MOTIONJEPA: PREVENTING TEMPORAL FEATURE COLLAPSE BY CAPTURING VISUAL CHANGES IN LATENT SPACE

Anonymous authors
Paper under double-blind review

ABSTRACT

Joint Embedding Predictive Architectures (JEPAs) are a promising paradigm for learning task-agnostic latent world models without visual reconstruction. However, standard JEPA training exhibits a strong inductive bias towards slow features, causing feature suppression and the collapse of latent representation. While inverse dynamics provides temporal anti-collapse, it relies on action labels and offers little incentive to embed general, unlabeled dynamics. We introduce **D**ifference **I**mage and **S**ingle image embedding **Reg**ularization (DISReg), a novel regularizer that builds on an inverse-dynamics-style module that predicts temporal difference image embeddings without any pixel reconstruction loss, encouraging balanced static and dynamic feature learning. DISReg consists of a static term that shapes the distribution of the image embedding and encourages slow features, and a dynamic term, which, unlike direct regularization on the embedding, imposes no constraint on the image embedding's shape or distribution and instead only incentivizes that dynamic features be present. By integrating this regularizer into a standard JEPA, we establish our new architecture, MotionJEPA. Latent probing demonstrates that MotionJEPA produces more complete representations than other methods, and our trajectory analysis shows it maintains geometrically simple latent embeddings with low curvature. We further show that MotionJEPA improves downstream planning success under static-background distractors across four environments. Source code will be made publicly available upon publication.

1 INTRODUCTION

Joint Embedding Predictive Architectures (JEPAs) (LeCun, 2022) have emerged as a powerful paradigm for representation learning by predicting future states directly within an abstract embedding space. However, a fundamental question in training JEPA architectures lies in determining exactly which features the model should capture. Previous works (Pathak et al., 2017) suggest encoding features that are either controlled by the agent or can affect the agent, ignoring uncontrollable



Figure 1: **Feature Suppression**. We compare three JEPA models with different collapse-prevention mechanisms in a Pong environment. When models are trained from scratch, visual probes of the image embeddings show that SIGReg-based LeWM (Maes et al., 2026) encodes the slowly changing scoreboard and bats but suppresses the ball. IDM-based SMWM (Ivashkov et al., 2026) encodes only the action-controlled bats. Our DISReg-based MotionJEPA preserves all three features.

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.

background features. Yet the optimal feature hierarchy is inherently problem- and task-dependent. Short-time planning operates on a short timescale, predicting fast-changing dynamics, whereas long-time planning operates on larger timescales with slower-changing features (LeCun, 2022).

Although the primary objective of JEPA frameworks is to learn task-agnostic world models, standard formulations exhibit a well-documented inductive bias toward encoding slow features (Sobal et al., 2022; Team, 2026). In JEPA architectures, this trade-off is mainly controlled by the collapse prevention mechanism (Sobal et al., 2022). We illustrate this in Figure 1, comparing SIGReg-based LeWorldModel (Maes et al., 2026), Inverse Dynamics Module (IDM)-based SMWM (Ivashkov et al., 2026), and our DISReg-based MotionJEPA. The **D**ifference **I**mage and **S**ingle image embedding **Reg**ularization (DISReg) is a novel regularizer that encourages a complementary and balanced embedding of both static and dynamic features.

DISReg is made of two terms. The static term shapes the distribution of the single-image embedding, encouraging the slow features that standard JEPA training already favors. The dynamic term instead predicts the embedding of the temporal difference image from consecutive image embeddings. This dynamic term does not constrain the shape or distribution of the image embedding and only incentivizes that features that change between frames are present, which we show results in geometrically simple latent trajectories with low curvature in Table 2. Our main contributions are:

- **DISReg**: A novel regularization term that incentivizes a balanced embedding of static and dynamic features by utilizing embeddings of the temporal difference images.
- **MotionJEPA**: A novel JEPA framework that utilizes DISReg, demonstrating better downstream performance in the presence of static distractors and robustness to feature collapse.
- A detailed collapse and latent trajectory analysis on a novel collection of three synthetic game environments (Pong, Dino, and Golf) with clearly separable, well-defined features.
- Downstream control evaluation on experiments demonstrating the strengths of MotionJEPA in the presence of static distractors.

2 Related Work

JEPA-style Latent World Models. The Joint Embedding Predictive Architecture (JEPA) is an architectural design that aims to learn latent representations without reconstructing the observation. As latent world models, these models learn to predict the dynamics of a system in latent space. The V-JEPA models (Bardes et al., 2024; Assran et al., 2025; Mur-Labadia et al., 2026) utilize a two-stage training paradigm by first pre-training on unlabeled video and then fine-tuning for the desired task. Recent work has further explored latent predictive models for control and planning, including LeWorldModel (LeWM) (Maes et al., 2026), which learns action-conditioned dynamics directly in representation space, and DINO-WM (Zhou et al., 2024), which builds a world model on top of pretrained visual features for downstream planning tasks with Model Predictive Control (MPC). Temporal Difference in Vision (TDV) (Daithankar et al., 2026) uses the pixel difference between consecutive frames to condition the forward latent prediction without requiring action labels. In our work, the predictor is conditioned on pre-defined action labels; we utilize temporal difference images as an auxiliary representation-learning objective to encourage the latent space to preserve changing scene information.

Collapse Prevention in JEPA. Without a reconstruction loss, the main challenge of JEPAs is to avoid convergence to a trivial solution where all inputs map to a constant latent representation. Contrastive methods use negative samples to explicitly push embeddings apart (Chen et al., 2020). Non-contrastive methods utilize stop-gradient teachers (Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025; Mur-Labadia et al., 2026; Daithankar et al., 2026). Other approaches avoid negative pairs entirely, such as VICReg, which prevents informational collapse by applying explicit variance, invariance, and covariance regularization directly to the embeddings (Bardes et al., 2022). More recent methods avoid computing heavy covariance matrices by shaping the latent space directly into an isotropic Gaussian distribution. Specifically, LeJEPA (Balestriero & LeCun, 2025) introduces SIGReg as its sole collapse-prevention loss, using the Epps-Pulley test (Epps & Pulley, 1983) rather than the Sliced Wasserstein Distance (SWD) (Bonneel et al., 2015) to sketch the distribution. How-

ever, avoiding complete representational collapse does not ensure that the learned representation preserves all task-relevant factors of variation.

Feature Suppression and Slow-Feature Bias. Beyond complete collapse, self-supervised models may exhibit *feature suppression*, retaining only a subset of relevant information despite globally diverse representations (Li et al., 2023; Robinson et al., 2021). In temporal predictive learning, this can appear as a preference for slowly varying features. While JEPA performs on par with or better than reconstruction-based methods in settings where noise changes every time step, JEPA-style models fail when fixed noise or features are present. Sobal et al. (2022) show that JEPAs can encode fixed trajectory-specific distractors while neglecting faster foreground dynamics, a failure later reproduced across different noise timescales and predictive architectures (Gulati & Nemenman, 2026). Similar sensitivity to background changes has been observed in JEPA-based planning (Toso et al., 2026), while recent theory further connects LeJEPA's temporal alignment objective to Slow Feature Analysis (Klindt et al., 2026). LeNEPA (Chemeris et al., 2026) applies SIGReg along the time dimension, and PLDM (Sobal et al., 2025) uses VISReg-style variance and covariance along time and batch dimensions to encourage temporal diversity.

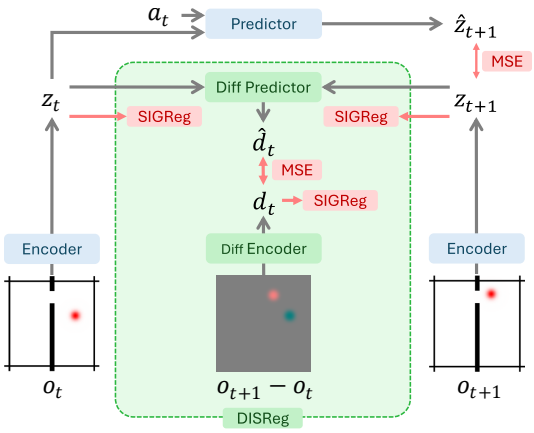


Figure 2: **MotionJEPA Overview**. Our novel regularizer DISReg utilizes a temporal difference image embedding d_t as visually grounded collapse prevention for temporally changing features. The core JEPA encoder and predictor architecture concept is illustrated in blue, and our DISReg in green. The JEPA forward prediction loss, as well as DISReg losses are highlighted in red.

Inverse dynamics provides another way to counteract this bias by forcing the representation to preserve information about action-relevant changes. Sensorimotor World Models (SMWM) (Ivashkov et al., 2026) introduces an inverse model that predicts the action of two consecutive frames, given only the image embeddings. Experiments show that this additional module prevents collapse and enables the model to learn actual controllable degrees of freedom. Similarly, Delta-JEPA (Zhang et al., 2026) also relies purely on the action prediction for collapse prevention, but provides the difference of two consecutive embeddings as input to the action decoder. A core limitation of inverse action prediction is that action labels are required. Additionally, these methods may be less suitable in scenarios where a given action causes a reaction in the observation with a time delay.

These observations motivate our evaluation setting, where we introduce static background distractors into control environments to explicitly test whether the learned representation preserves changing, task-relevant scene information in the presence of slow features. Motivated by the same limitation, our method introduces a visually grounded temporal-difference objective that explicitly encourages the latent representation to preserve changing information without requiring action supervision.

3 Method

Problem formulation. We are given an offline dataset of trajectories $\mathcal{D} = \{(o_1, a_1, \ldots, o_T)\}$ with visual observations $o_t \in \mathbb{R}^{3\times H_{\text{obs}}\times W_{\text{obs}}}$ and continuous action labels $a_t \in \mathbb{R}^{D_a}$. The trajectories are not required to follow a task-specific policy and can be collected from any unknown or arbitrary behavior policy. The goal is to train a JEPA-style latent world model consisting of an encoder $\text{Enc}_\theta : \mathbb{R}^{3\times H_{\text{obs}}\times W_{\text{obs}}} \to \mathbb{R}^{D_z}$ that maps individual observations o_t to a latent state $z_t \in \mathbb{R}^{D_z}$, and a forward predictor Pred_ϕ that estimates the deterministic state transition $\hat{z}_{t+1} = \text{Pred}_\phi(z_{t-H+1:t}, a_{t-H+1:t})$ where H is the size of the history context window. In the JEPA setting, this training process does not involve any reconstruction loss on the visual observations o_t, resulting in the primary challenge of preventing z_t from collapsing to trivial solutions (LeCun, 2022).

Collapse prevention with DISReg. As illustrated in Figure 2, our proposed DISReg introduces two new components, an encoder for encoding the temporal difference image $\text{DiffEnc}_\alpha : \mathbb{R}^{3\times H_{\text{obs}}\times W_{\text{obs}}} \to \mathbb{R}^{D_d}$ and an inverse dynamics module-inspired predictor $\text{DiffPred}_\beta : \mathbb{R}^{D_z} \times \mathbb{R}^{D_z} \to \mathbb{R}^{D_d}$.

$$d_t = \text{DiffEnc}_\alpha(o_{t+1} - o_t) \tag{1}$$

$$\hat{d}_t = \text{DiffPred}_\beta(z_t, z_{t+1}) \tag{2}$$

DiffPred_β learns to predict the difference latent d_t of the temporal difference image given the corresponding state embeddings z_t and z_{t+1}. This design has the advantage that we do not require action or state labels as targets, therefore introducing no bias towards labeled features. Regarding training losses, in contrast to a conventional JEPA design, our architecture has two latent representations z and d that are prone to collapse to a constant. We choose SIGReg (Balestriero & LeCun, 2025) for collapse prevention of these representations, but other distribution regularization losses (Wu et al., 2026; Bonneel et al., 2015) can also be utilized for this purpose.

Algorithm 1. **Pseudocode of DISReg**. Inputs are an observations tensor $o \in \mathbb{R}^{N\times T\times 3\times H_{\text{obs}}\times W_{\text{obs}}}$ with the per-image embeddings tensor $z \in \mathbb{R}^{N\times T\times D_z}$.

```
def DISReg(o, z, w_z=0.25, w_d=2.0, w_pred=0.5):
    # 1. Compute temp. difference images
    o_current = o[:, :-1]
    o_next = o[:, 1:]
    o_diff = o_next - o_current

    # 2. Embed difference images
    d = DiffEnc(o_diff)

    # 3. Predict dynamics from latents
    z_current = z[:, :-1]
    z_next = z[:, 1:]
    d_pred = DiffPred(z_current, z_next)

    # 4. Estimate losses
    L_z = SIGReg(z)
    L_d = SIGReg(d)
    L_pred = (d - d_pred).pow(2).mean()

    # 5. Aggregate losses
    return w_z*L_z + w_d*L_d + w_pred*L_pred
```

$$\mathcal{L}_z = \text{SIGReg}(z) \qquad \mathcal{L}_d = \text{SIGReg}(d) \tag{3}$$

We further establish the prediction loss between the temporal difference image embedding d_t and the respective predictor's embedding $\hat{d}_t$, and aggregate all three loss terms in $\mathcal{L}_{\text{DISReg}}$.

$$\mathcal{L}_{\text{pred}} = \text{MSE}(d_t, \hat{d}_t) \tag{4}$$

$$\mathcal{L}_{\text{DISReg}} = \lambda_z\mathcal{L}_z + \lambda_d\mathcal{L}_d + \lambda_{\text{pred}}\mathcal{L}_{\text{pred}} \tag{5}$$

Crucially, the scalar hyperparameters λ_z and λ_d control the category of information encoded in z. Setting $\lambda_d = 0$ allows d to collapse and DiffPred_β to trivially predict a constant, imposing no additional dynamic feature embeddings on z_t and z_{t+1}, which resembles LeWM (Maes et al., 2026). By setting $\lambda_z = 0$, we maximize the information on embedded dynamics, creating a model that focuses on motions extracted from the temporal difference image. Because $\text{DiffPred}_\beta(z_t, z_{t+1})$ predicts the embedding of visual changes $\hat{d}_t$, minimizing $\mathcal{L}_{\text{pred}}$ results in both z_t and z_{t+1} learning transition-relevant information, preventing the collapse of the image embedding z without imposing additional constraints on the shape or distribution of the embedding. Finally, we join the forward prediction objective of JEPA with our DISReg to obtain the total loss function of MotionJEPA:

$$\mathcal{L}_{\text{MotionJEPA}} = \mathcal{L}_{\text{DISReg}} + \text{MSE}(z_{t+1}, \hat{z}_{t+1}) \tag{6}$$

We provide a pseudocode implementation of the DISReg computation in Algorithm 1, which includes the default hyperparameter weights applied consistently throughout our experiments.

4 Experiments

The experiments are organized into two main sections. Section 4.1 investigates collapse scenarios on game environments with slow and static features by probing the latent embedding z and further analyzing its path straightness. Section 4.2 demonstrates our method's performance on downstream tasks with static distractors. Throughout all experiments, we use the same set of weights for MotionJEPA's DISReg: $\lambda_z = 0.25$, $\lambda_{\text{pred}} = 0.5$, and $\lambda_d = 2$.

Baselines. We compare our method with previous end-to-end JEPA-style latent world models, namely **LeWM** (Maes et al., 2026) and **SMWM** (Ivashkov et al., 2026), with a default SIGReg weight of 0.09 for LeWM and an inverse prediction weight of 1.0 for SMWM. We additionally provide altered versions of LeWM with common practice modifications used in the literature to

Table 1: **Probing Results on Game Environments**. We use high-capacity MLP probes with residual connections to probe the latent embedding z for predicting environment features and report the NMSE averaged over three runs. (a) Probes the image embedding z directly from the encoder; (b) probes on rolled-out states $\hat{z}$ of a rollout length of 5. Runs marked with * are from a hyperparameter search. For (a), based on the results of the random weights model on Pong, NMSE values ≤ 0.1 are marked in green as not collapsed; otherwise, red for collapsed. Best results in each column are highlighted in **bold** and second best underlined.

	Pong				Dino						Golf				
Method	avg.	ball	bats	score	avg.	cactus	clouds	dino	hearts	sky	avg.	bars	cross	ball	wood
ForwardOnly	1.058	1.260	1.001	0.930	0.949	1.003	1.002	1.002	1.031	0.511	1.980	1.233	1.558	1.077	4.324
Random	0.116	0.143	0.093	0.115	0.361	0.833	0.535	0.116	0.031	0.004	0.776	0.857	0.296	0.989	0.653
SMWM	0.657	1.001	0.002	1.002	0.649	0.434	1.075	0.028	0.787	0.071	0.727	1.000	0.986	**0.002**	1.003
LeWM	0.658	0.999	1.002	**0.003**	0.394	0.769	0.248	0.983	0.012	0.005	0.763	1.002	1.001	0.942	0.084
LeWM-Detached	0.171	0.174	0.331	0.007	0.645	1.002	0.727	1.002	0.017	0.006	0.811	0.924	0.683	0.930	0.572
LeWM-Time	1.077	1.100	1.071	1.063	0.913	1.003	0.992	1.002	1.029	0.248	1.053	1.008	1.014	1.004	1.195
LeWM-Flat	0.284	0.890	0.014	0.005	0.331	0.522	0.593	0.002	0.006	0.007	0.694	1.002	0.379	0.993	0.052
PLDM	0.672	0.973	0.835	0.233	0.711	0.978	0.783	0.995	0.400	0.006	0.590	1.000	0.033	0.781	0.032
SMWM*	0.651	0.983	**0.002**	1.003	0.606	0.398	1.005	0.003	0.775	0.069	0.727	1.000	0.986	0.002	1.003
LeWM*	0.061	0.163	0.018	0.012	0.282	0.990	0.059	0.751	0.011	0.012	0.760	1.001	1.002	0.909	0.110
LeWM-Detached*	0.048	0.103	0.038	0.008	0.332	1.005	0.073	0.959	0.011	0.017	0.745	1.002	1.002	0.908	0.048
LeWM-Time*	0.687	0.022	0.972	1.002	0.697	0.005	1.004	0.003	0.971	1.004	0.985	0.972	0.992	0.984	1.002
LeWM-Flat*	0.008	0.012	0.004	0.008	0.301	0.650	0.498	0.002	0.007	0.016	0.299	0.317	0.059	0.312	0.377
PLDM*	0.193	0.571	0.006	0.038	0.353	**0.004**	0.755	**0.001**	**0.003**	**0.003**	0.373	0.997	**0.026**	0.004	**0.021**
MotionJEPA (Ours)	**0.004**	**0.005**	0.003	0.004	**0.021**	0.011	**0.037**	0.006	0.006	0.004	**0.061**	**0.095**	0.083	0.033	0.033

(a) Offline Residual MLP Probe NMSE↓ of Image Embedding z

	Pong				Dino						Golf				
Method	avg.	ball	bats	score	avg.	cactus	clouds	dino	hearts	sky	avg.	bars	cross	ball	wood
ForwardOnly	1.010	1.041	1.006	0.992	1.081	1.019	1.006	1.004	1.017	1.634	1.003	1.002	1.001	1.005	1.005
Random	0.554	0.721	0.478	0.507	0.617	0.974	0.764	0.631	0.279	0.026	0.934	0.995	0.913	0.998	0.781
SMWM	0.630	1.001	0.002	1.002	0.686	0.839	1.002	0.092	0.933	0.076	0.729	1.001	1.001	**0.006**	1.002
LeWM	0.644	1.002	1.002	**0.002**	0.407	0.856	0.251	0.995	0.017	0.005	0.767	1.002	1.001	0.947	0.094
LeWM-Detached	0.239	0.397	0.347	0.007	0.646	0.988	0.732	1.002	0.028	0.006	0.855	0.967	0.821	0.930	0.618
LeWM-Time	1.001	1.003	0.998	1.002	1.106	1.011	1.009	1.003	1.026	1.845	1.003	1.006	1.006	1.000	1.002
LeWM-Flat	0.278	0.968	0.039	0.005	0.396	0.877	0.633	0.055	0.009	0.007	0.722	1.002	0.590	0.994	0.061
PLDM	0.673	1.002	0.835	0.254	0.716	0.992	0.784	1.001	0.417	0.005	0.616	1.002	0.208	0.795	0.036
SMWM*	0.630	1.001	**0.001**	1.002	0.671	0.801	0.999	0.075	0.878	0.079	0.729	1.001	1.001	0.006	1.002
LeWM*	0.138	0.471	0.020	0.009	0.334	0.991	0.060	1.003	0.018	0.012	0.763	1.002	1.002	0.913	0.117
LeWM-Detached*	0.129	0.395	0.053	0.008	0.343	0.988	0.078	1.002	0.020	0.014	0.746	1.002	1.001	0.909	0.051
LeWM-Time*	0.868	0.507	1.002	1.002	0.710	0.120	1.004	0.020	0.959	1.003	0.997	0.997	0.995	0.991	1.002
LeWM-Flat*	0.054	0.183	0.004	0.007	0.353	0.923	0.524	0.066	0.013	0.018	0.448	0.540	0.320	0.402	0.425
PLDM*	0.252	0.861	0.008	0.046	0.361	**0.065**	0.756	**0.011**	**0.004**	**0.005**	0.393	1.001	**0.184**	0.007	**0.021**
MotionJEPA (Ours)	**0.036**	**0.122**	0.004	0.005	**0.055**	0.243	**0.040**	0.051	0.007	0.005	**0.164**	**0.262**	0.348	0.065	0.037

(b) Offline Residual MLP Probe NMSE↓ of Predicted Rollout $\hat{z}$

improve robustness against feature collapse. **LeWM-Time** applies SIGReg along the time dimension (Chemeris et al., 2026), **LeWM-Flat** applies SIGReg along the flattened batch and time dimension, and **LeWM-Detached** adds a StopGradient operator in the JEPA predictor loss target embedding $\text{MSE}(\text{StopGradient}(z_{t+1}), \hat{z}_{t+1})$ (Grill et al., 2020; Assran et al., 2023; Bardes et al., 2024). We include **PLDM** (Sobal et al., 2025), which uses VICReg-style (Bardes et al., 2022) variance and covariance along batch and time dimensions separately, together with temporal-alignment and inverse dynamics loss terms. Besides these baselines, we also include **ForwardOnly**, which has only the forward prediction loss and no anti-collapse mechanism, and **Random**, with fixed, randomly initialized weights, showing what information is present in the latent at initialization that might be suppressed during training. Finally, we also perform a hyperparameter search for each baseline environment combination to demonstrate that the collapse is not an issue of hyperparameter selection but a fundamental limitation of JEPA anti-collapse mechanisms (Sobal et al., 2022). We mark such models with a * throughout this work.

4.1 Collapse and Latent Embedding Analysis

Environments. We designed three fully synthetic, deterministic game environments with clearly separable feature groups. Samples for each environment are generated on the fly during training and evaluation to prevent overfitting[1]. Initial states and actions are sampled randomly for each episode

[1]Episode sampling is seeded to ensure that all methods use identical training and evaluation data.

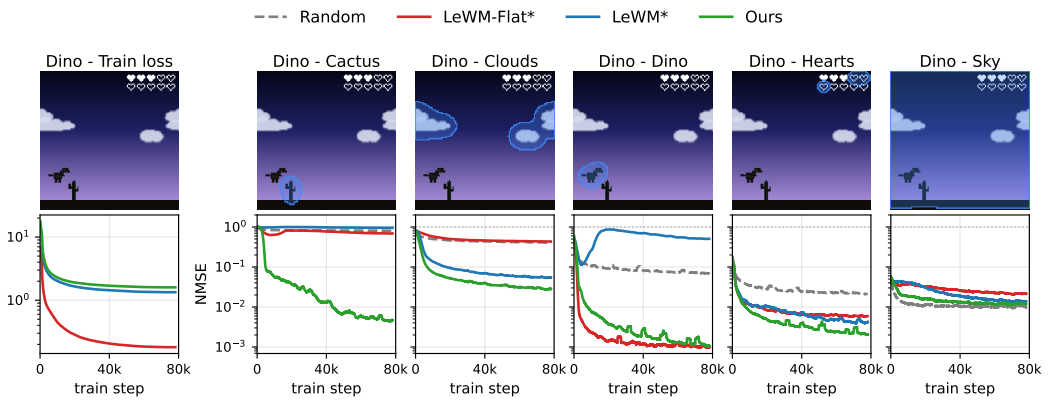


Figure 3: **State Probing during Training**. We train a detached state decoder alongside training the JEPA model. The first column of each environment is the training loss (log scale). The remaining curves are per-feature probe NMSE. Image areas affected by the corresponding feature are highlighted in blue. Models marked with * are optimized with a hyperparameter search. The fixed-weight Random baseline has no training loss.

to minimize correlations between features. This reduces the risk of shortcut learning (Geirhos et al., 2020); for example, in Pong, a probe estimating the bat's position solely from the ball's location would falsely suggest that bat information is encoded when it is not. We propose three environments of increasing dynamical complexity: **Pong**, where the agent controls both bats to hit the ball; **Dino**, a 2D side-scroller where an agent must jump over a cactus; and **Golf**, in which the player navigates a ball toward a hole while avoiding moving obstacles. We design the state dimension of each environment (excluding velocities: Pong: 6, Dino: 8, Golf: 6) to be significantly lower than the latent embedding dimension D_z of 192 to ensure that the latent dimensionality is not a bottleneck that artificially induces collapse. Implementation details are provided in Appendix E.

Evaluation Setup. We follow (Sobal et al., 2022) and analyze the collapse of features by probing the learned image embedding z. To keep the comparison fair, all methods use the same image encoder, latent forward predictor, learning rate schedule, and optimizer from (Maes et al., 2026). After training from scratch on 10M randomly sampled episodes, we freeze the base model, train probes on 1M training samples, and evaluate those probes on 10k episodes. For the collapse analysis, we choose higher-capacity MLP probes with residual connections instead of linear probes, as we are not interested in the linear separability or ease of accessibility of features, but rather in measuring the general presence or suppression of information in the latent embedding z. For example, a world model which uses a non-linear, highly correlated representation in z might still be functional, yet linear probes might fail to decode that information. We report Normalized Mean Squared Error (NMSE) for each probe. An NMSE of 1.0 corresponds to predicting the evaluation target mean, indicating a complete failure to extract useful information from z. We train and evaluate each method-environment pair for three seeds and report the average results. We provide more details about the training and evaluation process as well as architectures of the probes in Section E.

Probing Results. Table 1 shows the MLP probing results on Pong, Dino, and Golf for both the encoder embeddings z as well as predicted rollouts $\hat{z}$. The goal is to obtain the lowest average loss across all categories for each environment, indicating stable and balanced learning of all features[2]. Looking at the results of probing the image embedding z, as expected, the control method ForwardOnly collapsed on all three tasks, resulting in an NMSE greater than or about 1.0 for all features except the sky color of Dino, with 0.511, which is arguably the easiest feature of these environments. Switching to the Random model, we observe that even though the weights are random, the probes are still able to extract a significant amount of information, with an average NMSE of 0.116 for Pong and even 0.004 for the sky in the Dino environment. Models performing worse than this baseline therefore indicate active suppression of information available at initialization.

[2]The avg. column is a weighted mean, where each feature group's weight corresponds to its dimensionality.

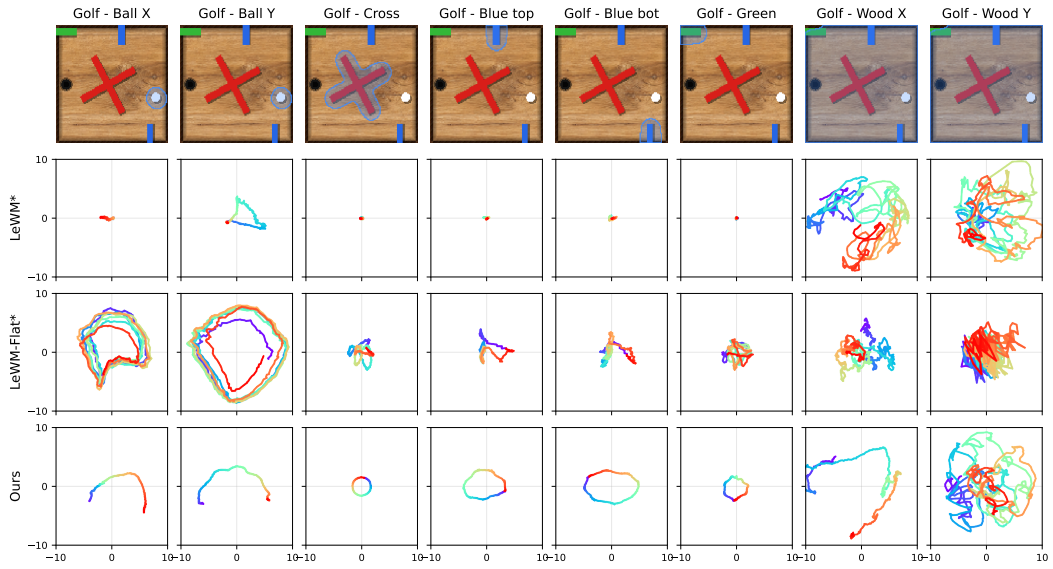


Figure 4: **Latent Trajectories**. We visualize how individual state features map into the latent space. For each subplot, we vary a single feature across a 1D range while holding the remaining state variables fixed, render the resulting sequence into images, and encode them into latent representations z. We then project these latent trajectories onto their first two principal components without variance rescaling. We further color the state range in rainbow colors from 0: violet to 1: red.

Training LeWM with default settings results in feature suppression across all three environments, embedding only slow features such as the score in Pong. While the hyperparameter-optimized version LeWM* improves the average probing loss on Pong from 0.658 to 0.061, in the presence of more dynamic features, as in Dino and Golf, such hyperparameter optimization does not significantly improve results (see Appendix Figure 7), highlighting a core limitation of regularizing along the batch dimension. PLDM* and the LeWM-Flat* and LeWM-Time* variants add regularization along the time dimension. LeWM-Time* and PLDM* successfully encode fast-changing features such as the cactus in Dino, yet fail in the presence of multiple moving objects, suppressing the clouds in Dino or bars in Golf. SMWM*, using an IDM, performs well on features directly affected by actions but largely fails to capture other information.

Our method achieves the lowest average feature probing loss across all three environments for both image embedding and rollout prediction while using the **same set of hyperparameters** across all three environments. MotionJEPA is also the only method that achieves a balanced probing loss below 0.1 on all categories, indicating robustness against feature collapse. Especially for the bar obstacle category of Golf, where MotionJEPA obtains 0.095 while all other methods suppress this category with losses higher than 0.3. We also provide results with linear probes in Table 8 of the appendix, in which our model also achieves the lowest average probing loss across all environments.

When does collapse happen? In Figure 3 we show the training loss curves with additional online residual MLP probes for the Dino environment. The probes are trained on the z embedding with a stop-gradient operator to prevent information leaking to the encoder. For LeWM*, which applies SIGReg along the batch dimension, the slow features of the sky, hearts, and clouds suppress the embedding of the cactus and dinosaur. This is especially visible for the dinosaur, as the probing loss first decreases to approximately 0.1 at 5k iterations and then increases back to 1.0 at 20k iterations, from which it is not able to recover. LeWM-Flat*, which enforces a distribution across both batch and time dimensions, seeks variance along the time dimension of the embedding as well and therefore quickly picks up the dinosaur. This is probably because its limited movement of jumping is very simple, but provides consistent temporal change. Crucially, the collapse is not visible in the training loss (see first column), demonstrating that monitoring loss alone is insufficient to diagnose collapse without explicit latent probing or downstream evaluation.

Latent Path Straightness. The temporal straightening hypothesis (Hénaff et al., 2019) states that useful representations turn curved input dynamics into near-straight latent paths. It has proven informative in practice, as a signal for identifying AI-generated video (Internò et al., 2025), for judging physical plausibility (Internò et al., 2026), and as a planning objective (Wang et al., 2026). We show empirical results of the latent path straightness in Table 2. SMWM achieves the straightest latent trajectories close to the ideal straight-line value of 1.0 in both Pong and Golf, as it has no distribution regularization on z. Introducing distribution regularization along the time axis, such as in variants LeWM-Time* and LeWM-Flat* causes the opposite effect, significantly increasing the latent embedding curvature to above 20 for both Pong and Golf. As our method is architecturally closer to SMWM's IDM module, we observe straighter paths than all distribution regularization methods on Pong and Golf.

Table 2: **Latent path straightness**. We calculate the mean encoder arc length over the chord length for 100 random linear state-space interpolations. Lower is straighter, with 1.0 being a straight line.

Method	Pong	Dino	Golf
ForwardOnly	45.514	**4.045**	26.412
Random	25.278	13.660	26.842
SMWM	**1.207**	7.557	**1.078**
LeWM	5.170	<u>4.826</u>	8.471
LeWM-Detached	3.067	18.372	15.455
LeWM-Time	29.911	4.955	27.190
LeWM-Flat	3.031	12.986	5.298
PLDM	8.060	17.686	4.725
SMWM*	<u>1.427</u>	8.631	<u>1.078</u>
LeWM*	8.170	7.755	8.191
LeWM-Detached*	4.111	8.806	7.935
LeWM-Time*	23.376	9.979	38.839
LeWM-Flat*	22.159	15.610	26.663
PLDM*	4.740	13.065	14.147
MotionJEPA (Ours)	2.756	5.789	3.305

This can also be observed in Figure 4, where we show example latent trajectories of each feature of the golf environment. Our method produces nearly straight latent embeddings for ball position and single-loop embeddings for the circular cross and bar obstacles[3]. LeWM* collapses all obstacles into almost a single point while showing high-curvature embeddings in the distracting wood background. While the representations of the LeWM-Flat* variant do not collapse into a single point, the embedding has more curvature and entanglement, making it difficult to separate and predict individual features. Additional latent trajectory visualizations are provided in the Appendix G.

4.2 Downstream Tasks

As in LeWorldModel (Maes et al., 2026), probing is intended primarily as an analysis of representation content rather than to substitute for closed-loop control evaluation. We take the experimental setup of LeWM (Maes et al., 2026) and evaluate the model's success rate in the presence of slow features.

Environments. All four environments originally contain no static or slow visual features. To introduce a static-background distractor, we replace the original background with a wood-texture image, mimicking games played on a wooden table. For each episode, a different crop of the image texture is sampled and kept fixed throughout it. For PushT and TwoRoom, the original white background pixels are directly replaced with the wood texture. For Reacher and OG-Cube, we first estimate a static-background mask using the pixel-wise median over the training dataset and replace the corresponding regions with the texture. Example observations from all four environments are shown in Figure 5. All models are trained for 10 epochs with a batch size of 64 using AdamW (Loshchilov & Hutter, 2019) with a learning rate of 5×10^{-5} and weight decay of 10^{-3}.

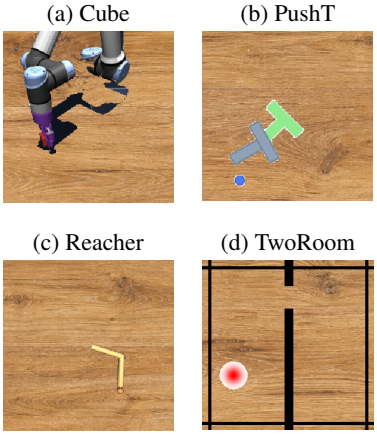


Figure 5: Example observations from the four LeWorldModel environments with a fixed per-episode cropped wood-texture background used as a static visual distractor.

Evaluation Setup. We use CEM-based planning following the protocol of Maes et al. (2026). For IDM, we use our own implementation of inverse-dynamics supervision, predicting the raw action from $\Delta z_t = z_{t+1} - z_t$ with a three-layer MLP. This follows the Δz input formulation of Delta-JEPA (Zhang et al., 2026). Initial states are randomly sampled from the test split, and we evaluate planning horizons of 25 and 50 steps.

[3]The bar obstacles reappear on the opposite side if exiting the image, resulting in looped trajectories.

Table 3: Planning success rate (%) under static-background distractors. Our method gives the best standalone mean performance, while combining it with IDM yields the best overall results.

	25-step planning					50-step planning				
Method	Cube	PushT	Reach.	2Room	Mean	Cube	PushT	Reach.	2Room	Mean
LeWM	43.6	4.0	5.2	28.8	20.4	24.4	2.4	2.0	18.4	11.8
LeWM-Flat	55.2	14.4	49.2	48.0	41.7	42.4	2.0	60.4	22.8	31.9
IDM	76.0	**83.6**	49.2	**100.0**	77.2	53.2	26.0	62.0	96.0	59.3
Ours ($\lambda_z = 0$)	60.4	76.8	24.8	99.6	65.4	48.0	21.2	33.6	94.0	49.2
Ours	78.0	78.4	71.2	99.6	81.8	**63.2**	28.0	98.0	92.0	70.3
Ours + IDM	**80.8**	82.4	**82.4**	**100.0**	**86.4**	61.2	**29.6**	**98.4**	**97.2**	**71.6**

Downstream Results. The results are summarized in Table 3. Overall, the original LeWM baseline degrades substantially in the presence of the static-background distractor, particularly on PushT and Reacher, while LeWM-Flat provides a stronger baseline but still remains clearly below IDM and our method in mean performance. IDM improves robustness across the four environments, while our standalone method achieves stronger overall mean performance, with particularly large gains on Cube and Reacher under the longer 50-step planning horizon. Combining our objective with IDM yields the best overall results, further indicating that the two training signals are complementary. On TwoRoom, both approaches achieve near-saturated performance, whereas the original LeWM baseline remains considerably lower. We further probe task-relevant states from the frozen representations on the four LeWorldModel benchmarks with static wood-texture distractors. Our method substantially improves state recoverability over LeWorldModel, consistent with its stronger downstream planning performance. Detailed probing and reconstruction results are in Section F.

Ablation Studies. For the four LeWM tasks, our method adds a temporal difference-image prediction objective to LeWM. As shown in Table 3, removing regularization on the original latent representation ($\lambda_z = 0$) degrades performance, indicating that global state information remains important alongside change-aware supervision. Applying the same objective to IDM-based LeWM also improves 25-step planning, showing that it complements inverse-dynamics supervision.

5 Limitations

DISReg introduces two additional modules, DiffEnc_α and DiffPred_β, both increasing the parameter count and compute cost relative to a plain JEPA baseline. However, the training loss curves in Figure 3 show that collapse is not resolved simply by training longer or with more capacity. The baselines plateau in NMSE despite continued training, indicating that feature suppression is not an issue of compute budget. Furthermore, the modules introduced by DISReg are only relevant during training and are discarded afterward, adding no computational overhead during inference. Compared to LeWM, which uses a single hyperparameter, DISReg introduces three tunable hyperparameters. However, we demonstrate in the appendix Appendix A that the performance of our method is generally robust, allowing us to use a single, consistent set of parameter values across all our experiments. As both the game environments and LeWM benchmarks are synthetic, we provide additional probing results on BridgeData V2 (Walke et al., 2023) in the appendix Section D, showing that our model significantly outperforms other non-IDM-based baselines.

6 Conclusion

We introduced DISReg, a regularizer that encourages JEPA world models to encode dynamic features by predicting an embedding of the temporal difference image, without action labels or pixel reconstruction. Integrated into MotionJEPA and using one fixed set of DISReg loss weights throughout, our model captures both static and dynamic features more completely than prior methods while maintaining simple latent trajectories. This translates to improved downstream planning under static-background distractors. Overall, encouraging dynamics through prediction rather than direct latent constraints offers a simple, effective route to collapse-resistant JEPA world models.

AI USE STATEMENT

In this work, we used generative AI tools to *assist coding / implementing methods*. We have not used generative AI tools to *generate synthetic data sets*[4], *help develop theoretical models or conceptual frameworks*, *propose or refine hypotheses*, *design or provide feedback on research methodology or experiments*, *clean and reformat dataset*, *support qualitative and thematic data analysis*, *interpret results*, and *formulate mathematical claims*, *provide critical ingredients for proving mathematical claims*, *assist in the writing of proofs*, *assist with translation*, are not applicable to this work.

We further use generative AI for polishing the writing, but not for drafting sections, paragraphs, etc. Generative AI is also used to aid the retrieval and discovery of related work.

We have reviewed all AI-assisted work, including the generated code and polished writing. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

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

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

Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics, 2025. URL https://arxiv.org/abs/2511.08544.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *ICLR*, 2022.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video, 2024. URL https://arxiv.org/abs/2404.08471.

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and Radon Wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision*, 51(1):22–45, 2015. doi: 10.1007/s10851-014-0506-3.

Alexander Chemeris, Ming Jin, and Randall Balestriero. Lenepa: No-augmentation next-latent prediction for time-series representation learning, 2026. URL https://arxiv.org/abs/2607.00958.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chen20j.html.

Ninad Daithankar, Alexi Gladstone, Yann LeCun, and Heng Ji. You don't need strong assumptions: Visual representation learning via temporal differences, 2026. URL https://arxiv.org/abs/2606.15956.

[4]Here we are referring to not generating synthetic data directly from AI tools. The images of each game environment are fully generated by code, using shapes and mathematical functions.

Thomas W. Epps and Lawrence B. Pulley. A test for normality based on the empirical characteristic function. *Biometrika*, 70(3):723–726, 1983.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020. doi: 10.1038/s42256-020-00257-z.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21271–21284. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf.

Paarth Gulati and Ilya Nemenman. Contrast encodes inductive bias: separating slow noise from dynamics in predictive representation learning. *arXiv preprint arXiv:2606.07770*, 2026.

O J Hénaff, R L T Goris, and E P Simoncelli. Perceptual straightening of natural videos. *Nature Neuroscience*, 22(6):984–991, Apr 2019. doi: 10.1038/s41593-019-0377-4. URL https://www.nature.com/articles/s41593-019-0377-4.

Christian Internò, Robert Geirhos, Markus Olhofer, Sunny Liu, Barbara Hammer, and David Klindt. AI-generated video detection via perceptual straightening. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=LsmUgStXby.

Christian Internò, Alexander Pondaven, Habon Issa, Fabio Pizzati, Francesco Pinto, Markus Olhofer, Ivan Laptev, Philip Torr, Eero P. Simoncelli, Barbara Hammer, and David Klindt. Geophys: The geometry of physical plausibility, 2026. URL https://arxiv.org/abs/2606.20707.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 448–456, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/ioffe15.html.

Petr Ivashkov, Randall Balestriero, and Bernhard Schölkopf. Sensorimotor world models: Perception for action via inverse dynamics, 2026. URL https://arxiv.org/abs/2606.20104.

David Klindt, Yann LeCun, and Randall Balestriero. When does lejepa learn a world model? *arXiv preprint arXiv:2605.26379*, 2026.

Yann LeCun. A path towards autonomous machine intelligence, 2022. URL https://openreview.net/forum?id=BZ5a1r-kVsf. OpenReview position paper.

Tianhong Li, Lijie Fan, Yuan Yuan, Hao He, Yonglong Tian, Rogerio Feris, Piotr Indyk, and Dina Katabi. Addressing feature suppression in unsupervised visual representations. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1411–1420. IEEE, 2023.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. URL https://arxiv.org/abs/2603.19312.

Lorenzo Mur-Labadia, Matthew Muckley, Amir Bar, Mido Assran, Koustuv Sinha, Mike Rabbat, Yann LeCun, Nicolas Ballas, and Adrien Bardes. V-jepa 2.1: Unlocking dense features in video self-supervised learning, 2026. URL https://arxiv.org/abs/2603.14482.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017.

Joshua Robinson, Li Sun, Ke Yu, Kayhan Batmanghelich, Stefanie Jegelka, and Suvrit Sra. Can contrastive learning avoid shortcut solutions? *Advances in neural information processing systems*, 34:4974–4986, 2021.

Vlad Sobal, Jyothir S V, Siddhartha Jalagam, Nicolas Carion, Kyunghyun Cho, and Yann LeCun. Joint embedding predictive architectures focus on slow features. In *NeurIPS Workshop on Self-Supervised Learning*, 2022.

Vlad Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Stress-testing offline reward-free reinforcement learning: A case for planning with latent dynamics models. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025. URL https://openreview.net/forum?id=jON7H6A9UU.

Enigma Team. The obsessed encoder, 2026. URL https://enigma.inc/posts/obsessed-encoder. Accessed: 2026-08-29.

Leonardo F Toso, Davit Shadunts, Yunyang Lu, Nihal Sharma, Donglin Zhan, Nam H Nguyen, and James Anderson. Learning invariant visual representations for planning with joint-embedding predictive world models. *arXiv preprint arXiv:2602.18639*, 2026.

Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023.

Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim G. J. Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Ik1mKtUYlZ.

Haiyu Wu, Randall Balestriero, and Morgan Levine. Visreg: Variance-invariance-sketching regularization for jepa training, 2026. URL https://arxiv.org/abs/2606.02572.

Zhenghao Zhang, Yuanxiang Wang, Zhenyu Guan, Yujia Yang, Bingkang Shi, Tianyu Zong, Hongzhu Yi, Guoqing Chao, Xingchen Chen, Tiankun Yang, Chenxi Bao, Tao Yu, Jingjing Zhou, and Jungang Xu. Delta-jepa: Learning action-sensitive world models via latent difference decoding, 2026. URL https://arxiv.org/abs/2606.31232.

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 MotionJEPA Hyperparameter Sensitivity

We analyze the sensitivity of our novel DISReg to changes in the λ_z, λ_d, and λ_{pred} hyperparameters. In Figure 6, we vary each of these parameters and show the resulting change in each of the three game environments. For Dino and Golf, we observe higher sensitivity, as in Pong, as these environments are more complex. In Dino, all three parameters achieve better results than LeWM-Flat* and PLDM* for the full range of 0.05 to 5. In general, we find that lower weights on the static term λ_z, higher weights on the dynamic term λ_d, and setting λ_{pred} to approximately 0.5 works consistently well on all three environments.

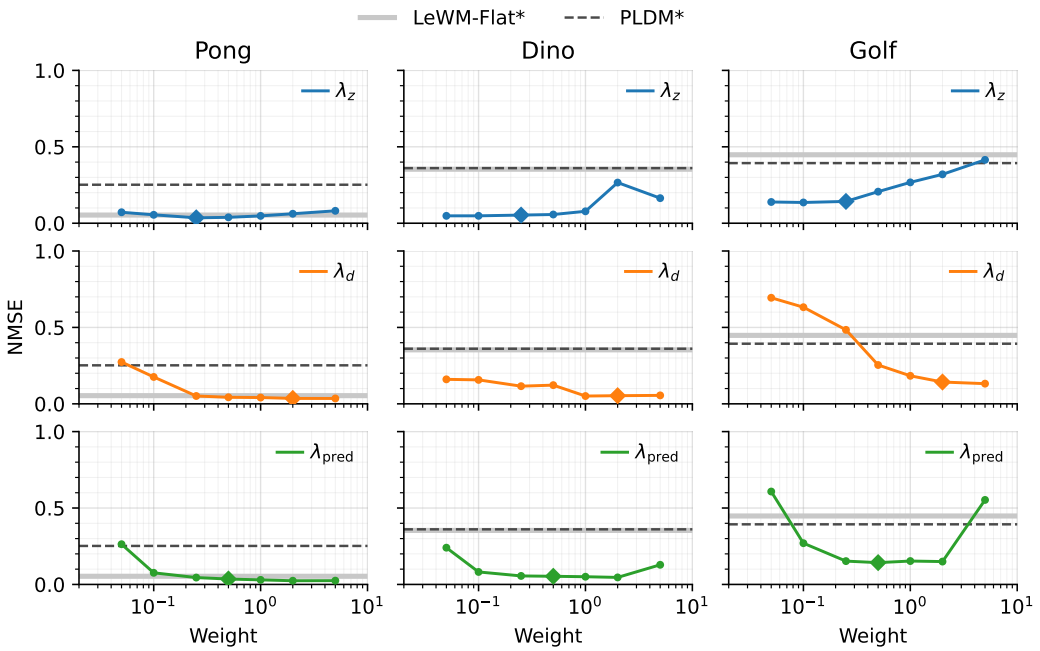


Figure 6: **Hyperparameter Sensitivity of DISReg.** We show the impact of changing each of the three hyperparameters λ_z, λ_d, and λ_{pred} introduced by DISReg. Reported is the average NMSE of all features for probing the 5-step residual MLP probes. Horizontal lines are the average results of LeWM-Flat* and PLDM* three seed runs, which are also shown in Table 1. The selected hyperparameters for MotionJEPA are highlighted with a diamond shape.

B Baselines Hyperparameter Search

For baselines trained on the game environments, we perform a grid search over a full 10M-episode run. For SMWM (Ivashkov et al., 2026), we grid-search the action-prediction loss weight; for LeWM (Maes et al., 2026), LeWM-Detached, LeWM-Time, and LeWM-Flat, we grid-search the SIGReg weight. For PLDM, we use the loss terms and default weights from (Maes et al., 2026):

$$\mathcal{L}_{\text{PLDM}} = \mathcal{L}_{\text{pred}} + \lambda_{\text{var}}\mathcal{L}_{\text{var}} + \lambda_{\text{cov}}\mathcal{L}_{\text{cov}} + \lambda_{\text{t-sim}}\mathcal{L}_{\text{time-sim}} + \lambda_{\text{t-var}}\mathcal{L}_{\text{time-var}} + \lambda_{\text{t-cov}}\mathcal{L}_{\text{time-cov}} + \lambda_{\text{IDM}}\mathcal{L}_{\text{IDM}} \tag{7}$$

With six tunable weights, an exhaustive grid search is not feasible, so we focus our search on the temporal VICReg weights $\lambda_{\text{t-std}}$ and $\lambda_{\text{t-cov}}$, as these are by design most relevant for embedding dynamic features. We also found that a general dynamics term of $\lambda_{\text{IDM}} = 1$ improves performance, so we keep it fixed at this value. The selected configuration is the one with the lowest NMSE for offline residual MLP probing, averaged over all state groups. We use the probe of the predicted rollout $\hat{z}$ instead of the direct image embedding z, as we are interested in a world model that is able to predict future states accurately, not a pure image embedding. In Table 4 we show each model's chosen hyperparameter, as well as the corresponding search space. For methods with only one hyperparameter, we show line plots in Figure 7; PLDM is shown in Figure 8.

Table 4: **Hyperparameter Search Configurations and Selected Parameters**. We show the search space of each method together with the selected hyperparameters for each environment and the default settings. The selected parameters are used for the starred (*) methods throughout this work.

Method	Search space	Selected			Default
		Pong	Dino	Golf	
SMWM	$\lambda_{\text{IDM}} \in \{0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100\}$	100	0.2	1	1
LeWM	$\lambda_{\text{SIGReg}} \in \{0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100\}$	100	2	0.1	0.09
LeWM-Detached	$\lambda_{\text{SIGReg}} \in \{0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100\}$	0.2	20	0.02	0.09
LeWM-Time	$\lambda_{\text{SIGReg}} \in \{0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100\}$	10	2	20	0.09
LeWM-Flat	$\lambda_{\text{SIGReg}} \in \{0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50, 100\}$	1	0.2	1	0.09
PLDM	$\lambda_{\text{t-var}} \in \{0.2, 0.35, 0.7, 1.4, 5, 10, 20\}$	0.2	20	5	0.7
	$\lambda_{\text{t-cov}} \in \{0, 0.1, 0.5, 1, 2, 5, 10, 20, 50, 100\}$	100	50	20	0
	$\lambda_{\text{IDM}} \in \{1\}$	1	1	1	0
	λ_{var}=18, λ_{cov}=12, $\lambda_{\text{t-sim}}$=0.2 (fixed)				
MotionJEPA (Ours)	λ_z=0.25, λ_{pred}=0.5, λ_d=2 (fixed)				

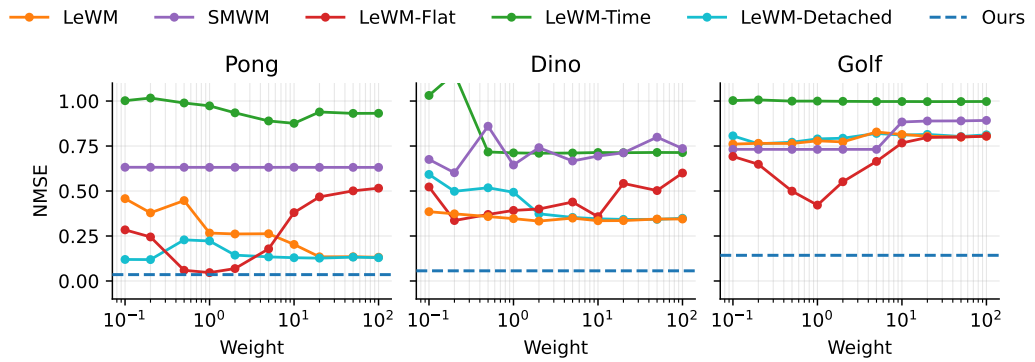


Figure 7: **Hyperparameter Search Visualization.** The NMSE loss is estimated from offline residual MLP probes on the predicted 5-step rollout $\hat{z}$. In Golf, which has more dynamic features than the other two games, varying the hyperparameter has minimal impact for all listed methods except LeWM-Flat, with the probing loss remaining above 0.7. For LeWM-Flat, we observe a sharp minimum at a SIGReg weight of 1.0. Our method (horizontal dashed line) performs significantly better on Golf and Dino, and achieves similar performance to LeWM-Flat on Pong, despite using a single, fixed set of hyperparameters across all three environments.

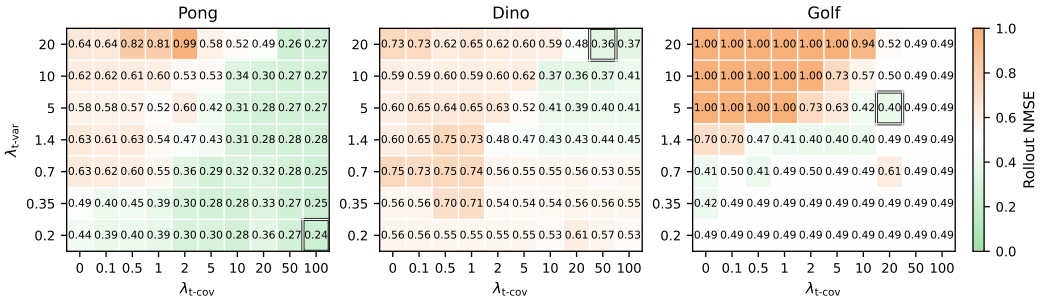


Figure 8: **Hyperparameter Search for PLDM.** We show the NMSE loss from offline residual MLP probes on the predicted 5-step rollout $\hat{z}$. We observe that $\lambda_{\text{t-cov}}$, the weight of the covariance along the time axis, is quite stable across environments, achieving the best results in the range of 20 to 100. However, $\lambda_{\text{t-var}}$ requires very different settings for each environment to work properly, ranging from 0.2 in Pong to 20 in Dino and 5 in Golf.

C Measuring Latent Path Straightness

For each environment, we estimate the path straightness shown in Table 2 by sampling $N = 100$ pairs of start and end states from the state variables defined in Section E.2, with two exceptions: the score in Pong and the wood texture offset in Golf. We exclude the Pong score because its 0 to 99 range introduces non-continuous, looping visual features as the smaller digit rolls over. Similarly, the Golf wood texture offset is omitted because it produces texture crops that lack a meaningful linear structure. We then compute $T = 64$ linearly interpolated states for each trajectory. These states are rendered as images and encoded with the respective image encoder Enc_θ, yielding a tensor of latent trajectories $z \in \mathbb{R}^{N \times T \times 192}$. Finally, we compute the path straightness as the average ratio of the arc length to the chord length:

$$\frac{1}{N} \sum_{i=1}^{N} \frac{\sum_{t=1}^{T-1} \|z_{i,t+1} - z_{i,t}\|_2}{\|z_{i,T} - z_{i,1}\|_2} \tag{8}$$

We use this ratio because it is less sensitive to the choice of interpolation step size than cosine-based curvature, which depends on step granularity and is especially noisy for the small, low-resolution images in our setting.

D Experiments on BridgeData V2

Setup. BridgeData V2 (Walke et al., 2023) is a large and diverse dataset of robotic manipulation behaviors that provides both action and state labels. Filtering for human-teleoperated demonstrations yields 24,825 episodes with a total of 960,823 frames. We sample frames exclusively from the primary over-the-shoulder camera and resize them from 640×480 to 112×112. Example clips of the processed frames are shown in Figure 10. As we can not evaluate the downstream planning success rate on pre-recorded videos, we follow the experimental setup established in Section 4.1. We train the models on 10M randomly sampled 4-frame clips, freeze the encoder, and train probes on 1M episodes using the learned image embeddings z. For each model's hyperparameter configuration, we utilize the median of the parameters found via the grid search on the game environments (Table 4).

BridgeData V2 provides the following state and action labels:

State Variables: end-effector x (m), end-effector y (m), end-effector z (m), roll (rad), pitch (rad), yaw (rad), gripper (0=closed, 1=open).

Actions: Δx (m), Δy (m), Δz (m), Δroll (rad), Δpitch (rad), Δyaw (rad), gripper command (0=close, 1=open).

As we do not have labels for all features, we also train a 3-layer patch decoder visual probe. It starts from a fixed 8×8 grid of 2D sine–cosine tokens (14×14 patches on the 112×112 image, width 256). In each layer, the latent is mapped by a 2-layer GELU MLP and added to every token, then a residual GELU MLP (width 1024) is applied per token. A linear head turns each token into an RGB patch and is trained with a pixel MSE loss. No attention mechanism is used for this decoder.

Results. The probing results on BridgeData V2 are shown in Table 5. We split the evaluation into two groups: one with and one without using inverse dynamics (IDM) as regularization. Our primary interest is in regularization without action labels, as we aim to learn dynamic features without the need for action annotation. Compared to LeWM and LeWM-Flat, we achieve better probing results for the residual MLP probe, as well as the linear and visual probes. Probing the end-effector position with a residual MLP probe demonstrates that our method successfully captures dynamic features (ours: 0.086, LeWM-Flat: 0.258, LeWM: 0.304). Simultaneously, it maintains a balanced, general representation of the scene, yielding a visual probe loss of 0.10, compared to 0.16 for both LeWM baselines. When augmenting MotionJEPA with an inverse dynamics model (IDM) and comparing it to PLDM and SMWM, we achieve results comparable to SMWM. This demonstrates that our model integrates effectively with an additional IDM component. Adding an IDM improves performance since every probed state has a corresponding action label. Finally, online probing results during training (Figure 9) reveal that the poor position probe performance of LeWM and LeWM-Flat is not due to insufficient training time, but rather early saturation during training.

Table 5: **Probing Results on BridgeData V2.** We evaluate probing on the learned image embedding z. We mark the best results in **bold** and underline the second-best for both groups (with and without inverse dynamics). We report the MSE for the visual probe, with images normalized to $[0, 1]$.

Method	Offline Residual MLP Probe NMSE↓				Offline Linear Probe NMSE↓				Visual Probe MSE↓
	avg.	position	orientation	gripper	avg.	position	orientation	gripper	
w/o IDM									
LeWM	0.480	0.304	0.523	0.871	0.805	0.662	0.896	0.968	0.016
LeWM-Flat	0.517	0.258	0.661	0.877	0.782	0.601	0.904	0.978	0.016
MotionJEPA (Ours)	**0.354**	**0.086**	**0.490**	**0.764**	**0.556**	**0.213**	**0.786**	**0.923**	**0.010**
with IDM									
PLDM	0.267	0.056	0.462	0.347	0.390	0.094	0.679	0.463	0.012
SMWM	0.228	0.054	0.414	**0.223**	**0.340**	0.095	**0.615**	**0.300**	0.014
MotionJEPA (Ours) + IDM	**0.207**	**0.042**	**0.371**	0.238	0.355	**0.082**	0.649	0.342	**0.010**

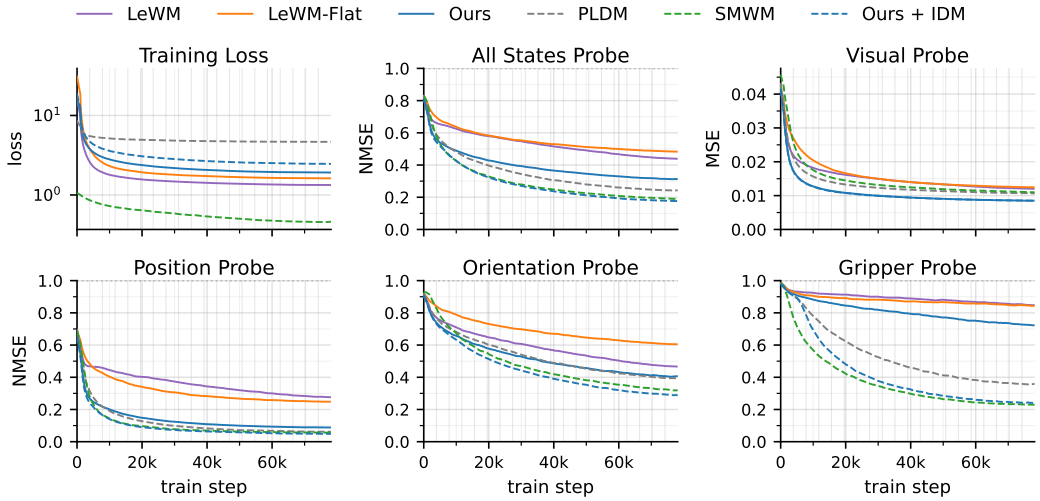


Figure 9: **Probing during Training on BridgeData V2.** We train detached state and visual decoders alongside the JEPA training on the image embedding z. Dashed lines indicate methods that use inverse dynamics, while solid lines represent those that do not.



Figure 10: **Sample Clips of BridgeData V2.** We show six randomly sampled 4-frame clips from the BridgeData V2 (Walke et al., 2023) dataset. We exclusively use the over-the-shoulder camera to maintain a consistent viewing angle, ensuring alignment with the action and state labels. The dataset features multiple scenes with various objects to manipulate, yet action and state labels are only provided for the robot arm.

E Implementation Details

E.1 Model and Component Architectures

Encoder and Predictor Architecture. We use the same encoder and predictor architectures as (Maes et al., 2026). The encoder is a ViT-Tiny Vision Transformer with a patch size of 14×14 pixels. As in (Maes et al., 2026), the $D_z = 192$ dimensional embedding vector is obtained from the [CLS] token of the final layer, followed by a 1-layer MLP with Batch Normalization (Ioffe & Szegedy, 2015). The predictor is a six-layer causal transformer with 16 attention heads and learned positional embeddings (Maes et al., 2026). We use a history size $H = 3$ for all experiments.

DISReg Components. We use the same architecture for DiffEnc_α as for Enc_θ, with an output vector dimensionality $D_d = 192$ for all experiments except the LeWM Benchmarks, where we set $D_d = 8$. The difference predictor DiffPred_β is a three-layer MLP with hidden layers of size 512 that uses LayerNorm and GELU after all but the final linear layer. The input is a concatenated vector of the two consecutive embeddings z_t and z_{t+1}.

Residual MLP Probe. The residual MLP probe has the highest capacity and is intended to check the general presence of information, not ease of accessibility. It is an MLP of depth 6 and width 128 with residual connections between every second layer and pre-activation batch normalization.

MLP Probe. We train a two-layer MLP on top of the frozen latent representation to measure non-linear accessibility of the states, comprising a linear layer with 512 hidden units, a GELU activation, and a final linear output layer. Angular targets are represented by their sine and cosine values and trained with MSE; angular error is computed after decoding the predicted angle.

Linear Probe. We train a single affine linear layer, including a bias term, on top of the frozen latent representation. No hidden layers, normalization, or activation functions are used. For LeWM benchmark problems, angular targets are represented by their sine and cosine values and trained with MSE; angular error is computed after decoding the predicted angle.

E.2 Game Environments

For all games, the states and actions are normalized to the range $[0, 1]$. Initial states for the first frame and actions are sampled independently from a uniform distribution. These states are then simulated deterministically with the given actions to generate a total of 4 states and corresponding frames. Sample episodes of each dataset are shown in Figure 11. Because velocity cannot be estimated from a single image, we ignore velocity features when probing and evaluating the image embedding z and rollouts $\hat{z}$. Throughout the following environment descriptions, we use * to indicate which state variables are considered for evaluation.

E.2.1 Pong

State Variables: left bat position*, right bat position*, ball x-position*, ball y-position*, ball x-velocity, ball y-velocity, score left player*, score right player*.

Actions: left bat velocity, right bat velocity.

The scores, positions of the ball, and the players' bats are uniformly initialized. If a player hits the ball, the ball's trajectory changes towards the opponent. The change in angle is dependent on the bat's position, and the horizontal velocity accelerates with each hit up to a maximum velocity. If one player scores a goal, the score of the respective player increases by one, and the ball is reset to the center, moving towards the player who was scored against. The task requires the model to estimate the ball's velocity in order to predict the ball's next position.

E.2.2 DINO

State Variables: dinosaur y-position*, dinosaur y-velocity, cactus x-position*, cactus x-velocity, normalized number of hearts*, sky color*, clouds height*, cloud 0 scroll*, cloud 1 scroll*, cloud 2 scroll*.

Actions: jump.

A side scroller in which the goal is to time the dinosaur's jumps to avoid upcoming cacti. Clouds each move with an individual fixed velocity for each episode. If the dinosaur collides with one of the cacti, the number of hearts is reduced by 1 (corresponding to a change of 0.1 in the normalized state vector). Once all lives are depleted, the game freezes for all remaining frames.

E.2.3 GOLF

State Variables: ball x-position*, ball y-position*, cross angle*, cross angular velocity, blue bar 1 x-scroll*, blue bar 1 scroll x-velocity, blue bar 2 x-scroll*, blue bar 2 x-scroll velocity, green bar y-scroll*, green bar y-scroll velocity.

Actions: ball x-velocity, ball y-velocity.

In this game, the player navigates a ball through a 2D course of moving obstacles. Each obstacle is initialized at a random position, together with a random velocity that is fixed for each episode. The bars, when exiting the screen on one side, reappear on the other side of the screen. Colliding with obstacles stops the game, freezing it for the remaining frames and ignoring all action inputs. As the red cross is symmetric to a quarter of a full rotation, we normalize the angle $[0, 90]$ to a range $[0, 1]$.

E.2.4 DETAILS FOR TRAINING ON GAME ENVIRONMENTS

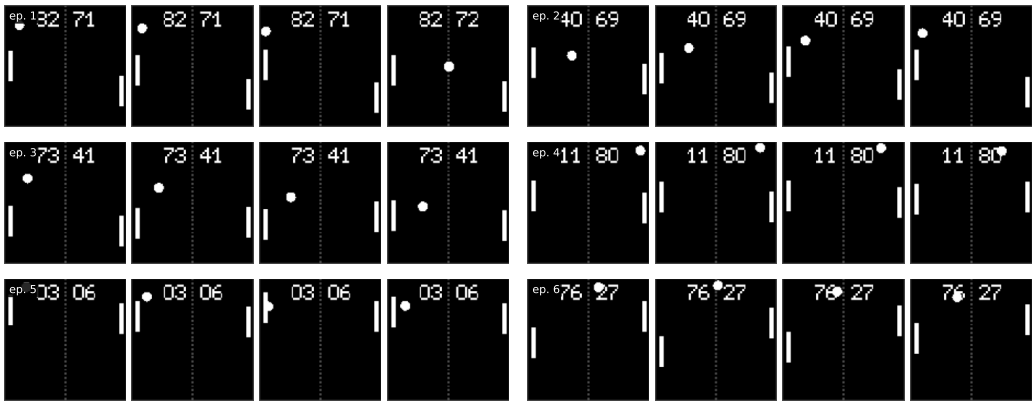
Pre-training on Game Environments. Experiments on the three game environments are conducted at a resolution of 112×112 pixels with no skip frames (i.e., simple next-frame prediction). We train the models on 10 million four-frame episodes with a batch size of 128, resulting in approximately 78k training steps. We use the AdamW (Loshchilov & Hutter, 2019) optimizer with a weight decay of 10^{-3}. For the learning rate, we employ a linear-warmup cosine-annealing schedule, matching the LeWM (Maes et al., 2026) defaults: the learning rate rises linearly from 0 to 5×10^{-5} over the first 1% of training (781 steps), then decays to zero following a cosine schedule over the remaining steps. This schedule is stepped after every optimizer update.

Fitting Probes on Game Environment Models. After pre-training, we load and freeze the weights of the resulting encoder Enc_θ and predictor Pred_ϕ. We generate 1 million eight-frame episodes and, for each episode, embed the first three frames into latent vectors $z_{1:3}$ using the encoder Enc_θ. Next, we use the first three embeddings $z_{1:3}$ as context and apply the predictor Pred_ϕ autoregressively with a sliding window of size three and conditioned on the episode's ground-truth actions to generate a rollout of five predicted embeddings $\hat{z}_{4:8}$. Either image embeddings $z_{1:3}$ or predicted rollout embeddings $\hat{z}_{4:8}$ are then used as inputs for the residual MLP or linear probes. This results in a total of four probes: a linear probe on image embeddings, a linear probe on rollout embeddings, a residual MLP probe on image embeddings, and a residual MLP probe on rollout embeddings. The training target for each probe is the state vector detailed in Section E.2. We use the AdamW optimizer with a weight decay of 10^{-3}, a fixed learning rate of 3×10^{-3}, and a batch size of 128. We evaluate each probe's performance on an additional 10k episodes.

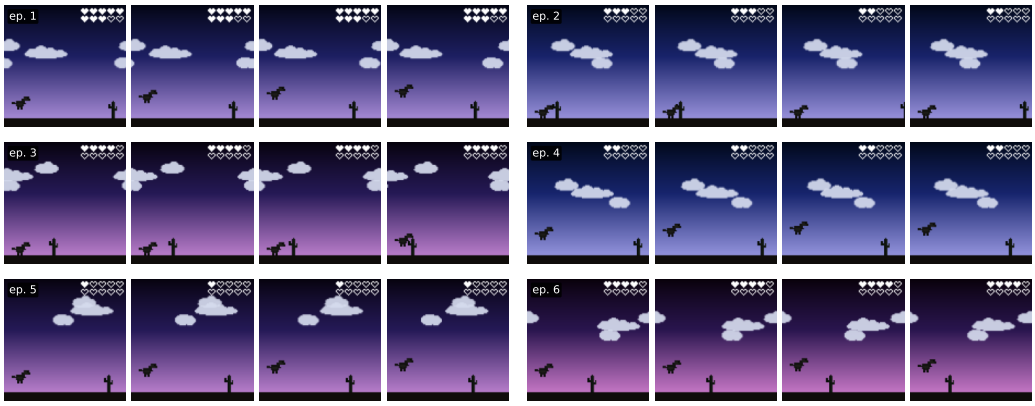
E.3 LEWM BENCHMARKS

For experiments on **PushT**, **Two-Room**, **OG-Cube**, and **Reacher**, we use the setup of (Maes et al., 2026) together with the provided expert rollouts for each task. As in (Maes et al., 2026), we apply a 5-frame skip and group the consecutive actions between sampled frames into a single action vector. For the 25-step setting, we follow the CEM planning configuration of (Maes et al., 2026), using 300 candidate sequences, 30 elites, and the same environment-specific optimization budget. With a frame skip of 5, CEM plans five model steps ahead, corresponding to 25 environment steps. For the 50-step setting, we keep the 25-step CEM setup unchanged except for doubling the planning budget, resulting in doubled number of plannings over the trajectory.

(a) Pong



(b) Dino



(c) Golf

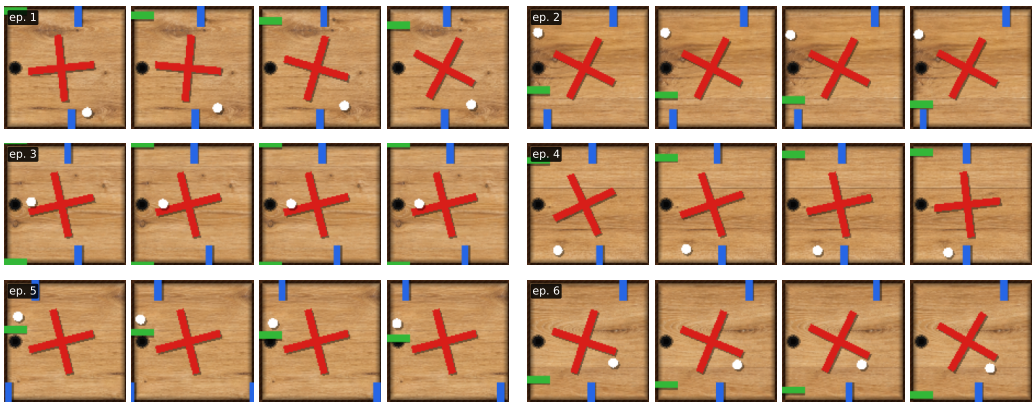


Figure 11: **Randomly sampled episodes of game environments**. We show six randomly sampled 4-frame episodes of each game environment. Each episode is the same length as used during training. Episode one of Pong shows the right player scoring, and episode two of Dino demonstrates the player losing one heart by running into the cactus.

Table 6: Physical-state probing on **PushT** and **TwoRoom** with a static wood-texture distractor. Translational quantities are reported as mean Euclidean error in the native environment coordinates, while angular error is the mean absolute angular error in radians. Lower is better, with the best MLP and linear probe results in **bold**.

		PushT			TwoRoom
Method	Probe	Agent pos. ↓	Object pos. ↓	Object angle ↓	Agent pos. ↓
LeWM	Linear	131.61	87.03	0.859	47.01
LeWM	MLP	121.17	73.66	0.788	45.33
LeWM-Flat	Linear	81.80	50.63	0.754	39.44
LeWM-Flat	MLP	55.67	28.88	0.419	37.01
IDM	Linear	3.99	**24.47**	**0.670**	3.71
IDM	MLP	**3.66**	18.01	0.413	3.29
Ours	Linear	17.46	27.33	0.763	6.34
Ours	MLP	8.60	18.20	0.397	1.48
Ours+IDM	Linear	**3.58**	26.52	0.726	**1.55**
Ours+IDM	MLP	4.32	**17.12**	**0.367**	**0.73**

Table 7: Physical-state probing on **OG-Cube** and **Reacher** with a static wood-texture distractor. Position errors are mean Euclidean errors, while angular quantities are mean absolute errors in radians. Lower is better, with the best MLP and linear probe results in **bold**.

		OG-Cube				Reacher			
Method	Probe	Agent xyz	Object xyz	Agent yaw	Object yaw	Joint 0	Joint 1	Fingertip xy	Target xy
LeWM	Linear	0.182	0.187	1.394	**1.436**	1.468	1.378	0.169	**0.124**
LeWM	MLP	0.130	0.126	**0.588**	**0.529**	0.735	0.880	0.126	0.052
LeWM-Flat	Linear	0.132	0.151	1.417	1.508	0.292	1.078	0.126	0.125
LeWM-Flat	MLP	0.0162	0.0546	1.325	1.382	0.111	0.553	0.0738	0.121
IDM	Linear	**0.0030**	**0.0068**	**1.371**	1.539	1.385	0.895	0.168	0.125
IDM	MLP	**0.0026**	**0.0046**	1.274	1.522	1.162	0.820	0.166	0.125
Ours	Linear	0.0236	0.0214	1.403	1.473	0.0265	0.1246	0.0241	0.124
Ours	MLP	0.0113	0.0103	0.732	0.708	0.0141	0.0321	0.0061	**0.0463**
Ours+IDM	Linear	0.0040	0.0089	1.374	1.482	**0.0139**	**0.0392**	**0.0107**	0.124
Ours+IDM	MLP	0.0048	0.0059	0.808	0.841	**0.0100**	**0.0234**	**0.0042**	0.0487

F State Probing on LeWorldModel Benchmarks with Static Wood-Texture Backgrounds

We evaluate the task-relevant physical quantities used for controlling each environment with offline MLP state probes. For **PushT**, we probe the 2D position of the agent, the 2D position of the T-shaped object, and the object orientation. For **TwoRoom**, we probe the 2D agent position. For **OG-Cube**, we probe the 3D positions and yaw angles of both the robot end-effector and the manipulated object. For **Reacher**, we probe the two robot joint angles, the 2D fingertip position, and the 2D target position. For translational variables, we report the mean Euclidean distance in the native coordinates of the corresponding environment. For angular quantities, we report the mean absolute angular error in radians. Lower values are better in all cases. We compare LeWorldModel, LeWM-Flat, IDM, our method, and our method combined with IDM. For LeWorldModel, we use the original loss weights $\lambda_{\text{sigreg}} = 0.09$ (Maes et al., 2026). For LeWM-Flat and SMWM, we use the median of the found hyperparameters in Table 4, resulting in the weights of $\lambda_{\text{sigreg}} = 1.0$ and $\lambda_{\text{IDM}} = 1$.

Analysis. The probing results reveal substantial differences in the task-relevant physical information retained by the learned representations under static background distractors. LeWM performs poorly across several dynamic state variables, particularly on PushT, TwoRoom, and Reacher. LeWM-Flat improves recoverability in several cases, suggesting that stronger regularization over flattened batch and temporal dimensions partially alleviates the problem, but a substantial gap re-

mains to the dynamics-aware objectives. For example, on PushT the MLP agent-position error decreases from 121.17 for LeWM to 55.67 for LeWM-Flat, while our method reaches 8.60.

Our method substantially improves physical-state recoverability across the evaluated environments. The effect is especially pronounced on Reacher, where the linear probe obtains joint-angle errors of 0.0265 and 0.1246,rad and a fingertip-position error of 0.0241, compared with 1.468, 1.378,rad, and 0.169 for LeWM. The strong linear performance indicates that the controllable robot state is not only retained but also organized in a readily accessible form. IDM shows complementary behavior, providing particularly strong translational probing on PushT and OG-Cube, while remaining substantially weaker than our method on Reacher joint and fingertip states.

Combining our objective with IDM further improves several controllable variables, including TwoRoom agent position and Reacher joint and fingertip states. Overall, the results support the interpretation that static-background degradation is associated with the representation emphasizing highly predictable visual content at the expense of task-relevant dynamics. Stronger SigReg alone, as in LeWM-Flat, helps but does not fully resolve this issue, while our objective and IDM provide stronger supervision for preserving dynamic physical state. Their combination further improves recoverability for several variables, consistent with the complementary planning gains observed in the main experiments.

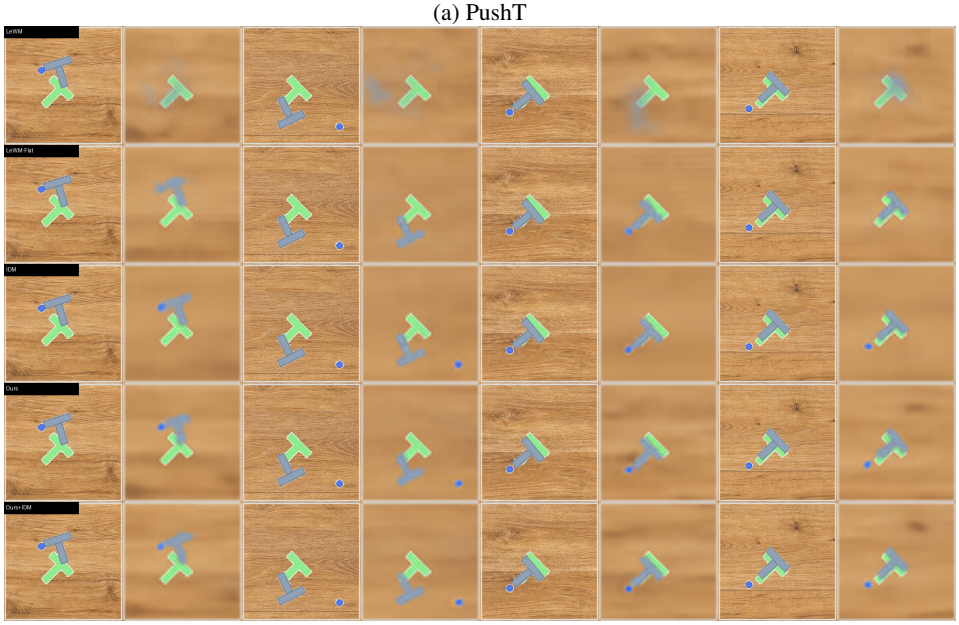
F.1 Reconstruction-Based Visualization of Latent Information

To complement the quantitative probing results, we train a lightweight image decoder on top of each frozen latent representation as an offline visual probe. Figures 12 and 13 show representative reconstructions for all four environments. Within each row, original observations and their latent reconstructions are shown alternately, with methods ordered as LeWM, LeWM-Flat, IDM, ours, and Ours+IDM.

The qualitative results are consistent with the state probing results. LeWM often reconstructs the dominant static appearance while only weakly recovering the task-relevant foreground state. LeWM-Flat improves foreground reconstruction in several environments, indicating that stronger regularization already alleviates part of this effect, but the recovered dynamic state remains less consistent than for the dynamics-aware objectives. IDM, Ours, and Ours+IDM generally preserve moving agents, manipulated objects, and articulated robot configurations more clearly. This difference is particularly visible on TwoRoom and OG-Cube, where LeWM reconstructions largely lose the foreground configuration, and on Reacher, where the dynamics-aware representations recover the robot pose substantially more clearly.

At the same time, our reconstructions still retain coarse scene appearance rather than becoming invariant to static visual content. Fine wood-texture details are largely smoothed out, while broad background structure remains recoverable. This is desirable because static or slowly varying scene information may still encode useful context such as goals or obstacles. Together with the quantitative probes, these results suggest that the proposed objective reduces the dominance of highly predictable static appearance while preserving both task-relevant dynamics and coarse contextual information.

(a) PushT



(b) TwoRoom

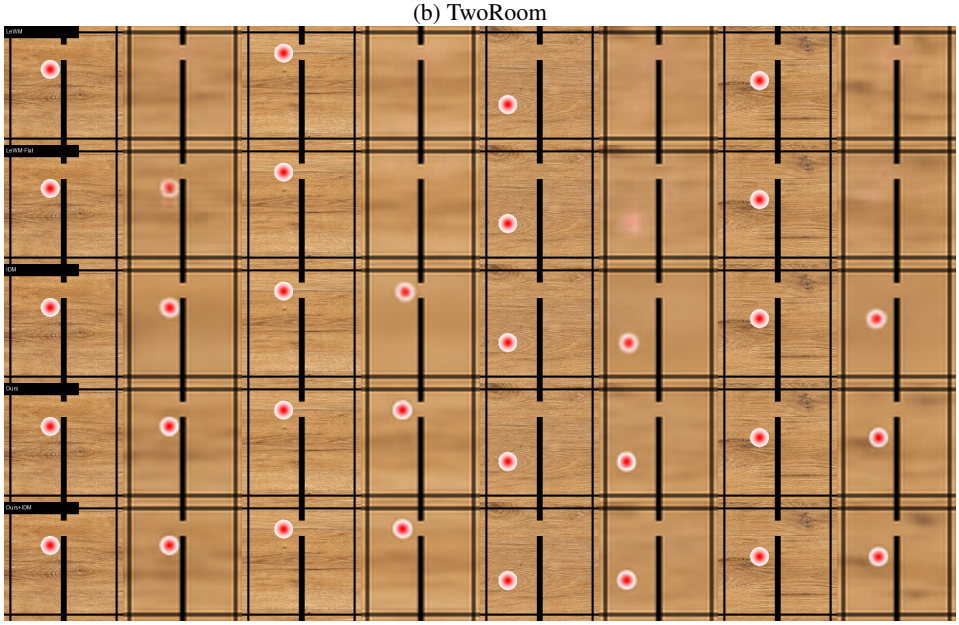
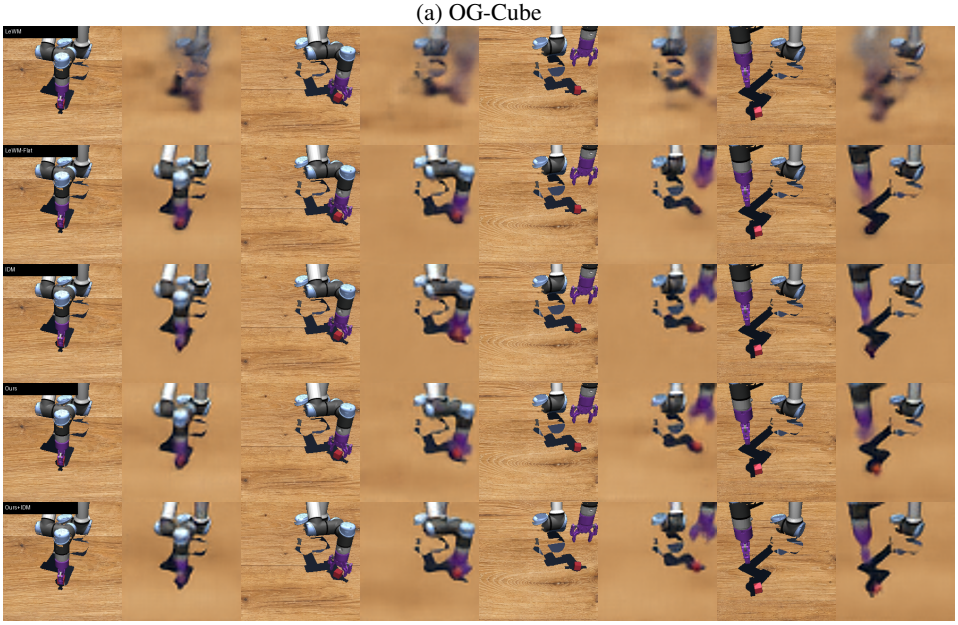


Figure 12: **Qualitative reconstruction from frozen latent representations on PushT and TwoRoom.** Rows within each environment correspond to LeWM, LeWM-Flat, IDM, our method, and Ours+IDM, with original observations and their reconstructions shown alternately. LeWM often reconstructs the dominant static appearance while only weakly recovering the task-relevant foreground. LeWM-Flat improves foreground recovery, while IDM, our method, and Ours+IDM generally preserve the moving entities more clearly.

(a) OG-Cube



(b) Reacher

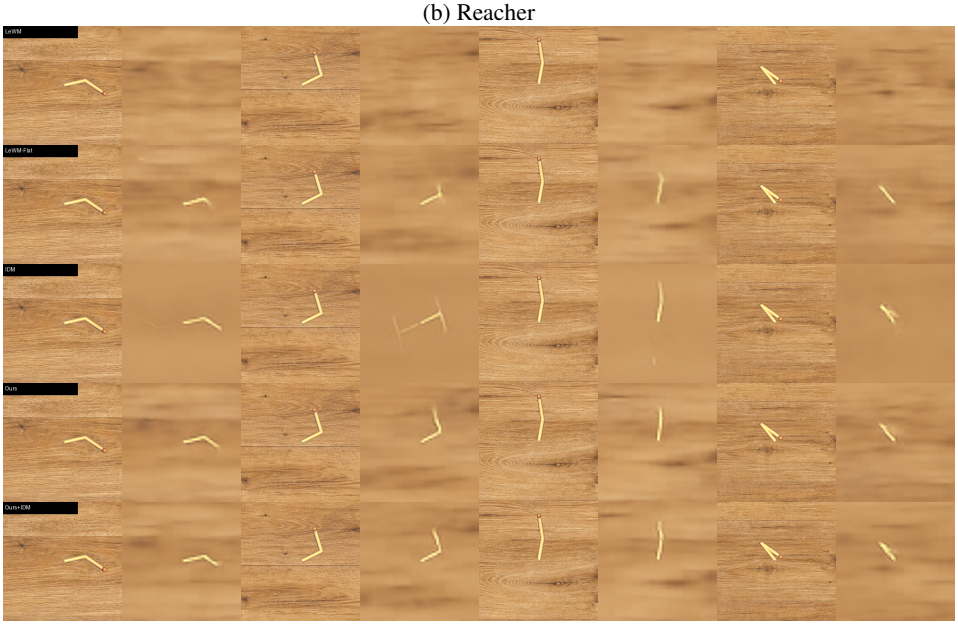


Figure 13: **Qualitative reconstruction from frozen latent representations on OG-Cube and Reacher.** Rows within each environment correspond to LeWM, LeWM-Flat, IDM, our method, and Ours+IDM, with original observations and their reconstructions shown alternately. Dynamics-aware representations recover the articulated foreground state substantially more clearly than LeWM, while LeWM-Flat provides an intermediate improvement. This is particularly visible on Reacher, where the robot configuration is more faithfully reconstructed despite the dominant wood-texture background.

G Extended Tables and Figures

This section provides extended tables and figures to supplement the results presented in Section 4.1. We show results for linear probes in Table 8. Our method achieves the lowest overall loss when probing both the image embedding z and the predicted rollout $\hat{z}$ across all game environments.

Variance Across Training Seeds. We provide additional tables detailing the standard deviation across three random seed runs in the game environments. Results for MLP probes with residual connections are shown for Pong in Table 9, Dino in Table 10, and Golf in Table 11.

Latent Trajectories. We also provide expanded latent trajectory plots comparing all baseline methods evaluated in Section 4.1. Each game environment is detailed in its own figure: Pong in Figure 14, Dino in Figure 15, and Golf in Figure 16.

Table 8: **Offline linear probes**. Our method achieves the lowest average feature probing loss across all environments, indicating not only good feature coverage but also better linear separability.

	Pong				Dino						Golf				
Method	avg.	ball	bats	score	avg.	cactus	clouds	dino	hearts	sky	avg.	bars	cross	ball	wood
ForwardOnly	0.965	0.999	1.000	0.899	0.911	1.002	0.999	1.001	1.001	0.239	1.002	1.002	1.001	1.002	1.004
Random	0.507	0.607	0.390	0.532	0.536	0.931	0.717	0.484	0.089	**0.002**	0.917	0.973	0.808	1.003	0.788
SMWM	0.662	1.008	**0.000**	1.013	0.655	0.643	1.003	0.044	0.887	0.099	0.748	1.027	1.027	**0.000**	1.031
LeWM	0.722	1.087	1.086	0.025	0.558	1.033	0.502	1.045	0.030	0.051	0.975	1.086	1.064	1.005	0.727
LeWM-Detached	0.411	0.706	0.429	0.126	0.762	1.082	0.858	1.054	0.149	0.204	0.985	1.020	0.778	1.035	0.979
LeWM-Time	0.993	0.994	0.995	0.989	0.904	1.000	0.993	0.999	0.993	0.205	1.000	1.000	1.000	1.000	0.997
LeWM-Flat	0.391	1.069	0.120	0.048	0.445	0.785	0.767	**0.021**	0.026	0.012	0.947	1.090	0.542	1.069	0.797
PLDM	0.777	1.096	0.918	0.345	0.771	1.075	0.842	1.045	0.489	0.021	0.798	1.085	0.122	0.910	**0.576**
SMWM*	0.651	0.945	0.001	1.037	0.665	0.703	1.000	0.041	0.956	0.083	0.748	1.027	1.027	0.000	1.031
LeWM*	0.221	0.387	0.111	0.183	0.498	1.082	0.362	1.051	0.043	0.029	0.973	1.085	1.074	0.972	0.756
LeWM-Detached*	0.251	0.538	0.105	0.137	0.539	1.083	0.422	1.049	0.089	0.096	0.982	1.090	1.079	1.003	0.749
LeWM-Time*	1.014	1.002	1.015	1.024	0.969	0.790	1.018	0.900	1.003	1.013	1.011	1.004	0.986	1.022	1.020
LeWM-Flat*	0.145	0.276	0.077	0.096	0.454	0.927	0.742	0.030	0.041	0.031	0.559	0.534	0.202	0.496	0.846
PLDM*	0.356	0.988	0.010	0.131	0.388	**0.017**	0.813	0.025	**0.005**	0.021	0.570	1.075	**0.074**	0.005	0.690
MotionJEPA (Ours)	**0.038**	**0.084**	0.010	**0.023**	**0.174**	0.416	**0.242**	0.080	0.020	0.011	**0.407**	**0.337**	0.422	0.195	0.746

(a) Offline Linear Probe NMSE↓ of Image Embedding z

	Pong				Dino						Golf				
Method	avg.	ball	bats	score	avg.	cactus	clouds	dino	hearts	sky	avg.	bars	cross	ball	wood
ForwardOnly	1.000	1.002	1.001	0.997	0.999	1.001	1.002	1.000	1.003	0.979	1.002	1.002	1.001	1.000	1.004
Random	0.877	0.913	0.837	0.891	0.808	1.024	0.937	0.928	0.607	0.093	0.988	1.027	0.986	1.008	0.910
SMWM	0.638	1.015	**0.000**	1.014	0.721	0.907	1.003	0.188	0.950	0.134	0.749	1.029	1.021	**0.005**	1.035
LeWM	0.706	1.091	1.083	**0.026**	0.564	1.065	0.503	1.049	0.038	0.055	0.981	1.086	1.069	1.022	0.733
LeWM-Detached	0.418	0.778	0.436	0.129	0.764	1.065	0.861	1.046	0.170	0.215	1.005	1.048	0.894	1.025	0.973
LeWM-Time	0.996	1.000	0.989	1.000	0.994	1.000	1.001	1.000	1.001	0.944	1.000	1.000	1.000	0.999	1.001
LeWM-Flat	0.365	1.084	0.138	0.058	0.477	0.978	0.772	0.069	0.030	0.018	0.961	1.085	0.668	1.071	0.795
PLDM	0.763	1.103	0.912	0.351	0.764	1.063	0.831	1.043	0.486	0.021	0.826	1.099	0.269	0.927	**0.577**
SMWM*	0.653	1.037	0.001	1.039	0.721	0.944	1.002	0.160	0.981	0.122	0.749	1.029	1.021	0.005	1.035
LeWM*	0.262	0.623	0.095	0.162	0.492	1.059	0.352	1.047	0.056	0.028	0.977	1.085	1.067	0.984	0.764
LeWM-Detached*	0.274	0.682	0.110	0.137	0.526	1.057	0.400	1.043	0.104	0.085	0.984	1.084	1.066	1.020	0.751
LeWM-Time*	1.003	0.959	1.022	1.017	0.947	0.786	1.009	0.832	0.984	1.003	1.011	1.011	1.010	1.010	1.013
LeWM-Flat*	0.110	0.294	0.027	0.058	0.471	0.987	0.742	0.086	0.044	0.038	0.636	0.691	0.387	0.492	0.838
PLDM*	0.348	1.080	0.024	0.132	0.397	**0.085**	0.814	**0.035**	**0.007**	0.023	0.588	1.079	**0.214**	0.011	0.687
MotionJEPA (Ours)	**0.069**	**0.208**	0.010	0.026	**0.191**	0.492	**0.239**	0.130	0.024	**0.012**	**0.465**	**0.444**	0.525	0.217	0.745

(b) Offline Linear Probe NMSE↓ of Predicted Rollout $\hat{z}$

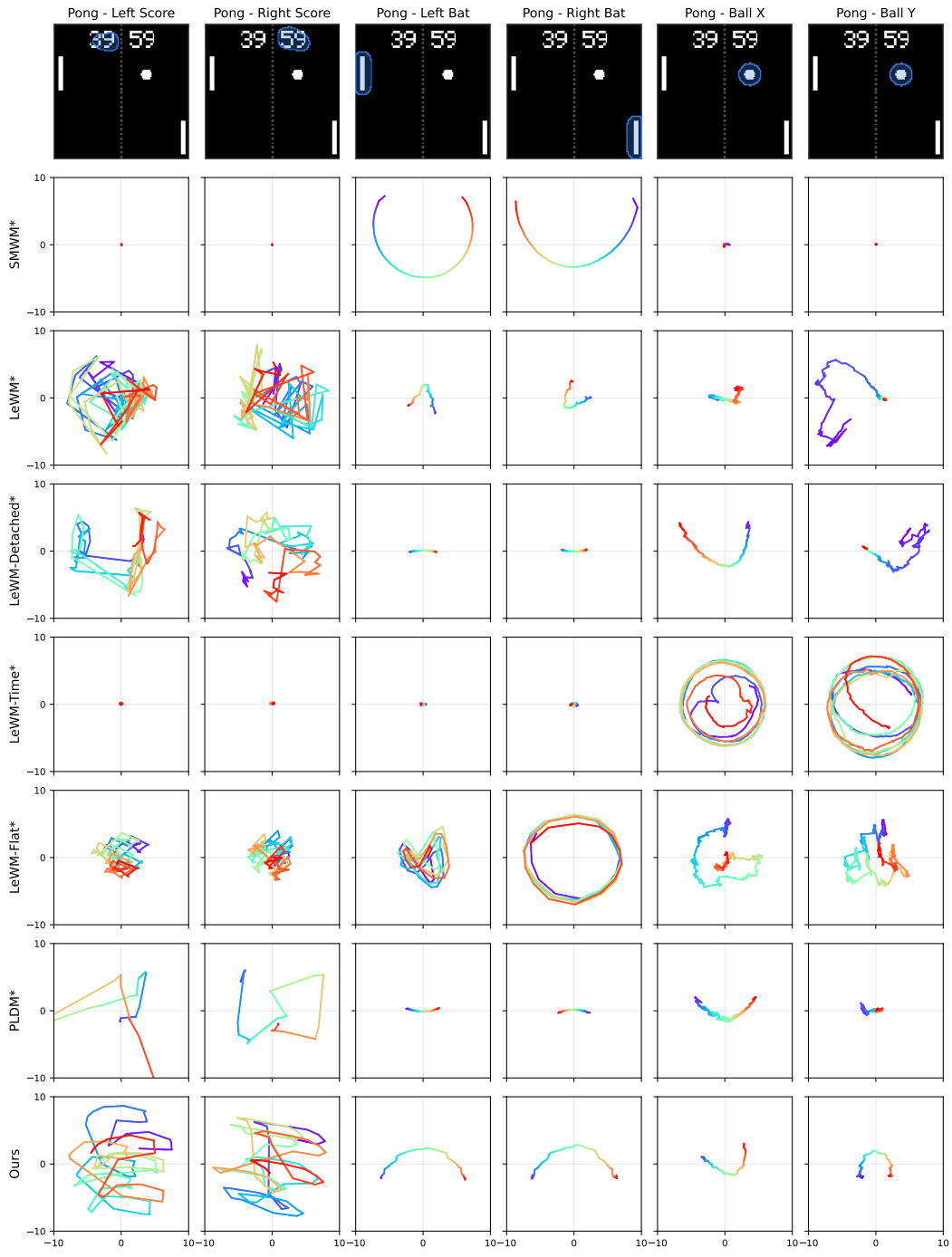


Figure 14: **Per-Feature Latent Trajectories Visualization for Pong**.

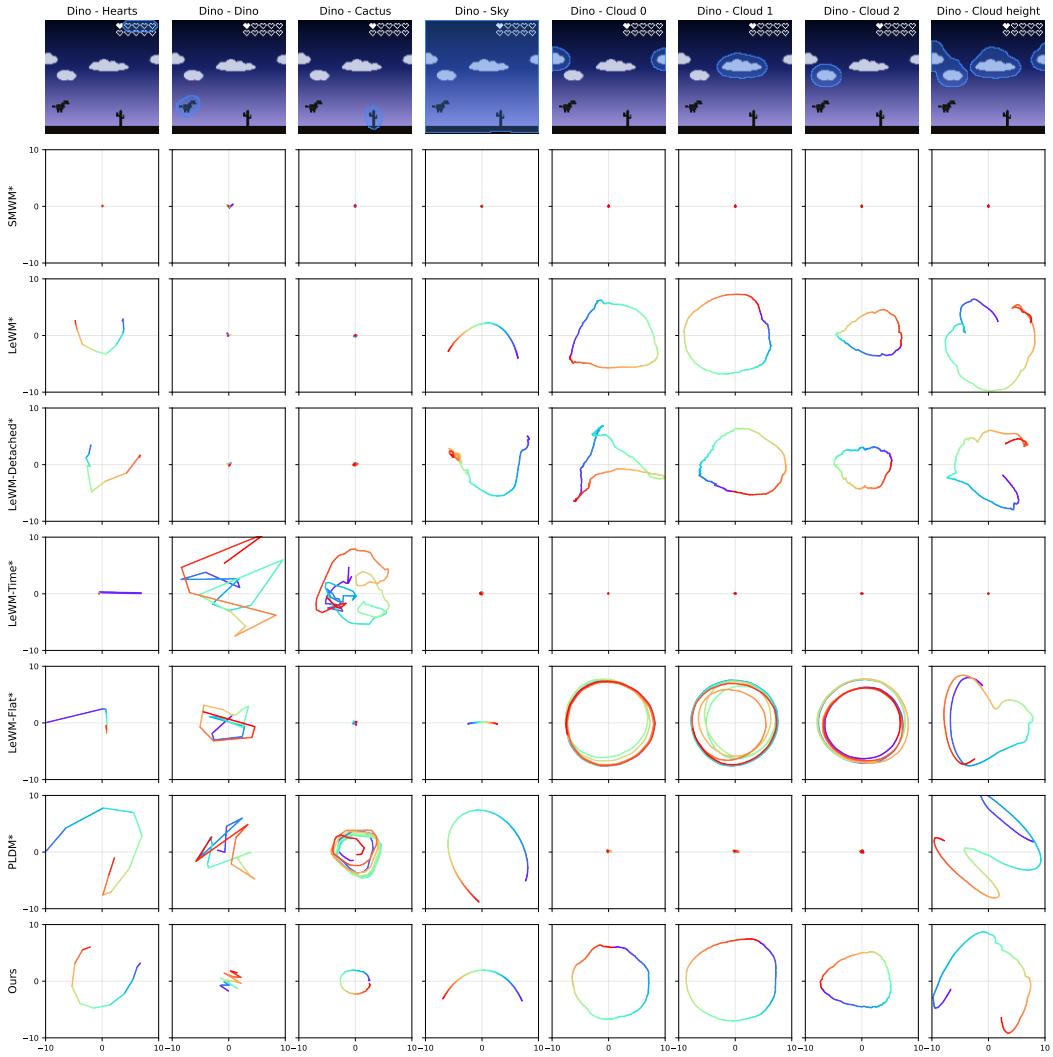


Figure 15: **Per-Feature Latent Trajectories Visualization for Dino**.

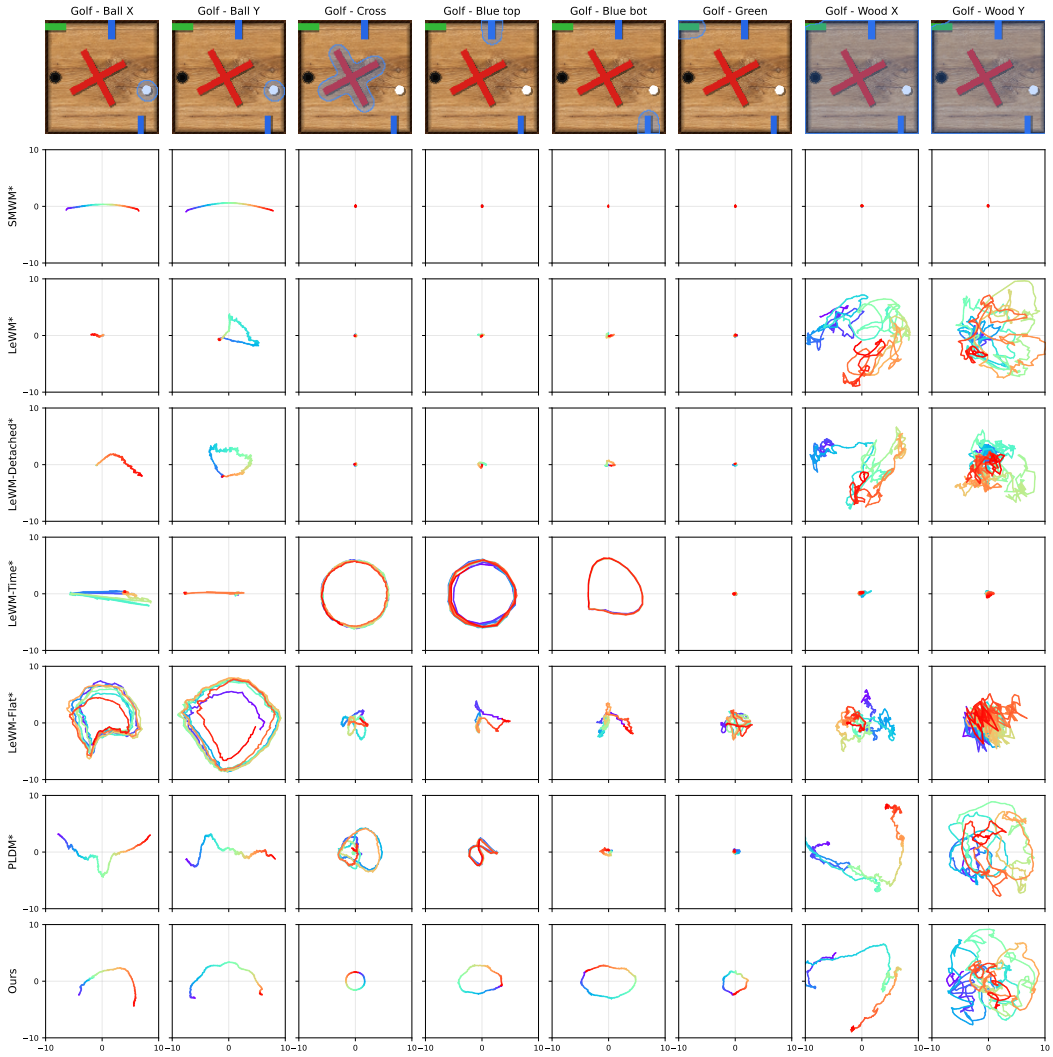


Figure 16: **Per-Feature Latent Trajectories Visualization for Golf**.

Table 9: **Pong Standard Deviation of Three Seed Runs**.

Method	Pong avg.	ball	bats	score
ForwardOnly	1.058 ± 0.099	1.260 ± 0.449	1.001 ± 0.001	0.930 ± 0.121
Random	0.116 ± 0.009	0.143 ± 0.014	0.093 ± 0.011	0.115 ± 0.005
SMWM	0.657 ± 0.004	1.001 ± 0.001	0.002 ± 0.000	1.002 ± 0.001
LeWM	0.658 ± 0.003	0.999 ± 0.004	1.002 ± 0.000	$\mathbf{0.003 \pm 0.001}$
LeWM-Detached	0.171 ± 0.092	0.174 ± 0.013	0.331 ± 0.255	0.007 ± 0.000
LeWM-Time	1.077 ± 0.096	1.100 ± 0.145	1.071 ± 0.096	1.063 ± 0.066
LeWM-Flat	0.284 ± 0.006	0.890 ± 0.003	0.014 ± 0.011	0.005 ± 0.000
PLDM	0.672 ± 0.015	0.973 ± 0.048	0.835 ± 0.289	0.233 ± 0.233
SMWM*	0.651 ± 0.007	0.983 ± 0.029	$\mathbf{0.002 \pm 0.001}$	1.003 ± 0.001
LeWM*	0.061 ± 0.008	0.163 ± 0.025	0.018 ± 0.002	0.012 ± 0.000
LeWM-Detached*	0.048 ± 0.003	0.103 ± 0.008	0.038 ± 0.002	0.008 ± 0.001
LeWM-Time*	0.687 ± 0.006	0.022 ± 0.004	0.972 ± 0.017	1.002 ± 0.001
LeWM-Flat*	0.008 ± 0.001	0.012 ± 0.001	0.004 ± 0.001	0.008 ± 0.001
PLDM*	0.193 ± 0.025	0.571 ± 0.116	0.006 ± 0.001	0.038 ± 0.034
MotionJEPA (Ours)	$\mathbf{0.004 \pm 0.000}$	$\mathbf{0.005 \pm 0.001}$	0.003 ± 0.001	0.004 ± 0.001

(a) Offline Residual MLP Probe NMSE↓ of Image Embedding z

Method	Pong avg.	ball	bats	score
ForwardOnly	1.010 ± 0.015	1.041 ± 0.068	1.006 ± 0.008	0.992 ± 0.017
Random	0.554 ± 0.022	0.721 ± 0.031	0.478 ± 0.028	0.507 ± 0.019
SMWM	0.630 ± 0.003	1.001 ± 0.001	0.002 ± 0.000	1.002 ± 0.001
LeWM	0.644 ± 0.003	1.002 ± 0.000	1.002 ± 0.001	$\mathbf{0.002 \pm 0.000}$
LeWM-Detached	0.239 ± 0.095	0.397 ± 0.005	0.347 ± 0.258	0.007 ± 0.001
LeWM-Time	1.001 ± 0.008	1.003 ± 0.002	0.998 ± 0.019	1.002 ± 0.002
LeWM-Flat	0.278 ± 0.013	0.968 ± 0.017	0.039 ± 0.035	0.005 ± 0.000
PLDM	0.673 ± 0.044	1.002 ± 0.000	0.835 ± 0.289	0.254 ± 0.227
SMWM*	0.630 ± 0.002	1.001 ± 0.001	$\mathbf{0.001 \pm 0.000}$	1.002 ± 0.001
LeWM*	0.138 ± 0.005	0.471 ± 0.019	0.020 ± 0.000	0.009 ± 0.001
LeWM-Detached*	0.129 ± 0.011	0.395 ± 0.037	0.053 ± 0.004	0.008 ± 0.001
LeWM-Time*	0.868 ± 0.028	0.507 ± 0.101	1.002 ± 0.001	1.002 ± 0.001
LeWM-Flat*	0.054 ± 0.003	0.183 ± 0.016	0.004 ± 0.001	0.007 ± 0.002
PLDM*	0.252 ± 0.013	0.861 ± 0.056	0.008 ± 0.001	0.046 ± 0.046
MotionJEPA (Ours)	$\mathbf{0.036 \pm 0.001}$	$\mathbf{0.122 \pm 0.003}$	0.004 ± 0.000	0.005 ± 0.001

(b) Offline Residual MLP Probe NMSE↓ of Predicted Rollout $\hat{z}$

Table 10: **Dino Standard Deviation of Three Seed Runs**.

Method	Dino avg.	cactus	clouds	dino	hearts	sky
ForwardOnly	0.949 ± 0.054	1.003 ± 0.004	1.002 ± 0.010	1.002 ± 0.003	1.031 ± 0.045	0.511 ± 0.422
Random	0.361 ± 0.006	0.833 ± 0.010	0.535 ± 0.016	0.116 ± 0.005	0.031 ± 0.002	0.004 ± 0.002
SMWM	0.649 ± 0.040	0.434 ± 0.213	1.075 ± 0.061	0.028 ± 0.028	0.787 ± 0.061	0.071 ± 0.057
LeWM	0.394 ± 0.032	0.769 ± 0.304	0.248 ± 0.073	0.983 ± 0.033	0.012 ± 0.002	0.005 ± 0.002
LeWM-Detached	0.645 ± 0.020	1.002 ± 0.002	0.727 ± 0.043	1.002 ± 0.002	0.017 ± 0.007	0.006 ± 0.002
LeWM-Time	0.913 ± 0.019	1.003 ± 0.001	0.992 ± 0.013	1.002 ± 0.002	1.029 ± 0.034	0.248 ± 0.130
LeWM-Flat	0.331 ± 0.008	0.522 ± 0.073	0.593 ± 0.026	0.002 ± 0.000	0.006 ± 0.001	0.007 ± 0.001
PLDM	0.711 ± 0.064	0.978 ± 0.037	0.783 ± 0.002	0.995 ± 0.010	0.400 ± 0.525	0.006 ± 0.003
SMWM*	0.606 ± 0.042	0.398 ± 0.047	1.005 ± 0.038	0.003 ± 0.000	0.775 ± 0.177	0.069 ± 0.080
LeWM*	0.282 ± 0.013	0.990 ± 0.008	0.059 ± 0.006	0.751 ± 0.072	0.011 ± 0.002	0.012 ± 0.002
LeWM-Detached*	0.332 ± 0.004	1.005 ± 0.002	0.073 ± 0.003	0.959 ± 0.011	0.011 ± 0.004	0.017 ± 0.002
LeWM-Time*	0.697 ± 0.002	0.005 ± 0.001	1.004 ± 0.002	0.003 ± 0.001	0.971 ± 0.007	1.004 ± 0.002
LeWM-Flat*	0.301 ± 0.008	0.650 ± 0.128	0.498 ± 0.033	0.002 ± 0.000	0.007 ± 0.001	0.016 ± 0.003
PLDM*	0.353 ± 0.002	**0.004 ± 0.001**	0.755 ± 0.001	**0.001 ± 0.000**	**0.003 ± 0.001**	**0.003 ± 0.001**
MotionJEPA (Ours)	**0.021 ± 0.002**	0.011 ± 0.002	**0.037 ± 0.003**	0.006 ± 0.003	0.006 ± 0.002	0.004 ± 0.001

(a) Offline Residual MLP Probe NMSE↓ of Image Embedding z

Method	Dino avg.	cactus	clouds	dino	hearts	sky
ForwardOnly	1.081 ± 0.067	1.019 ± 0.016	1.006 ± 0.005	1.004 ± 0.004	1.017 ± 0.024	1.634 ± 0.537
Random	0.617 ± 0.032	0.974 ± 0.008	0.764 ± 0.008	0.631 ± 0.042	0.279 ± 0.212	0.026 ± 0.012
SMWM	0.686 ± 0.046	0.839 ± 0.139	1.002 ± 0.003	0.092 ± 0.033	0.933 ± 0.186	0.076 ± 0.064
LeWM	0.407 ± 0.024	0.856 ± 0.143	0.251 ± 0.065	0.995 ± 0.010	0.017 ± 0.003	0.005 ± 0.002
LeWM-Detached	0.646 ± 0.016	0.988 ± 0.002	0.732 ± 0.035	1.002 ± 0.001	0.028 ± 0.007	0.006 ± 0.002
LeWM-Time	1.106 ± 0.103	1.011 ± 0.009	1.009 ± 0.005	1.003 ± 0.002	1.026 ± 0.036	1.845 ± 0.860
LeWM-Flat	0.396 ± 0.009	0.877 ± 0.055	0.633 ± 0.028	0.055 ± 0.006	0.009 ± 0.001	0.007 ± 0.001
PLDM	0.716 ± 0.061	0.992 ± 0.009	0.784 ± 0.002	1.001 ± 0.001	0.417 ± 0.512	0.005 ± 0.003
SMWM*	0.671 ± 0.015	0.801 ± 0.022	0.999 ± 0.005	0.075 ± 0.013	0.878 ± 0.081	0.079 ± 0.072
LeWM*	0.334 ± 0.003	0.991 ± 0.002	0.060 ± 0.006	1.003 ± 0.001	0.018 ± 0.001	0.012 ± 0.002
LeWM-Detached*	0.343 ± 0.004	0.988 ± 0.005	0.078 ± 0.008	1.002 ± 0.002	0.020 ± 0.003	0.014 ± 0.003
LeWM-Time*	0.710 ± 0.001	0.120 ± 0.002	1.004 ± 0.002	0.020 ± 0.000	0.959 ± 0.009	1.003 ± 0.002
LeWM-Flat*	0.353 ± 0.021	0.923 ± 0.027	0.524 ± 0.056	0.066 ± 0.007	0.013 ± 0.003	0.018 ± 0.004
PLDM*	0.361 ± 0.001	**0.065 ± 0.007**	0.756 ± 0.001	**0.011 ± 0.001**	**0.004 ± 0.001**	**0.005 ± 0.001**
MotionJEPA (Ours)	**0.055 ± 0.001**	0.243 ± 0.012	**0.040 ± 0.001**	0.051 ± 0.004	0.007 ± 0.001	0.005 ± 0.001

(b) Offline Residual MLP Probe NMSE↓ of Predicted Rollout $\hat{z}$

Table 11: **Golf Standard Deviation of Three Seed Runs**.

Method	avg.	Golf bars	Golf cross	Golf ball	Golf wood
ForwardOnly	1.980 ± 0.769	1.233 ± 0.200	1.558 ± 0.866	1.077 ± 0.108	4.324 ± 2.622
Random	0.776 ± 0.006	0.857 ± 0.011	0.296 ± 0.018	0.989 ± 0.002	0.653 ± 0.002
SMWM	0.727 ± 0.002	1.000 ± 0.001	0.986 ± 0.001	**0.002 ± 0.001**	1.003 ± 0.001
LeWM	0.763 ± 0.009	1.002 ± 0.001	1.001 ± 0.000	0.942 ± 0.061	0.084 ± 0.033
LeWM-Detached	0.811 ± 0.100	0.924 ± 0.074	0.683 ± 0.532	0.930 ± 0.037	0.572 ± 0.126
LeWM-Time	1.053 ± 0.076	1.008 ± 0.005	1.014 ± 0.025	1.004 ± 0.002	1.195 ± 0.302
LeWM-Flat	0.694 ± 0.066	1.002 ± 0.001	0.379 ± 0.539	0.993 ± 0.013	0.052 ± 0.025
PLDM	0.590 ± 0.016	1.000 ± 0.001	0.033 ± 0.005	0.781 ± 0.064	0.032 ± 0.009
SMWM*	0.727 ± 0.002	1.000 ± 0.001	0.986 ± 0.001	0.002 ± 0.001	1.003 ± 0.001
LeWM*	0.760 ± 0.007	1.001 ± 0.001	1.002 ± 0.001	0.909 ± 0.084	0.110 ± 0.063
LeWM-Detached*	0.745 ± 0.008	1.002 ± 0.001	1.002 ± 0.001	0.908 ± 0.022	0.048 ± 0.015
LeWM-Time*	0.985 ± 0.005	0.972 ± 0.013	0.992 ± 0.015	0.984 ± 0.002	1.002 ± 0.001
LeWM-Flat*	0.299 ± 0.027	0.317 ± 0.058	0.059 ± 0.018	0.312 ± 0.066	0.377 ± 0.044
PLDM*	0.373 ± 0.001	0.997 ± 0.001	**0.026 ± 0.005**	0.004 ± 0.001	**0.021 ± 0.002**
MotionJEPA (Ours)	**0.061 ± 0.012**	**0.095 ± 0.012**	0.083 ± 0.004	0.033 ± 0.023	0.033 ± 0.008

(a) Offline Residual MLP Probe NMSE↓ of Image Embedding z

Method	avg.	Golf bars	Golf cross	Golf ball	Golf wood
ForwardOnly	1.003 ± 0.003	1.002 ± 0.001	1.001 ± 0.001	1.005 ± 0.008	1.005 ± 0.004
Random	0.934 ± 0.004	0.995 ± 0.001	0.913 ± 0.007	0.998 ± 0.002	0.781 ± 0.014
SMWM	0.729 ± 0.002	1.001 ± 0.000	1.001 ± 0.000	**0.006 ± 0.001**	1.002 ± 0.001
LeWM	0.767 ± 0.011	1.002 ± 0.000	1.001 ± 0.001	0.947 ± 0.063	0.094 ± 0.034
LeWM-Detached	0.855 ± 0.054	0.967 ± 0.038	0.821 ± 0.313	0.930 ± 0.045	0.618 ± 0.102
LeWM-Time	1.003 ± 0.004	1.006 ± 0.005	1.006 ± 0.008	1.000 ± 0.002	1.002 ± 0.002
LeWM-Flat	0.722 ± 0.045	1.002 ± 0.001	0.590 ± 0.356	0.994 ± 0.014	0.061 ± 0.030
PLDM	0.616 ± 0.016	1.002 ± 0.001	0.208 ± 0.013	0.795 ± 0.057	0.036 ± 0.012
SMWM*	0.729 ± 0.002	1.001 ± 0.000	1.001 ± 0.000	0.006 ± 0.001	1.002 ± 0.001
LeWM*	0.763 ± 0.007	1.002 ± 0.001	1.002 ± 0.001	0.913 ± 0.080	0.117 ± 0.063
LeWM-Detached*	0.746 ± 0.008	1.002 ± 0.001	1.001 ± 0.001	0.909 ± 0.020	0.051 ± 0.017
LeWM-Time*	0.997 ± 0.000	0.997 ± 0.002	0.995 ± 0.010	0.991 ± 0.001	1.002 ± 0.002
LeWM-Flat*	0.448 ± 0.030	0.540 ± 0.048	0.320 ± 0.044	0.402 ± 0.062	0.425 ± 0.048
PLDM*	0.393 ± 0.002	1.001 ± 0.002	**0.184 ± 0.028**	0.007 ± 0.000	**0.021 ± 0.004**
MotionJEPA (Ours)	**0.164 ± 0.025**	**0.262 ± 0.029**	0.348 ± 0.005	0.065 ± 0.049	0.037 ± 0.006

(b) Offline Residual MLP Probe NMSE↓ of Predicted Rollout $\hat{z}$