REPRESENTATION GEOMETRY MATTERS FOR PLANNING WITH JEPA WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Joint-embedding predictive world models support planning through latent predictions, but unconstrained joint training can collapse distinct observations to identical embeddings. Two prominent strategies for avoiding collapse are to inherit pretrained features, as in DINO-WM, or to learn representations end-to-end with anti-collapse regularization, as in LeWorldModel (LeWM). Yet avoiding collapse does not ensure that latent distances distinguish outcomes in ways that matter for the task. In object manipulation, for example, success depends on the object's position and orientation relative to the goal. Such task-relevant state information can remain accurately decodable while barely influencing latent distance. The resulting planning cost may fail to reflect how close a predicted outcome is to the task goal. In this paper, we propose SCALE (State-CAlibrated Latent Embeddings), a method that correlates sampled pairwise latent distances with distances in task-relevant state space. Added to LeWM's existing objective, SCALE preserves its architecture, requires privileged state only during training, and adds no planning-time computation. We show that SCALE improves planning success over LeWM across manipulation and navigation tasks with multiple solvers and provide a comprehensive analysis of how SCALE reshapes representation geometry to support planning.

1 INTRODUCTION

Learning representations that support both prediction and planning is a central challenge in visual control. Joint-embedding predictive architectures (JEPAs) learn by predicting the representation of a target from an observed context, without reconstructing that target in pixel space (Assran et al., 2023). Action-conditioned JEPA world models apply this principle to temporal prediction: an encoder maps observations to a latent space, and a predictor forecasts how those embeddings evolve under candidate actions (Zhou et al., 2024; Sobal et al., 2025; Maes et al., 2026). In goal-conditioned latent-space planning, solvers such as CEM or iCEM search over action sequences by minimizing the distance between predicted terminal embeddings and a goal embedding. For this objective to be meaningful, latent distances should reflect task-relevant state differences. For example, in pose-sensitive manipulation tasks, observations with similar object poses should have nearby embeddings, whereas larger task-relevant pose differences should correspond to greater latent separation. Candidate scores therefore depend on how the representation expresses differences between outcomes. However, accurately decodable task-state information does not necessarily imply such representation geometry: task-relevant differences may contribute little to Euclidean distance and, consequently, to the costs used by the planner.

Existing latent world models obtain their representations in different ways. DINO-WM builds on pretrained visual features, whereas LeWorldModel (LeWM) learns its encoder end-to-end with SIGReg anti-collapse regularization (Zhou et al., 2024; Maes et al., 2026; Balestriero & LeCun, 2025). Our held-out diagnostics in Sec. 4 reveal that DINO-WM exhibits stronger alignment between latent and task-state distances on four of five tasks, even though task-relevant state remains readily decodable from LeWM's representations. This contrast highlights a distinction between preserving state information and making that information influential in the metric used for planning. To characterize this distinction, we introduce *metric leverage*, defined as a latent direction's expected contribution to squared pairwise Euclidean distance. State information can therefore be accurately recovered by a decoder while residing in directions that contribute little to the planning cost. We formalize this sep-

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

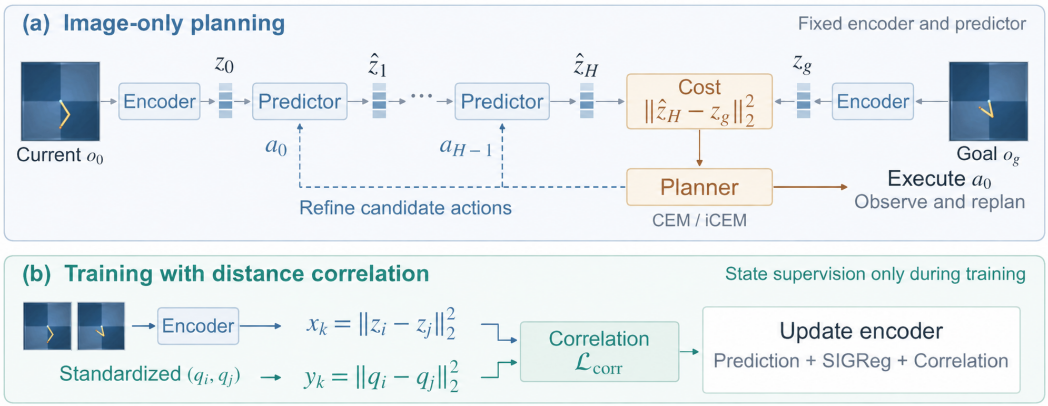

Figure 1: (a) A fixed encoder and predictor support action search using the terminal–goal latent distance. The planner executes the first action block and replans. (b) During training, SCALE correlates sampled latent and task-state distance using the correlation-based regularizer.

aration through a constructive counterexample in Sec. 4, showing that even exact state decodability does not guarantee distance alignment.

Motivated by this contrast, we seek to transfer the favorable geometric property observed in DINO-WM without inheriting its computational cost. In our implementation, DINO-WM requires approximately $75\times$ the planning time of LeWM at the largest matched planning budget (Appendix C.3). We therefore propose SCALE (State-CAlibrated Latent Embeddings), a lightweight training-time method that augments LeWM's existing objective (Fig. 1). SCALE calibrates geometry by correlating sampled pairwise latent distances with distances in standardized task-relevant state space. Selected simulator variables supply supervision only during training; the architecture and image-only planner are unchanged, with no additional planning-time computation.

We evaluate SCALE on Push-T, Reacher, Cube, Two-Room, and PointMaze using several sampling-based solvers, varying the number of candidate action sequences and search iterations to control planning computation. Averaged over these search settings, SCALE improves planning success over LeWM on all five tasks with each solver. To distinguish geometry calibration from improved state recovery, we introduce an auxiliary state-regression control, Aux, using the same supervision. Aux makes state highly decodable but yields less consistent planning gains. Together, these diagnostics show that planning depends on how task-relevant information shapes latent distance, not merely on whether it is preserved. Our main contributions are as follows:

- We characterize *metric leverage* and the separation between state decodability and distance alignment through empirical diagnostics and a constructive counterexample (Sec. 4).
- We introduce SCALE, give a conditional candidate-selection bound separating distance misalignment from rollout error, and provide a comprehensive explanation of how SCALE improves planning by reshaping representation geometry (Secs. 5 and 6).
- We demonstrate that SCALE consistently improves over LeWM across five tasks and two planning solvers, averaged over multiple compute budgets, and use a matched state-regression baseline to investigate the benefits of distance alignment beyond improved state decodability (Sec. 6).

2 Related Work

Predictive representations and collapse prevention. JEPA learns representations by predicting target embeddings from an observed context; I-JEPA applies this principle to masked image regions (Assran et al., 2023). Action-conditioned world models extend the prediction task to future observations. Because the encoder and predictor can trivially agree by mapping every observation to a constant, the representation-learning objective must also prevent collapse. One approach fixes a representation learned beforehand: DINO-WM predicts DINOv2 patch features, inheriting their visual

structure while keeping the encoder frozen (Oquab et al., 2023; Zhou et al., 2024). Another learns the representation together with dynamics. PLDM combines latent prediction with VICReg-inspired regularization and inverse dynamics (Bardes et al., 2021; Sobal et al., 2025), whereas LeWM uses next-embedding prediction and SIGReg, which encourages an isotropic Gaussian embedding distribution (Balestriero & LeCun, 2025; Maes et al., 2026). LpWM explores non-negative sparse representations as an alternative to LeWM's dense Gaussian-matched embeddings (Kuang et al., 2026). These designs determine how information is encoded and how collapse is avoided. SCALE retains LeWM's learning framework and adds supervision for a separate property: whether latent distances reflect differences in the selected task state.

Supervision for planning. Several approaches add supervision to make JEPA representations more useful for planning. Value-guided action planning aligns latent distances with goal-conditioned value (Destrade et al., 2025). SCALE instead uses pairwise distances in logged task state as its supervision signal, without requiring reward labels, a learned value function, or a policy. State-grounding methods provide another closely related comparison. PhyLatent diagnoses physically separated states that remain close in LeWM's latent space and incorporates physical-state supervision (Zeng et al., 2026); PSG-JEPA includes auxiliary objectives for physical state and state changes (Yan et al., 2026). Their state-prediction heads make physical information accessible through a learned readout. Accurate recovery through such a head, however, does not by itself require that information to influence the Euclidean distance used by the planner. Our Aux control isolates this supervision mechanism. Appendix A reviews planning-aware world models, behavioral metrics, and plannable representations.

3 PRELIMINARIES

3.1 LATENT-SPACE PLANNING

Let o_t be an image frame and $\mathbf{a}_t$ an action. An episode is one recorded environment trajectory; sub-trajectories are its contiguous segments. LeWM uses a ViT-Tiny encoder E_θ and an action-conditioned causal Transformer predictor P_ψ (Appendix C). They produce $\mathbf{z}_t = E_\theta(o_t) \in \mathbb{R}^{d_z}$ and $\hat{\mathbf{z}}_{t+1} = P_\psi(\mathbf{z}_t, \mathbf{a}_t)$. At test time, H is the planning horizon in model steps. Given candidate actions $\mathbf{a}_{0:H-1}$ and current frame o_0, the model sets $\hat{\mathbf{z}}_0 = E_\theta(o_0)$ and rolls out

$$\hat{\mathbf{z}}_{t+1} = P_\psi(\hat{\mathbf{z}}_t, \mathbf{a}_t), \qquad t = 0, \ldots, H-1. \tag{1}$$

We suppress the predictor's finite history in this notation; t indexes model steps and $\mathbf{a}_t$ denotes an action block. For a fixed initial observation and history, let $\hat{z}_H(a_{0:H-1})$ denote the terminal embedding predicted by rolling out the model under the candidate action sequence $a_{0:H-1}$. For a goal observation o_g with embedding $\mathbf{z}_g = E_\theta(o_g)$, the planner ranks candidates by a cost function

$$J(\mathbf{a}_{0:H-1}) = C\big(\hat{\mathbf{z}}_H(\mathbf{a}_{0:H-1}),\, \mathbf{z}_g\big), \tag{2}$$

where C scores a terminal–goal embedding pair. The planner is the optimizer that searches over action sequences using J. Following LeWM and DINO-WM, we adopt the squared Euclidean distance as the cost function:

$$J(\mathbf{a}_{0:H-1}) = \big\|\hat{\mathbf{z}}_H(\mathbf{a}_{0:H-1}) - \mathbf{z}_g\big\|_2^2. \tag{3}$$

3.2 LEWM TRAINING OBJECTIVE

For a transition $(o_t, \mathbf{a}_t, o_{t+1})$, the encoder and predictor are trained with the one-step latent-prediction loss

$$\mathcal{L}_{\text{pred}} = \big\|\hat{\mathbf{z}}_{t+1} - \mathbf{z}_{t+1}\big\|_2^2, \tag{4}$$

where $\mathbf{z}_{t+1} = E_\theta(o_{t+1})$ is the target embedding. This loss alone admits collapsed solutions in which the encoder maps every observation to the same point. For a batch of N encoder outputs $Z = \{\mathbf{z}_i\}_{i=1}^N$, SIGReg samples random unit directions in latent space, projects Z onto each, and penalizes deviations of the projected empirical distributions from a univariate standard normal (Balestriero & LeCun, 2025; Maes et al., 2026). Applied across many directions, this drives the embedding distribution toward an isotropic Gaussian and maintains global spread. LeWM therefore optimizes

$$\mathcal{L}_{\text{LeWM}} = \mathcal{L}_{\text{pred}} + \lambda_{\text{sig}}\, \text{SIGReg}(Z). \tag{5}$$

4 WHERE TASK-RELEVANT STATE LIVES

Under the squared Euclidean objective in Eq. (3), the solver favors candidate action sequences whose predicted terminal embeddings lie closer to the goal embedding. For this comparison to guide task completion, latent distances should reflect differences in task-relevant state. In object manipulation, for example, differences in object position and orientation relative to the target should affect the planning cost. A cost that is insensitive to these differences may provide little guidance toward the task goal. To examine whether the representations capture this relationship, we compute the Spearman rank correlation ρ_S between $\|\mathbf{z}_i - \mathbf{z}_j\|_2^2$ and $\|\mathbf{q}_i - \mathbf{q}_j\|_2^2$ over held-out pairs of image frames, where $\mathbf{q}$ denotes the task-relevant state variables logged alongside each trajectory (formally specified in Sec. 5). We rank the two distance lists separately, assigning average ranks to ties. A high positive correlation indicates that pairs farther apart in task state also tend to be farther apart in latent space.

Table 1: Spearman correlation ρ_{S} between latent and task-state distances for LeWM and DINO-WM on held-out frame pairs.

Task	LeWM	DINO-WM
Push-T	.13	.70
Reacher	.18	.46
Cube	.00	.46
Two-Room	.41	.85
PointMaze	.52	.50

Table 1 shows higher distance alignment for DINO-WM on four tasks and similar alignment on PointMaze. Yet probes (i.e., regression models trained to predict state variables from frozen embeddings) recover most task-relevant state accurately from both full embeddings (see Appendix D for more details). Why can state be recoverable while distances reflect it poorly? Proposition 4.1 makes this separation precise: even perfect linear recovery need not imply aligned distances. To see why a decoder can recover but a fixed Euclidean cost can underweight, we examine each direction's contribution to that cost.

Let $z \in \mathbb{R}^{d_z}$ be a square-integrable encoder output, mean $\mu = \mathbb{E}[z]$, and covariance $\Sigma_z = \mathbb{E}[(z - \mu)(z - \mu)^\top] = U\Lambda U^\top$. Here $U^\top U = I_{d_z}$, $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_{d_z})$, and $\lambda_1 \geq \cdots \geq \lambda_{d_z} \geq 0$. Writing $U = (\mathbf{u}_1, \ldots, \mathbf{u}_{d_z})$ and $c = U^\top(z - \mu)$ gives

$$z - \mu = \sum_{j=1}^{d_z} c_j \mathbf{u}_j, \qquad \mathrm{Var}(c_j) = \lambda_j. \tag{6}$$

For any two embeddings $\mathbf{z}_i$ and $\mathbf{z}_g$, orthogonality yields the exact decomposition

$$\left\|\mathbf{z}_i - \mathbf{z}_g\right\|_2^2 = \sum_{j=1}^{d_z} \left(c_{ij} - c_{gj}\right)^2. \tag{7}$$

We define the *metric leverage* of direction j as its expected contribution to squared distance between independent copies z, z' of the encoder output, namely $\mathbb{E}[(c_j - c'_j)^2]$. The following proposition relates this quantity to variance and shows that accurate state recovery alone does not guarantee distance alignment.

Proposition 4.1 (Metric leverage and state decodability)**.** *For independent copies z, z', with principal coordinates c, c', the metric leverage of coordinate j is*

$$\mathbb{E}[(c_j - c'_j)^2] = 2\lambda_j. \tag{8}$$

Nevertheless, for any integers $d_z > d_q \geq 1$ and any $\delta > 0$, there exists a centered jointly Gaussian pair (q, z) with $\Sigma_q := \mathbb{E}[qq^\top] = I_{d_q}$, $\Sigma_z \succ 0$, and $\mathbb{E}\|z - z'\|_2^2 \geq 1$, such that $h(z) = q$ almost surely for some linear decoder h, yet

$$0 \leq \rho_{\mathrm{P}}(\|z - z'\|_2^2, \|q - q'\|_2^2) < \delta.$$

Here (q', z') is an independent copy of (q, z) and ρ_{P} denotes Pearson correlation, distinct from the Spearman diagnostic in Table 1.

Remark 4.2. The Gaussianity assumption in Proposition 4.1 is used for expositional convenience; the separation it illustrates is not specific to the Gaussian distribution itself. Proposition 4.1 shows that a decoder can rescale the coordinates encoding state to recover it exactly, even when those coordinates contribute arbitrarily little to the squared latent distance on average. Thus,

$$\mathbf{q} \text{ is decodable from } \mathbf{z} \quad \not\Rightarrow \quad \mathbf{q} \text{ strongly affects } C(\mathbf{z}, \mathbf{z}_g) \text{ on average.}$$

The proof of Proposition 4.1 is given in Appendix B.

Together, Table 1 and Proposition 4.1 motivate explicitly relating latent distances to task-state differences during training, beyond making state decodable. We refer to this as *distance calibration*. SCALE implements this through the correlation regularizer $\mathcal{L}_{\text{corr}}$ (see more details in Sec. 5), which encourages sampled pairwise latent distances to correlate with the corresponding task-state distances while retaining LeWM's prediction objective and planning procedure.

5 METHOD

In this section, we formally present our methodology. Consider a minibatch of B sub-trajectories, each containing T frames. We flatten its $N = BT$ frames into $\{(\mathbf{z}_n, \mathbf{q}_n, b_n, e_n)\}_{n=1}^{N}$, where $\mathbf{z}_n \in \mathbb{R}^{d_z}$ is an encoder output, $\mathbf{q}_n \in \mathbb{R}^{d_q}$ is its standardized task-relevant state, b_n identifies its sub-trajectory, and e_n its source episode. At each step, we sample an indexed list $\mathcal{P} = ((i_k, j_k))_{k=1}^{K}$ of non-self frame pairs ($i_k \neq j_k$), allowing repeated draws, with half drawn within sub-trajectories ($b_{i_k} = b_{j_k}$) and half across episodes ($e_{i_k} \neq e_{j_k}$). This stratification supplies both local and global geometric comparisons. We then define the latent and state distance profiles

$$x_k = \|\mathbf{z}_{i_k} - \mathbf{z}_{j_k}\|_2^2, \qquad y_k = \|\mathbf{q}_{i_k} - \mathbf{q}_{j_k}\|_2^2. \tag{9}$$

Let $\bar{x}$ and s_x be the mean and standard deviation of $\{x_k\}_{k=1}^{K}$; $\bar{y}$ and s_y are defined analogously. With a numerical constant $\varepsilon > 0$, the standardized profiles and correlation loss are

$$\tilde{x}_k = \frac{x_k - \bar{x}}{s_x + \varepsilon}, \qquad \tilde{y}_k = \frac{y_k - \bar{y}}{s_y + \varepsilon}, \qquad \mathcal{L}_{\text{corr}} = 1 - \frac{1}{K}\sum_{k=1}^{K} \tilde{x}_k \tilde{y}_k. \tag{10}$$

Component-wise standardization of $\mathbf{q}$ places the selected state variables on comparable scales, while standardizing the two distance profiles focuses $\mathcal{L}_{\text{corr}}$ on relative separation rather than absolute scale. Minimizing $\mathcal{L}_{\text{corr}}$ promotes positive correlation between the squared latent distances x_k and the corresponding squared task-state distances y_k, encouraging pairs with larger task-state differences to lie farther apart in latent space. Appendix D.1 details the task-specific choices of $\mathbf{q}$ and their full-state comparisons.

Therefore, the complete objective is defined as:

$$\mathcal{L}_{\text{SCALE}} \;=\; \mathcal{L}_{\text{pred}} \;+\; \lambda_{\text{sig}}\,\text{SIGReg}(Z) \;+\; \lambda_{\text{corr}}\,\mathcal{L}_{\text{corr}}. \tag{11}$$

The prediction term learns action-conditioned latent dynamics, SIGReg maintains global spread, and $\mathcal{L}_{\text{corr}}$ directly supervises the relation between latent and task-state distances. The complete $\mathcal{L}_{\text{SCALE}}$ updates the encoder E_θ, whereas the predictor P_ψ is updated only by $\mathcal{L}_{\text{pred}}$. Training pseudocode is provided in Appendix C.1.

From distance calibration to planning. By encouraging latent distances to reflect task-state differences, SCALE aims to make the planning cost more informative about which candidate outcomes are closer to the goal. The following result relates candidate selection to distance correlation and rollout prediction error.

Fix a start–goal pair and $M \geq 2$ candidate action sequences. For candidate m, let $z_H^{(m),\text{true}}$ and $q_H^{(m),\text{true}}$ denote its encoded simulator terminal observation and terminal task state, respectively, and let $\hat{z}_H^{(m)}$ be its predicted terminal embedding. For goal embedding z_g and task state q_g, write

$$\hat{J}_m = \|\hat{z}_H^{(m)} - z_g\|_2^2, \qquad J_m^{\text{true}} = \|z_H^{(m),\text{true}} - z_g\|_2^2.$$

Let s_z, s_q denote the empirical standard deviations of $\{J_m^{\text{true}}\}_{m=1}^{M}$ and $\{\|q_H^{(m),\text{true}} - q_g\|_2^2\}_{m=1}^{M}$, respectively, using divisor M and ρ_{P} be the Pearson correlation between these two lists.

Theorem 5.1 (Distance calibration and candidate selection). *Suppose $s_z, s_q > 0$, and set $C_z = \max_{1\le m\le M} \|z_H^{(m),\mathrm{true}} - z_g\|_2$ and $C_{\mathrm{roll}} = \max_{1\le m\le M} \|\hat{z}_H^{(m)} - z_H^{(m),\mathrm{true}}\|_2$. Then every $\hat{m} \in \arg\min_{1\le m\le M} \hat{J}_m$ satisfies*

$$\begin{aligned} 0 &\le \|q_H^{(\hat{m}),\mathrm{true}} - q_g\|_2^2 - \min_{1\le m\le M} \|q_H^{(m),\mathrm{true}} - q_g\|_2^2 \\ &\le 2s_q\sqrt{M(1-\rho_{\mathrm{P}})} + \frac{2s_q}{s_z}\left(2C_zC_{\mathrm{roll}} + C_{\mathrm{roll}}^2\right). \end{aligned} \tag{12}$$

In particular, if $\rho_{\mathrm{P}} = 1$ and $C_{\mathrm{roll}} = 0$, then $\hat{m}$ minimizes the terminal task-state distance among the candidates.

The first term reflects imperfect correlation between latent and task-state distances to the goal; the second accounts for rollout error. These terms vanish under perfect correlation and exact rollouts, respectively. The bound measures the selected candidate's excess squared task-state distance over the best candidate in the evaluated set. The proof is in Appendix B.

An auxiliary state-regression control (Aux). To investigate whether the benefits of state supervision arise simply from improved state decodability, we construct an auxiliary state-regression control, denoted Aux. It uses the same backbone, state targets, and training protocol as SCALE, replacing $\mathcal{L}_{\mathrm{corr}}$ with a regression loss through a training-only head h_ϕ:

$$\mathcal{L}_{\mathrm{aux}} = \left\|h_\phi(\mathbf{z}_n) - \mathbf{q}_n\right\|_2^2 \tag{13}$$

The complete objective is $\mathcal{L}_{\mathrm{pred}} + \lambda_{\mathrm{sig}}\,\mathrm{SIGReg}(Z) + \lambda_{\mathrm{aux}}\mathcal{L}_{\mathrm{aux}}$. Aux is intended as a mechanism control rather than a separately optimized planning method. Unlike distance calibration, the learned head can rescale weak latent coordinates to recover state without requiring them to carry substantial *metric leverage*. Aux therefore tests whether making state readily decodable is sufficient to improve planning geometry. Implementation details and hyperparameters are provided in Appendix C.

6 Experiments

6.1 Setup

We follow the LeWM goal-conditioned planning protocol and evaluate CEM (De Boer et al., 2005) and iCEM (Pinneri et al., 2021), with DINO-WM as a pretrained-feature reference. Table 2 reports planning success pooled over two independently trained runs, each evaluated on six held-out sets of 100 episodes. Table 3 evaluates rollout error and candidate-ranking fidelity. Planner budgets, horizons, evaluation construction, and complete tier-level results are detailed in Appendix C and Appendix E.

6.2 Planning performance

SCALE improves LeWM on every pooled task–solver average (10/10). Under iCEM (Table 2), its gains are 4.4, 2.6, 6.5, 10.2, and 2.5 percentage points on Push-T, Reacher, Cube, Two-Room, and PointMaze, respectively. The corresponding CEM gains are 2.4, 0.3, 3.2, 11.7, and 1.4 points. This consistency supports improvements across both search procedures and all five tasks.

The regression control helps distinguish distance calibration from state decodability. Full-embedding probes confirm that Aux successfully encodes the supervised state, even exceeding SCALE's decodability on Push-T and Cube (Appendix D). Aux improves seven of ten pooled task–solver averages, ties LeWM once, and is lower on both PointMaze averages. It trails SCALE on four of five tasks within each solver; Push-T is the exception. Precise pose recovery may favor Aux there: it has stronger position/orientation probes and slightly lower rollout error (Table 3).

DINO-WM remains strongest on Two-Room and PointMaze. Its pretrained spatial patch features (Zhou et al., 2024) may preserve navigation-relevant detail; it also has lower rollout error and higher candidate-ranking agreement (Table 3).

Table 2: Pooled iCEM success rates (%) across tasks and compute tiers. Each entry reports mean ± SD over 12 set-level measurements: two independently trained runs, each evaluated on six held-out sets of 100 episodes. Overall pools T1–T5. Bold marks the best method per column within each task. Appendix E reports both CEM and iCEM results.

Task	Method	T1	T2	T3	T4	T5	Overall
Push-T	LeWM	86.0±2.4	79.8±2.1	73.4±5.9	55.2±4.8	25.1±5.1	63.9±3.0
	SCALE	89.1±2.0	**86.3±2.8**	**79.7±3.6**	61.3±6.2	25.2±5.5	68.3±2.8
	Aux	**89.9±2.0**	86.2±2.3	79.6±3.4	**62.2±5.3**	**27.6±5.2**	**69.1±2.6**
	DINO-WM	70.0±4.5	65.3±4.5	58.8±5.1	41.5±8.8	23.1±3.7	51.7±4.3
Reacher	LeWM	84.9±2.6	84.2±3.4	83.8±3.4	79.0±6.1	62.2±3.6	78.8±1.7
	SCALE	**86.7±3.6**	**87.0±3.1**	**87.3±3.2**	**81.7±3.0**	**64.4±3.0**	**81.4±2.0**
	Aux	85.0±4.4	86.1±3.1	85.8±3.5	80.1±3.7	62.9±5.1	80.0±2.2
	DINO-WM	78.7±4.9	78.2±4.8	80.8±3.5	76.0±4.0	53.0±5.5	73.3±3.1
Cube	LeWM	68.9±4.9	67.1±3.6	68.7±4.1	61.9±3.9	54.9±5.7	64.3±4.0
	SCALE	**76.8±3.7**	**76.6±4.0**	**75.2±3.0**	**65.8±3.3**	**59.8±3.9**	**70.8±3.1**
	Aux	73.8±3.6	73.0±4.7	72.5±3.8	64.8±3.4	57.8±3.8	68.4±3.5
	DINO-WM	69.9±2.5	67.4±1.6	66.0±1.7	57.7±3.8	51.4±3.6	62.5±1.3
Two-Room	LeWM	91.1±1.6	92.9±1.7	88.5±2.0	77.3±2.2	62.4±2.8	82.5±1.2
	SCALE	99.9±0.3	99.8±0.6	97.7±1.2	90.2±3.0	76.2±2.7	92.7±1.0
	Aux	92.3±2.1	93.2±1.7	89.2±2.0	78.7±2.5	62.9±3.6	83.3±1.1
	DINO-WM	**100.0±0.0**	**100.0±0.0**	**99.8±0.5**	**98.6±0.8**	**92.5±2.1**	**98.2±0.5**
PointMaze	LeWM	84.0±4.1	83.6±4.3	82.4±3.1	81.2±4.6	73.2±4.8	80.9±3.5
	SCALE	87.6±3.3	85.8±4.5	84.6±3.2	81.9±4.7	77.0±4.4	83.4±2.5
	Aux	80.8±5.1	79.5±5.1	78.7±5.0	79.4±3.7	75.0±5.8	78.7±4.3
	DINO-WM	**93.4±2.6**	**92.2±2.6**	**90.8±3.4**	**90.3±2.6**	**85.2±2.6**	**90.4±2.1**

6.3 How SCALE reorganizes the latent geometry

To examine how SCALE changes the representation, we study its variance spectrum, the state information in its leading directions, and the fidelity of predicted candidate rankings.

SCALE concentrates variance in the leading latent directions. Figure 2 shows that SCALE allocates more normalized variance to the leading principal directions than LeWM and Aux on most tasks. DINO-WM also has a strongly weighted spectral head, but its larger representation produces a longer tail. By Sec. 4, direction j contributes $2\lambda_j$ in expectation to squared distance between independent embeddings. Hence $\lambda_j / \sum_\ell \lambda_\ell$ measures its share of that distance, not absolute latent scale. The spectrum alone does not identify which state variables acquire this leverage.

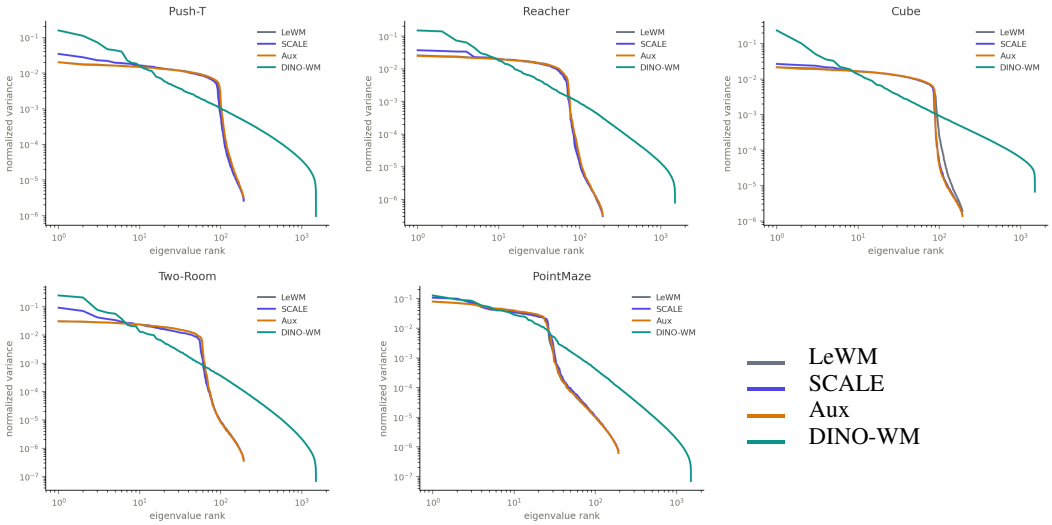

Figure 2: Normalized PCA eigenvalue spectra of LeWM, SCALE, Aux, and DINO-WM across five tasks. Eigenvalues are normalized by total variance, so each principal component's share corresponds to its relative metric leverage. SCALE concentrates more variance in the leading principal components than LeWM and Aux on most tasks.

The leading directions encode task-relevant variation. We next ask what the high-variance directions actually encode. For each frozen representation, we regress the selected task-relevant state $\mathbf{q}$ using only its leading k principal components. Figure 3 shows that the leading components of SCALE are substantially more predictive of $\mathbf{q}$ than those of LeWM. DINO-WM also concentrates task-state information in leading components: it is comparable to SCALE on Push-T and stronger on Two-Room, but trails SCALE on Reacher, Cube, and PointMaze. The separation from Aux is especially informative: although Aux can recover $\mathbf{q}$ accurately from the full embedding, less of that information appears in its leading components on four of the five tasks.

Together, these diagnostics associate the greater concentration of variance in leading directions with more accessible task-state information. Conversely, auxiliary regression can make the same state highly decodable without organizing it in the part of the representation that contributes most strongly to Euclidean distance. Consistent with this interpretation, SCALE also substantially increases held-out latent–state distance alignment, whereas Aux leaves it near the LeWM level despite strong state decodability (Appendix D.2).

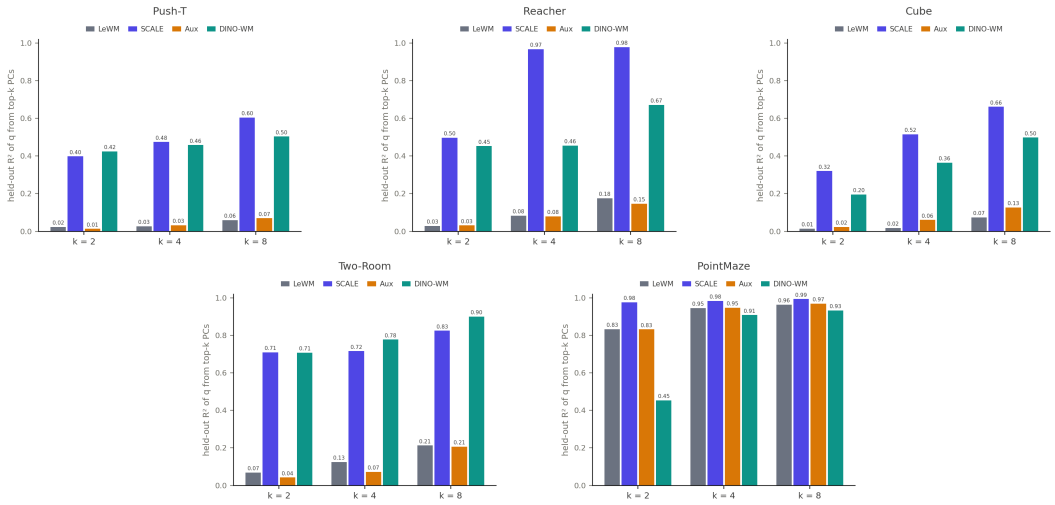

Figure 3: Task-state information in leading principal components. Held-out R^2 for predicting task-relevant state from the leading k principal components across five tasks. SCALE improves state predictability compared with LeWM and outperforms Aux on four of five tasks. This suggests that SCALE concentrates task-relevant information in directions with greater metric leverage.

Rollouts better preserve the encoded candidate ranking. The preceding analysis concerns encoded observations, whereas planning operates on predicted future embeddings. For each start and candidate action sequence m, the world model predicts a terminal embedding $\hat{\mathbf{z}}_H^{(m)}$. Executing the same sequence from the same start in the simulator produces $o_H^{(m),\mathrm{true}}$, which the same encoder maps to $\mathbf{z}_H^{(m),\mathrm{true}} = E_\theta(o_H^{(m),\mathrm{true}})$.

These paired outcomes define two measurements. First, rollout error compares the predicted and simulator-derived embeddings as $\|\hat{\mathbf{z}}_H^{(m)} - \mathbf{z}_H^{(m),\mathrm{true}}\|_2^2/s$, where s is the model's mean squared distance between random pairs of the simulator-derived terminal embeddings $\mathbf{z}_H^{(m),\mathrm{true}}$. Second, for the same goal embedding $\mathbf{z}_g$, we compute

$$\hat{J}_m = \|\hat{\mathbf{z}}_H^{(m)} - \mathbf{z}_g\|_2^2, \qquad J_m^{\mathrm{true}} = \|\mathbf{z}_H^{(m),\mathrm{true}} - \mathbf{z}_g\|_2^2,$$

and report Kendall's τ between the candidate rankings induced by $\{\hat{J}_m\}$ and $\{J_m^{\mathrm{true}}\}$. We use Kendall's τ to assess pairwise action-ranking reversals. Spearman's ρ_S in Table 1 instead measures monotonic latent–state distance association over observation pairs; the diagnostics answer different questions.

Table 3: Within-model rollout quality. Lower rollout error and higher Kendall τ are better. Parentheses report change relative to LeWM; when LeWM's τ is near zero, we report the absolute difference Δ. Results average 20 starts.

Task	Metric	LeWM	SCALE	Aux	DINO-WM
Push-T	RollErr ↓	.0350	.0277 (-21.0%)	**.0269** (-23.1%)	.2819 ($+704.9\%$)
	Kendall τ ↑	.7316	**.7857** ($+7.4\%$)	.7576 ($+3.6\%$)	.7000 (-4.3%)
Reacher	RollErr ↓	.1812	**.1548** (-14.6%)	.1686 (-7.0%)	.5860 ($+223.4\%$)
	Kendall τ ↑	.6233	**.6435** ($+3.2\%$)	.6328 ($+1.5\%$)	.6414 ($+2.9\%$)
Cube	RollErr ↓	.3195	.3174 (-0.7%)	**.2389** (-25.2%)	.3227 ($+1.0\%$)
	Kendall τ ↑	.3654	.4433 ($+21.3\%$)	.4150 ($+13.6\%$)	**.7775** ($+112.8\%$)
Two-Room	RollErr ↓	.3600	.2857 (-20.6%)	.3511 (-2.5%)	**.1479** (-58.9%)
	Kendall τ ↑	.5448	.6210 ($+14.0\%$)	.5389 (-1.1%)	**.7250** ($+33.1\%$)
PointMaze	RollErr ↓	2.4226	2.1132 (-12.8%)	2.2418 (-7.5%)	**1.9545** (-19.3%)
	Kendall τ ↑	$-.0029$	.1675 (n/a, $\Delta+.170$)	.0540 (n/a, $\Delta+.057$)	**.2850 (n/a,** $\Delta+.288$**)**

SCALE increases Kendall ranking agreement over LeWM on every task. The improvement is modest on Push-T and Reacher, where LeWM already preserves candidate ordering relatively well, but becomes much larger on Cube, Two-Room, and PointMaze. The higher agreement indicates that rollout predictions better preserve the candidate ordering induced by each model's encoded terminal states. This diagnostic measures fidelity to the model's own latent cost, rather than agreement with task-state distances.

Aux has the lowest rollout error on Push-T and Cube, yet SCALE improves Kendall agreement more on every task. DINO-WM combines favorable rollout and ranking behavior on the navigation tasks, but has much larger rollout errors on Push-T and Reacher; on Cube, it has the highest Kendall τ without the strongest planning performance. These comparisons distinguish fidelity to a model's latent cost from whether that cost reflects task-state differences, the separate requirements assessed by our rollout and distance-alignment diagnostics.

7 Conclusion

Task-relevant state can be decodable from a representation yet weakly influence latent-space planning. Motivated by the contrasting geometries of DINO-WM and LeWM, we introduce SCALE to directly calibrate latent distances with task-relevant state differences during training. Using privileged state only during training, SCALE retains LeWM's end-to-end learning framework and image-only planner while making task-relevant variation more prominent in the latent metric. Planning thus depends on both recoverable information and its influence on the latent cost. Our current $\mathbf{q}$ selection follows a simple human prior: use a compact set of static variables while removing velocities and highly redundant coordinates. More broadly, understanding and designing representation geometry for JEPA world models remains an important direction for future research.

AI Use Statement

Generative AI tools were used to assist with editing, LaTeX formatting, table construction, and the organization, summarization, and presentation of experimental results.

Ethics Statement

This work studies representation learning and planning in simulated benchmark environments. It does not involve human participants, personally identifiable information, or sensitive real-world data. The reported results should therefore not be interpreted as evidence of readiness for safety-critical deployment.

Reproducibility Statement

The mathematical formulation of SCALE and its training objective are given in Sec. 5, with pseudocode in Appendix C.1. Appendix C specifies the implementation details. We will publicly release our code to support reproduction and further research.

References

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 *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629. IEEE, 2023.

Jiaxin Bai and Jiaxuan Xiong. Temporal-distance jepa: Plan-aware representation learning for latent world model predictive control. *arXiv preprint arXiv:2607.25337*, 2026.

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

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

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.

Amirhosein Chahe, Siwei Cai, and Lifeng Zhou. Latent world models with monotone planning costs for image-goal navigation. *arXiv preprint arXiv:2608.09073*, 2026.

Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

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*, 2025.

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. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 25475–25493. PMLR, 2024.

Yilun Kuang, Yash Dagade, Quentin Le Lidec, Lucas Maes, Randall Balestriero, and Yann LeCun. Lpwm: A case for sparse representations in world models. *arXiv preprint arXiv:2608.22764*, 2026.

Liangyu Li, Shengzhi Wang, and Qingwen Liu. Beyond euclidean proximity: Repairing latent world models with horizon-matched trajectory reachability metrics. *arXiv preprint arXiv:2605.22164*, 2026a.

Wenyuan Li, Guang Li, Keisuke Maeda, Takahiro Ogawa, and Miki Haseyama. Predictive but not plannable: Rc-aux for latent world models. *arXiv preprint arXiv:2605.07278*, 2026b.

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.

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.

Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. In *Conference on Robot Learning*, pp. 1049–1065. PMLR, 2021.

Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models, 2025. URL https://arxiv.org/abs/2502.14819.

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.

Haiyu Wu, Randall Balestriero, and Morgan Levine. Viscore: Diagnosing planning-relevant quality in latent world models. *arXiv preprint arXiv:2608.11174*, 2026.

Zaishuo Xia, Yukuan Lu, Xinyi Li, Yifan Xu, and Yubei Chen. Cloning deterministic worlds: The critical role of latent geometry in long-horizon world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 32602–32612, 2026.

Haodong Yan, Jiaguan Zhu, Mingyuan Jia, Ruiqing Yin, Junjie He, Zhide Zhong, Junfeng Li, Jinxuan Lu, Hengtao Li, Tianran Zhang, et al. Is forward prediction enough? physical state grounding for jepa world models. *arXiv preprint arXiv:2608.06799*, 2026.

Ge Yang, Amy Zhang, Ari Morcos, Joelle Pineau, Pieter Abbeel, and Roberto Calandra. Plan2vec: Unsupervised representation learning by latent plans. In *Learning for Dynamics and Control*, pp. 935–946. PMLR, 2020.

Xi Zeng, Haojie Ren, and Ziying Song. Phylatent: Learning dynamics-relevant representations for jepa world models. *arXiv preprint arXiv:2608.05720*, 2026.

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.

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

Planning-aware latent world models. Several recent methods directly address the gap between predictive training and planning-time geometry. Temporal Straightening penalizes local curvature so that Euclidean distance better approximates trajectory geodesics and improves the conditioning of latent planning (Wang et al., 2026). RC-aux combines multi-horizon prediction with budget-conditioned reachability supervision and temporal hard negatives, and can expose a reachability-aware planning signal (Li et al., 2026b). Temporal-Distance JEPA mines directed temporal progress from offline trajectories and uses it both as a learned planning cost and as representation supervision (Bai & Xiong, 2026). Trajectory Reachability Metrics instead repair a fixed world model post hoc by learning a horizon-matched pairwise terminal-ranking head from logged trajectory structure (Li et al., 2026a). Monotone Planning Costs train candidate costs to respect an ordering induced by increasingly perturbed action sequences in image-goal navigation (Chahe et al., 2026). VIScore takes a complementary diagnostic view, measuring planning-relevant properties across the encoder, predictor, and planner (Wu et al., 2026). Complementary work on geometrically regularized world models uses temporal contrastive constraints to improve long-horizon predictive fidelity in deterministic environments, further demonstrating that dynamics accuracy depends on latent geometry (Xia et al., 2026). SCALE differs in retaining the original Euclidean image-goal planning interface: it does not deploy a learned cost or reachability head, but calibrates the representation during training so that the existing terminal–goal distance better reflects selected task-state differences.

Behavioral metrics and plannable representations. Earlier methods explicitly shape representations using decision-relevant structure. DBC matches latent distances to bisimulation distances, while MICo develops a sample-based behavioral metric (Zhang et al., 2020; Castro et al., 2021). Plan2vec distills long-range distances from planned paths, and PcLast organizes latent states by reachability (Yang et al., 2020; Koul et al., 2024). These methods establish a broader connection between representation geometry and decision making; SCALE applies this principle to the Euclidean image-goal cost used by JEPA world models.

B PROOF

Proof of Proposition 4.1. Since $\mathbb{E}[c] = 0$ and $\mathbb{E}[cc^\top] = U^\top \Sigma_z U = \Lambda$, independence gives

$$\mathbb{E}[(c_j - c'_j)^2] = \mathbb{E}[c_j^2] + \mathbb{E}[(c'_j)^2] - 2\mathbb{E}[c_j]\mathbb{E}[c'_j] = 2\lambda_j.$$

For the counterexample, choose $0 < \alpha < 1$ and take

$$z = c \sim N(0, \Sigma_z), \qquad \Sigma_z = \begin{pmatrix} I_{d_z - d_q} & 0 \\ 0 & \alpha^2 I_{d_q} \end{pmatrix}.$$

Here $U = I_{d_z}$. Define $q = h(z) = \alpha^{-1}(z_{d_z - d_q + 1}, \ldots, z_{d_z})^\top$. Then h is linear, (q, z) is jointly Gaussian, $\Sigma_q = I_{d_q}$, and $\Sigma_z \succ 0$. Moreover, $\mathbb{E}\|z - z'\|_2^2 = 2(d_z - d_q + \alpha^2 d_q) \geq 2$.

The squared latent distance decomposes as

$$\|z - z'\|_2^2 = \sum_{j=1}^{d_z - d_q} (c_j - c'_j)^2 + \alpha^2 \|q - q'\|_2^2.$$

Write the first term as A and set $B = \|q - q'\|_2^2$. The two Gaussian coordinate blocks are independent, so A and B are independent, with $A/2 \sim \chi^2_{d_z - d_q}$ and $B/2 \sim \chi^2_{d_q}$. Hence $\mathrm{Var}(A) = 8(d_z - d_q)$ and $\mathrm{Var}(B) = 8d_q$. In particular,

$$\begin{aligned} \mathrm{Cov}(A + \alpha^2 B, B) &= 8\alpha^2 d_q, \\ \mathrm{Var}(A + \alpha^2 B) &= 8(d_z - d_q) + 8\alpha^4 d_q. \end{aligned}$$

Consequently,

$$\begin{aligned} \rho_{\mathrm{P}}(\|z - z'\|_2^2, \|q - q'\|_2^2) &= \frac{8\alpha^2 d_q}{\sqrt{8(d_z - d_q + \alpha^4 d_q) \cdot 8 d_q}} \\ &= \left(1 + \frac{d_z - d_q}{\alpha^4 d_q}\right)^{-1/2} \longrightarrow 0 \qquad \text{as } \alpha \downarrow 0. \end{aligned}$$

Choosing $\alpha > 0$ sufficiently small makes the correlation smaller than δ while preserving exact linear recovery $h(z) = q$ and the stated nondegeneracy conditions. In this construction, each task-state coordinate has metric leverage $2\alpha^2$, despite exact state recovery. □

Proof of Theorem 5.1. Choose any $m^\star \in \arg\min_{1 \leq m \leq M} \|q_H^{(m),\mathrm{true}} - q_g\|_2^2$. Expanding the squared norms gives, for every m,

$$\begin{aligned} |\hat{J}_m - J_m^{\mathrm{true}}| &\leq 2\|z_H^{(m),\mathrm{true}} - z_g\|_2 \|\hat{z}_H^{(m)} - z_H^{(m),\mathrm{true}}\|_2 \\ &\quad + \|\hat{z}_H^{(m)} - z_H^{(m),\mathrm{true}}\|_2^2 \\ &\leq 2C_z C_{\mathrm{roll}} + C_{\mathrm{roll}}^2. \end{aligned}$$

Since $\hat{J}_{\hat{m}} \leq \hat{J}_{m^\star}$, it follows that

$$\begin{aligned} J_{\hat{m}}^{\mathrm{true}} - J_{m^\star}^{\mathrm{true}} &\leq |J_{\hat{m}}^{\mathrm{true}} - \hat{J}_{\hat{m}}| + |\hat{J}_{m^\star} - J_{m^\star}^{\mathrm{true}}| \\ &\leq 2\left(2C_z C_{\mathrm{roll}} + C_{\mathrm{roll}}^2\right). \end{aligned}$$

Next, write $Q_m = \|q_H^{(m),\mathrm{true}} - q_g\|_2^2$ and define the standardized vectors $u, v \in \mathbb{R}^M$ by

$$u_m = \frac{J_m^{\mathrm{true}} - \bar{J}}{s_z}, \qquad v_m = \frac{Q_m - \bar{Q}}{s_q},$$

where $\bar{J} = M^{-1} \sum_m J_m^{\mathrm{true}}$ and $\bar{Q} = M^{-1} \sum_m Q_m$. Because s_z, s_q use divisor M, $\|u\|_2^2 = \|v\|_2^2 = M$ and $u^\top v = M\rho_{\mathrm{P}}$. Thus $w = v - u$ satisfies $\|w\|_2^2 = 2M(1 - \rho_{\mathrm{P}})$. For standard basis vectors $e_1, \ldots, e_M$, Cauchy–Schwarz gives

$$|w_{\hat{m}} - w_{m^\star}| = |(e_{\hat{m}} - e_{m^\star})^\top w| \leq \sqrt{2}\,\|w\|_2 = 2\sqrt{M(1 - \rho_{\mathrm{P}})}.$$

This inequality also holds when $\hat{m} = m^\star$. Centering cancels in the identity $(Q_{\hat{m}} - Q_{m^\star})/s_q = (J^{\text{true}}_{\hat{m}} - J^{\text{true}}_{m^\star})/s_z + w_{\hat{m}} - w_{m^\star}$, so

$$\frac{\|q_H^{(\hat{m}),\text{true}} - q_g\|_2^2 - \|q_H^{(m^\star),\text{true}} - q_g\|_2^2}{s_q} \leq \frac{J^{\text{true}}_{\hat{m}} - J^{\text{true}}_{m^\star}}{s_z} + 2\sqrt{M(1-\rho_{\text{P}})}.$$

Multiplying by s_q and applying the preceding latent-cost bound proves the upper bound in (12). The lower bound follows from the definition of $m^\star$. When $\rho_{\text{P}} = 1$ and $C_{\text{roll}} = 0$, the upper bound vanishes, proving the final claim. □

C Implementation details

C.1 SCALE Pseudocode

Algorithm 1 SCALE training.

Require: $\mathcal{B} = (O, A, Q, e)$: aligned pixels O and states Q over $B \times T$ frames, actions A over $B \times (T-1)$ transitions, and episode IDs $e = (e_b)_{b=1}^B$
Require: E_θ, P_ψ, $K \in 2\mathbb{N}$, $\varepsilon, \delta_{\text{std}} > 0$, and $\lambda_{\text{sig}}, \lambda_{\text{corr}} \geq 0$
$Z \leftarrow E_\theta(O)$; $\widehat{Z} \leftarrow P_\psi(Z_{:,1:T-1}, A)$
$\mathcal{L}_{\text{pred}} \leftarrow \text{MSE}(\widehat{Z}, Z_{:,2:T})$
$z \leftarrow \text{reshape}(Z, (N, d_z))$; $q \leftarrow \text{detach}(\text{reshape}(Q, (N, d_q)))$ ▷ $N = BT$ frames
$b_n \leftarrow \lceil n/T \rceil$; $e_n \leftarrow e_{b_n}$ for $n = 1, \ldots, N$
Sample $K/2$ pairs within sub-trajectories ($b_i = b_j$) and $K/2$ across episodes ($e_i \neq e_j$), with $i \neq j$ and repeated draws allowed; concatenate the draws into $\mathcal{P}$
$x_k \leftarrow \|z_{i_k} - z_{j_k}\|_2^2$; $y_k \leftarrow \|q_{i_k} - q_{j_k}\|_2^2$ for $k = 1, \ldots, K$
if $\text{std}(x) < \delta_{\text{std}}$ **or** $\text{std}(y) < \delta_{\text{std}}$ **then**
    $\mathcal{L}_{\text{corr}} \leftarrow 0 \cdot \text{sum}(z)$ ▷ graph-connected zero
else
    $\tilde{x} \leftarrow (x - \text{mean}(x))/(\text{std}(x) + \varepsilon)$; $\tilde{y} \leftarrow (y - \text{mean}(y))/(\text{std}(y) + \varepsilon)$
    $\mathcal{L}_{\text{corr}} \leftarrow 1 - \text{mean}(\tilde{x} \odot \tilde{y})$
end if
$\mathcal{L}_{\text{SCALE}} \leftarrow \mathcal{L}_{\text{pred}} + \lambda_{\text{sig}}\,\text{SIGReg}(Z) + \lambda_{\text{corr}}\mathcal{L}_{\text{corr}}$
$g_\theta \leftarrow \nabla_\theta \mathcal{L}_{\text{SCALE}}$; $g_\psi \leftarrow \nabla_\psi \mathcal{L}_{\text{pred}}$
$(\theta, \psi) \leftarrow \text{Adam}\big((\theta, \psi), (g_\theta, g_\psi)\big)$

Hyperparameters. The encoder is a ViT-Tiny with patch size 14, 12 layers, 3 attention heads, and width 192. Its CLS token is passed through a linear–BatchNorm projector. The predictor is a six-layer causal transformer with 16 attention heads, dropout 0.1, a history length of 3, and actions injected through zero-initialized AdaLN; its output uses a matching projector. We use frame skip 5 and concatenate the five intervening actions into action blocks, whose dimension is five times the environment action dimension. Training uses four-frame sub-trajectories, batch size 128, and 10 epochs. We optimize with Adam using learning rate 5×10^{-5} and weight decay 10^{-3}. SIGReg uses 1024 random unit projections with $\lambda_{\text{sig}} = 0.09$. For $\mathcal{L}_{\text{corr}}$, we sample K=4096 pairs per training step with equal within- and cross-episode proportions and set $\varepsilon = 10^{-6}$. We use $\lambda_{\text{corr}} = 0.1$ for all tasks. For Aux, h_ϕ is a two-layer MLP, $192 \to 256 \to d_q$, with a ReLU after the first linear layer and no normalization; d_q matches the task-state dimension defined in Appendix D.1. We set λ_{aux} to 0.3, 0.4, 0.1, 0.1, and 0.1 for Push-T, Reacher, Cube, Two-Room, and PointMaze, respectively. This configuration was selected through limited preliminary checks for high state-probe accuracy, without a large-scale architecture or hyperparameter sweep. All reported SCALE and Aux results use these settings; Appendix D.1 evaluates the corresponding full-state alternatives for SCALE.

Planner. All planning solvers use a horizon of five action blocks (25 environment steps), an action-block size of five, a receding-horizon interval of five, a 50-step evaluation horizon, and goals sampled 25 steps ahead. Compute tiers T1–T5 specify candidate count / optimization iterations as 300/30, 100/20, 50/10, 20/5, and 10/3, corresponding to 9000, 2000, 500, 100, and 30 rollout evaluations per replanning step. The elite count is $\max(\text{round}(0.1 \cdot \text{candidates}), 2)$.

Paired evaluation. For CEM and iCEM, each of six evaluation seeds defines a set of 100 episodes sampled without replacement, and every training configuration replays the same 600 episodes for each solver and compute tier. Planner randomness is initialized from a hash of the episode identifier and tier that is shared across training configurations. All reported planning results pool two independently trained runs per method.

C.2 Correlation-weight sensitivity

Table 4: iCEM sensitivity to the SCALE correlation weight. Each cell reports the success rate (%) averaged across T1–T5 on the first line, followed by the five tier values. † marks the value used in the main experiments; its averages are bold. We use a single value of 0.1 across all tasks for a controlled comparison. Selecting λ_{corr} separately for each task gives higher averages on four of five tasks. Results are from a single training run.

Task	$\lambda_{\text{corr}}=0$	0.03	$0.1^{\dagger}$	0.3	1.0
Push-T	63.9 86/81/74/54/26	65.8 88/83/76/55/26	**68.2** 90/87/79/60/26	67.4 89/85/78/58/26	68.3 91/86/80/58/26
Reacher	78.0 86/82/83/78/62	79.6 84/85/85/81/64	**81.1** 87/86/87/82/64	77.3 82/84/81/78/62	79.4 84/84/86/80/64
Cube	65.3 71/68/70/63/55	67.8 73/74/72/64/57	**70.5** 77/76/75/65/60	70.8 77/77/75/65/60	70.1 75/76/76/65/59
Two-Room	82.9 91/93/89/78/63	91.7 100/100/96/88/76	**92.9** 100/100/97/90/77	94.3 100/100/98/92/81	95.5 100/100/98/95/84
PointMaze	80.2 82/83/81/82/74	91.7 97/96/94/88/83	**83.4** 87/83/84/83/79	80.2 84/83/78/80/76	75.2 79/76/75/76/71

C.3 Training and planning cost

Table 5 reports training throughput and planning latency on Cube. Planning uses the largest compute tier T1 for every model and solver, corresponding to $45{,}000$ world-model forward passes per plan. At this budget, LeWM amortizes to approximately $27\,\mu$s per world-model forward pass, compared with approximately $2{,}030\,\mu$s for DINO-WM, a roughly $75\times$ difference.

Training throughput is measured over 300 steps on the same A100 GPU. SCALE adds approximately 13% time per training iteration and Aux adds approximately 4%. DINO-WM trains faster because its encoder is frozen.

Table 5: Training throughput and median per-plan latency on Cube. Planning uses the largest matched compute tier T1.

Model	Training (it/s)	CEM (ms/plan)	iCEM (ms/plan)	Planning vs. LeWM
LeWM	4.35	1,213.3	1,173.0	1.00×
SCALE	3.85	1,195.8	1,184.9	1.00×
Aux	4.20	1,229.4	1,218.8	1.03×
DINO-WM	8.95	91,290.7	91,654.3	76.7×

D Additional representation and state-selection results

Tables 6–10 report held-out R^2 for probes from the full frozen embedding to the components of the selected task state $\mathbf{q}$ used for supervision. Linear denotes closed-form ridge regression. The MLP is a two-layer network with hidden width 256 and ReLU, trained for 30 epochs using Adam

with learning rate 10^{-3}, batch size 1024, and seed 0. Probe inputs contain all retained principal-component coordinates, divided by one global scalar: 1499 for DINO-WM and approximately 192 for the LeWM-family models. Here full-embedding probing means using all retained components. The finite-optimization MLP generally has lower absolute R^2 than the closed-form probe, so comparisons are made within each probe family. Aux denotes the latent-to-state regression control of Sec. 5.

Table 6: Per-component held-out R^2 for Push-T. ϑ is the block angle.

Probe	State component	LeWM	SCALE	Aux	DINO-WM
Linear	agent x	.9191	.9656	**.9851**	.9342
	agent y	.9194	.9715	**.9845**	.9394
	block x	.9686	.9820	**.9904**	.9291
	block y	.9603	.9746	**.9886**	.9722
	$\cos\vartheta$	.9019	.9023	**.9767**	.8353
	$\sin\vartheta$	.8694	.8855	**.9778**	.5943
	Mean	.9231	.9469	**.9839**	.8674
MLP	agent x	.8580	.8876	**.9187**	.7704
	agent y	.8676	**.9206**	.9066	.7676
	block x	.9219	.9005	**.9429**	.7847
	block y	.9149	.8943	**.9475**	.7685
	$\cos\vartheta$	.8572	.8569	**.8980**	.5792
	$\sin\vartheta$	.8404	.8605	**.9116**	.4869
	Mean	.8767	.8867	**.9209**	.6929

Table 7: Per-component held-out R^2 for Reacher. ϑ_0 and ϑ_1 denote the two joint angles; the four displayed sine and cosine components constitute the selected joint-only state $\mathbf{q}$.

Probe	State component	LeWM	SCALE	Aux	DINO-WM
Linear	$\cos\vartheta_0$	.9996	**.9998**	.9996	.9972
	$\sin\vartheta_0$	.9997	**.9998**	**.9998**	.9981
	$\cos\vartheta_1$	.9994	**.9997**	.9995	.9953
	$\sin\vartheta_1$	.9980	**.9996**	.9986	.9702
	Mean	.9992	**.9997**	.9994	.9902
MLP	$\cos\vartheta_0$	.9837	**.9875**	.9854	.9479
	$\sin\vartheta_0$	.9835	**.9882**	.9846	.9245
	$\cos\vartheta_1$	.9780	.9800	**.9820**	.8612
	$\sin\vartheta_1$	.9838	**.9886**	.9830	.7814
	Mean	.9823	**.9861**	.9838	.8788

D.1 Task-state selection sensitivity

The selected state is defined as follows. For Push-T, $\mathbf{q} = (x_{\text{agent}}, y_{\text{agent}}, x_{\text{block}}, y_{\text{block}}, \cos\theta_{\text{block}}, \sin\theta_{\text{block}})$. For Reacher, $\mathbf{q} = (\cos\theta_0, \sin\theta_0, \cos\theta_1, \sin\theta_1)$. For Cube, $\mathbf{q} = (x_e, y_e, z_e, \cos 2\psi, \sin 2\psi, g, x_b, y_b, z_b)$ contains the end-effector position, doubled-angle encoding of the gripper yaw, gripper opening, and block position. For Two-Room and PointMaze, $\mathbf{q}$ is the two-dimensional agent and ball position, respectively.

The paper selection uses a compact set of static variables, removing velocities and highly redundant coordinates according to a simple task prior. Table 11 compares this selection against the full native simulator state on the four tasks where they differ. Two-Room is omitted because its available state has no velocity columns and the paper selection is already the full native state.

The paper and full-state variants are close relative to the reported variability. Seven of the eight task–solver comparisons differ by at most 1.2 percentage points in overall success; the largest difference is the $+2.6$-point change for PointMaze with iCEM, while Cube with iCEM changes by -0.8 points. There is therefore no consistent degradation from including the native velocity or redundant config-

Table 8: Per-component held-out R^2 for Cube. φ denotes the gripper yaw; the doubled-angle encoding accounts for the parallel gripper's π symmetry. The nine displayed components constitute the selected state $\mathbf{q}$.

Probe	State component	LeWM	SCALE	Aux	DINO-WM
Linear	effector x	.9773	.9854	**.9934**	.9814
	effector y	.9661	.9778	.9943	**.9980**
	effector z	.9825	.9899	**.9928**	.9818
	$\cos 2\varphi$	.7599	.9888	**.9938**	.7627
	$\sin 2\varphi$	-.2577	.9880	**.9938**	.4213
	gripper	.9072	**.9783**	.9742	.6638
	block x	.9870	.9922	**.9950**	.9282
	block y	.9884	.9792	**.9924**	.9804
	block z	.9869	.9935	**.9937**	.8871
	Mean	.8109	.9859	**.9915**	.8450
MLP	effector x	**.8781**	.8326	.8630	.7686
	effector y	.9038	.9059	**.9200**	.8992
	effector z	.8420	**.8603**	.8111	.8106
	$\cos 2\varphi$	.6599	.9034	**.9091**	.5339
	$\sin 2\varphi$	-.0394	**.8868**	.8724	.2822
	gripper	.7974	**.8871**	.8627	.4786
	block x	.9083	.8537	**.9145**	.7291
	block y	**.9324**	.9081	.9123	.8928
	block z	.8846	.9024	**.9086**	.5871
	Mean	.7519	.8823	**.8860**	.6647

Table 9: Per-component held-out R^2 for Two-Room.

Probe	State component	LeWM	SCALE	Aux	DINO-WM
Linear	x	.9957	**.9982**	.9963	.9957
	y	.9942	.9977	**.9981**	.9975
	Mean	.9950	**.9979**	.9972	.9966
MLP	x	.9581	**.9742**	.9722	.9315
	y	.9660	**.9829**	.9714	.9209
	Mean	.9621	**.9786**	.9718	.9262

Table 10: Per-component held-out R^2 for PointMaze.

Probe	State component	LeWM	SCALE	Aux	DINO-WM
Linear	x	.9988	**.9992**	.9989	.9982
	y	.9989	**.9990**	.9989	.9982
	Mean	.9989	**.9991**	.9989	.9982
MLP	x	.9703	**.9869**	.9717	.9288
	y	**.9824**	.9733	.9796	.9609
	Mean	.9764	**.9801**	.9757	.9449

Table 11: Sensitivity to the task-state choice used by SCALE. Each CEM and iCEM cell reports the success rate (%) averaged across T1–T5 on the first line, followed by the five tier values (mean ± SD). Results are from a single training run.

Task	SCALE state $\mathbf{q}$	d_q	CEM	iCEM
Push-T	Selected subset	6	73.9±2.2 96±2.0/93±2.6/85±4.0/63±4.6/32±4.4	68.2±2.3 90±2.0/87±2.5/79±3.9/60±4.7/26±5.8
	Full state	8	74.5±2.6 94±2.9/92±2.6/85±3.0/67±5.0/34±5.8	68.3±2.8 90±1.6/87±1.8/79±4.8/59±6.9/26±1.2
Reacher	Selected subset	4	71.5±3.9 82±4.6/79±3.8/75±5.9/64±8.4/57±6.1	81.1±2.0 87±3.6/86±2.9/86±3.4/82±3.2/64±2.5
	Full state	8	72.1±1.0 82±2.4/81±1.0/73±2.8/66±3.4/59±6.1	82.3±1.4 89±2.3/86±2.4/89±2.9/84±4.9/64±2.7
Cube	Selected subset	9	64.1±4.0 74±3.4/71±4.2/66±4.8/56±5.2/54±5.0	70.5±3.3 77±3.9/76±4.5/75±3.5/65±3.7/60±4.5
	Full state	22	64.8±3.1 76±5.0/71±3.5/64±3.9/58±4.1/55±4.8	69.7±3.4 76±3.7/76±4.2/74±4.1/64±4.5/57±3.5
PointMaze	Selected subset	2	86.2±2.4 89±3.7/87±5.1/88±3.3/86±2.3/80±1.9	83.4±2.2 87±3.9/83±3.9/84±4.1/83±4.6/79±2.4
	Full state	4	86.2±3.6 88±4.9/88±4.9/89±4.4/84±4.4/82±4.5	86.0±4.1 88±4.5/88±3.7/87±4.2/85±3.9/81±6.8

uration variables in the distance target. These results indicate that SCALE is not sensitive to the particular compact state selection used in the paper.

D.2 Held-out latent–state distance alignment

We assess held-out distance alignment using Spearman rank correlation ρ_{S} under the protocol of Sec. 4. Whereas SCALE's loss uses Pearson correlation ρ_{P} of distance magnitudes, this diagnostic measures agreement between their ranks. It is therefore a complementary check of monotonic alignment, not the exact quantity optimized by $\mathcal{L}_{\mathrm{corr}}$, and does not by itself establish the mechanism examined in Sec. 6.3.

Table 12: Held-out latent–state distance rank alignment (Spearman ρ_{S}) over 1.12M frame pairs per task. LeWM and DINO-WM repeat Table 1 for comparison.

Task	LeWM	SCALE	Aux	DINO-WM
Push-T	.13	**.77**	.18	.70
Reacher	.18	**.53**	.17	.46
Cube	.00	**.56**	.03	.46
Two-Room	.41	.77	.41	**.85**
PointMaze	.52	**.80**	.52	.50

SCALE substantially increases latent–state rank alignment over LeWM on every task. Aux, in contrast, leaves the alignment essentially unchanged despite making task state highly recoverable from the full embedding. The latter provides an additional check on the distinction motivating SCALE: state decodability alone does not ensure that state differences are expressed in the latent metric.

E Full planning results

Tables 13–17 report success rates for every task, planning solver, and compute tier, including DINO-WM as the pretrained-feature reference. Each tier pools two training runs evaluated on six paired sets, giving $n = 12$ 100-episode measurements. The Overall column averages T1–T5 within each replicate.

Table 13: Pooled Push-T success rates (%). Mean ± SD over two training runs and six evaluation sets per run. The Overall column pools T1–T5. Bold marks the best method per column within each solver.

Solver	Method	T1	T2	T3	T4	T5	Overall
CEM	LeWM	92.3±2.3	89.3±3.2	80.5±4.1	57.8±5.6	33.8±3.6	70.7±2.7
	SCALE	**94.8±3.2**	92.1±2.6	84.2±3.6	61.6±6.9	32.8±5.5	73.1±2.7
	Aux	94.5±2.0	**92.6±2.4**	**84.4±4.1**	**65.1±6.9**	**34.9±6.4**	**74.3±3.4**
	DINO-WM	74.8±4.6	72.1±4.5	65.2±4.9	46.1±4.6	24.3±6.1	56.5±4.0
iCEM	LeWM	86.0±2.4	79.8±2.1	73.4±5.9	55.2±4.8	25.1±5.1	63.9±3.0
	SCALE	89.1±2.0	**86.3±2.8**	**79.7±3.6**	61.3±6.2	25.2±5.5	68.3±2.8
	Aux	**89.9±2.0**	86.2±2.3	79.6±3.4	**62.2±5.3**	**27.6±5.2**	**69.1±2.6**
	DINO-WM	70.0±4.5	65.3±4.5	58.8±5.1	41.5±8.8	23.1±3.7	51.7±4.3

Table 14: Pooled Reacher success rates (%). Mean ± SD over two training runs and six evaluation sets per run. The Overall column pools T1–T5. Bold marks the best method per column within each solver.

Solver	Method	T1	T2	T3	T4	T5	Overall
CEM	LeWM	**82.3±1.8**	**79.4±3.6**	73.3±4.8	**66.2±4.1**	55.8±5.6	71.4±2.3
	SCALE	82.2±3.8	79.1±5.1	74.5±4.4	65.2±6.0	**57.4±5.5**	**71.7±3.1**
	Aux	81.6±4.6	78.8±4.2	**74.9±3.2**	65.8±5.6	55.9±6.8	71.4±3.3
	DINO-WM	77.2±2.8	74.8±5.4	70.8±5.9	61.8±3.4	50.2±7.3	67.0±3.0
iCEM	LeWM	84.9±2.6	84.2±3.4	83.8±3.4	79.0±6.1	62.2±3.6	78.8±1.7
	SCALE	**86.7±3.6**	**87.0±3.1**	**87.3±3.2**	**81.7±3.0**	**64.4±3.0**	**81.4±2.0**
	Aux	85.0±4.4	86.1±3.1	85.8±3.5	80.1±3.7	62.9±5.1	80.0±2.2
	DINO-WM	78.7±4.9	78.2±4.8	80.8±3.5	76.0±4.0	53.0±5.5	73.3±3.1

Table 15: Pooled Cube success rates (%). Mean ± SD over two training runs and six evaluation sets per run. The Overall column pools T1–T5. Bold marks the best method per column within each solver.

Solver	Method	T1	T2	T3	T4	T5	Overall
CEM	LeWM	67.6±4.6	65.8±3.8	60.8±4.6	**56.8±3.5**	53.8±4.0	60.9±3.4
	SCALE	**73.9±3.9**	**71.1±4.1**	**64.7±3.8**	55.9±4.7	54.8±4.6	**64.1±3.6**
	Aux	71.6±4.7	70.2±4.5	62.8±5.2	56.6±4.6	**55.2±4.3**	63.3±3.6
	DINO-WM	73.4±3.1	66.9±2.3	58.4±4.4	51.6±4.5	53.9±3.8	60.9±2.9
iCEM	LeWM	68.9±4.9	67.1±3.6	68.7±4.1	61.9±3.9	54.9±5.7	64.3±4.0
	SCALE	**76.8±3.7**	**76.6±4.0**	**75.2±3.0**	**65.8±3.3**	**59.8±3.9**	**70.8±3.1**
	Aux	73.8±3.6	73.0±4.7	72.5±3.8	64.8±3.4	57.8±3.8	68.4±3.5
	DINO-WM	69.9±2.5	67.4±1.6	66.0±1.7	57.7±3.8	51.4±3.6	62.5±1.3

Table 16: Pooled Two-Room success rates (%). Mean ± SD over two training runs and six evaluation sets per run. The Overall column pools T1–T5. Bold marks the best method per column within each solver.

Solver	Method	T1	T2	T3	T4	T5	Overall
CEM	LeWM	87.8±3.0	87.1±2.8	83.0±3.3	71.8±2.7	57.8±3.9	77.5±1.5
	SCALE	99.7±0.7	99.3±0.8	94.0±2.0	83.5±2.6	69.7±4.2	89.2±0.8
	Aux	88.3±2.7	87.7±2.9	82.4±3.1	70.6±3.1	59.7±2.5	77.7±1.3
	DINO-WM	**100.0±0.0**	**99.8±0.4**	**98.2±1.2**	**94.1±2.2**	**85.8±3.7**	**95.6±1.0**
iCEM	LeWM	91.1±1.6	92.9±1.7	88.5±2.0	77.3±2.2	62.4±2.8	82.5±1.2
	SCALE	99.9±0.3	99.8±0.6	97.7±1.2	90.2±3.0	76.2±2.7	92.7±1.0
	Aux	92.3±2.1	93.2±1.7	89.2±2.0	78.7±2.5	62.9±3.6	83.3±1.1
	DINO-WM	**100.0±0.0**	**100.0±0.0**	**99.8±0.5**	**98.6±0.8**	**92.5±2.1**	**98.2±0.5**

Table 17: Pooled PointMaze success rates (%). Mean ± SD over two training runs and six evaluation sets per run. The Overall column pools T1–T5. Bold marks the best method per column within each solver.

Solver	Method	T1	T2	T3	T4	T5	Overall
CEM	LeWM	85.7±4.9	86.0±4.2	86.9±3.9	83.5±3.5	75.3±5.8	83.5±3.3
	SCALE	87.6±3.8	87.4±3.9	86.8±3.8	84.8±3.9	78.1±3.4	84.9±2.7
	Aux	85.0±4.6	83.9±4.7	82.7±4.8	79.8±4.7	78.1±5.6	81.9±3.9
	DINO-WM	**98.2±1.2**	**98.3±1.4**	**97.2±1.3**	**92.4±3.0**	**87.2±2.9**	**94.7±1.4**
iCEM	LeWM	84.0±4.1	83.6±4.3	82.4±3.1	81.2±4.6	73.2±4.8	80.9±3.5
	SCALE	87.6±3.3	85.8±4.5	84.6±3.2	81.9±4.7	77.0±4.4	83.4±2.5
	Aux	80.8±5.1	79.5±5.1	78.7±5.0	79.4±3.7	75.0±5.8	78.7±4.3
	DINO-WM	**93.4±2.6**	**92.2±2.6**	**90.8±3.4**	**90.3±2.6**	**85.2±2.6**	**90.4±2.1**