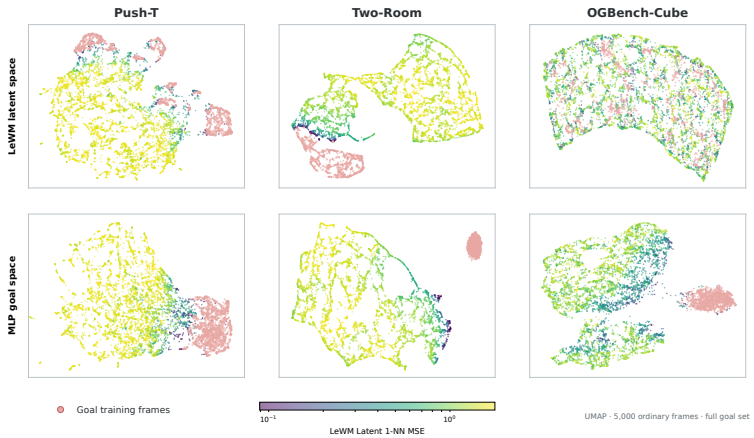

Figure 11: **Visualization of the Latent and Goal Space.** UMAP projections (McInnes et al., 2018) of the latent and learned space. Red points show goal images. In addition, we color 5,000 ordinary training frames by their shortest latent distance to the goal set (the 1-NN cost). The coloring is carried over from the latent space (top) into the goal space (bottom) where the overall structure is maintained.

E LIMITATIONS

Data and Supervision. Learning goal invariances requires successful examples that capture the variation permitted by the task. We identify these examples using simulator information, although planning uses images only. The trajectory-based losses also assume that recorded behavior progresses toward the goal and is preferable to sampled or altered action sequences. However, these assumptions need not to hold for every training sequence.

Model and Evaluation Scope. The projector cannot recover information discarded by the frozen encoder or correct inaccurate dynamics predictions. Moreover, compactness and regularization do not guarantee that every low-cost state is successful. Our experiments cover three simulated tasks and three world-model seeds, with a separate projector trained for each goal set. They demonstrate generalization to new evaluation states within these tasks, while transfer to unseen tasks, broader goal families, and real-world settings remains to be evaluated.