TS-DREAMER: ALIGNING REPRESENTATIONS IN WORLD MODELS AT DIFFERENT TEMPORAL SCALES

Anonymous authors
Paper under double-blind review

ABSTRACT

World models let agents learn from imagined trajectories, so their latent states must preserve information for prediction and control. Aligning states with image embeddings provides supervision without pixel reconstruction. Single-step alignment directly supervises correspondence with the current image, but not explicitly over multiple observations, where motion and other temporal properties can become apparent. Different observation intervals can reveal longer-term patterns. We ask whether alignment at different temporal scales can provide complementary supervision. TS-Dreamer maps latent states into the image embedding space and jointly aligns the two representations at individual steps and over temporal windows of different lengths. At each scale, normalized sums over matching windows are aligned using Barlow Twins, without adding trainable parameters or changing inference. TS-Dreamer leads the evaluated methods in mean and median returns on DeepMind Control Suite and remains competitive on Atari 100K. In FlappyBird, adding alignment windows up to 32 decisions yields $3.8\times$ the two-scale score at fixed capacity and training budget. Representation analyses show improved linear readout of physical quantities and their changes. These results highlight temporal scale as a design choice for world model supervision.

1 INTRODUCTION

World models learn to predict how an environment evolves in response to an agent's actions (Ha & Schmidhuber, 2018). In model-based reinforcement learning, these predictions allow agents to learn behavior from imagined experience and make better use of interactions with the environment (Hafner et al., 2020; 2025). For visual control, a recurrent world model summarizes observations and past actions in a latent state used for prediction and decision making. Both depend on what information this state preserves. A central question is therefore how to supervise the learning of these representations.

Dreamer-style world models have explored different sources of representation supervision. DreamerV3 reconstructs observations from latent states, providing supervision at the pixel level that can also encourage the retention of visual details unrelated to control (Hafner et al., 2025). Several approaches instead use representations of observations as targets, without reconstructing images. DreamerPro jointly learns prototypes from observations and recurrent states (Deng et al., 2022). R2-Dreamer projects latent states into the image embedding space and aligns the two representations using Barlow Twins (BT) (Morihira et al., 2026; Zbontar et al., 2021). Image embeddings thus provide targets that constrain what the states preserve without requiring pixel reconstruction.

In this formulation, each state projection is paired with the current image embedding, although the state summarizes a history of observations. The alignment directly constrains correspondence at the current step, but does not explicitly compare image and state representations over groups of observations. Such groups can expose information needed for control, including motion, that a single image need not determine. Moreover, different observation intervals can reveal different aspects of the same process, from local changes to longer-term patterns. For example, a sequence of Moon images can be described by the phase in each frame, changes between nearby frames, or waxing over a longer interval (Figure 1). These descriptions suggest that alignment over different intervals may provide complementary supervision. We therefore ask: **Can world model learning benefit from jointly aligning image and state representations at different temporal scales?**

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

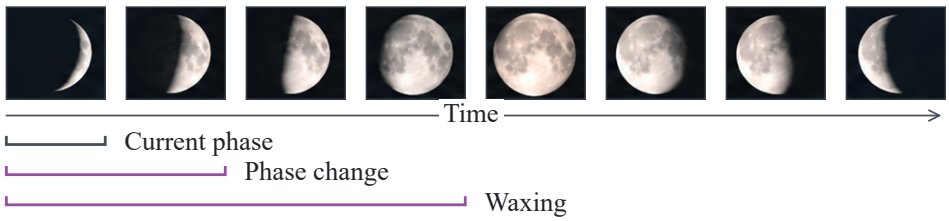


Figure 1: Schematic of temporal context at different scales. A single observation shows the current Moon phase; wider windows provide context for phase change and waxing.

We introduce **TS-Dreamer**, a reconstruction-free world model that retains single-step alignment and adds alignment over temporal windows. Here, a *temporal scale* is the length of a contiguous alignment window, measured in agent decisions. At each selected scale, image embeddings and state projections are aggregated over matching windows using normalized sums and aligned with Barlow Twins. All scales jointly supervise the same model. Alignment therefore also depends on how image and state representations relate across steps within each window (Section 3.2). This adds no trainable parameters and leaves the recurrent architecture and policy inference unchanged. We evaluate continuous and discrete control on DeepMind Control Suite (DMC) and Atari 100K, and use FlappyBird to study temporal scales at fixed model capacity and training budget. Representation analyses on Walker and a controlled ordered-motion task examine what temporal information is accessible in the learned states.

Our main contributions are:

- We study temporal scales as a design choice in world model representation learning, and introduce TS-Dreamer to jointly align image and state representations in a single recurrent model.
- TS-Dreamer achieves the highest mean and median returns among the evaluated methods on DMC and remains competitive on Atari 100K. In FlappyBird, alignment at one- and two-step scales improves the mean score over R2-Dreamer by 17.3%. Keeping these scales and adding windows of 4, 8, 16, and 32 decisions yields $3.8\times$ the two-scale score at fixed model capacity and training budget.
- Analyses on Walker show stronger image–state correspondence across temporal windows and improved linear readout of physical quantities and their changes. A controlled motion task shows that motion order is easier to identify from state sequences when model weights and the classifier training budget are held fixed.

2 Related Work

Representation learning in world models. Training objectives shape the information retained in world model representations. PlaNet and the Dreamer family learn representations through image reconstruction and reward prediction (Hafner et al., 2019; 2020; 2021; 2025). Reconstruction provides supervision at the pixel level and can also encourage the model to preserve visual details unrelated to control. HarmonyDream learns weights for the reconstruction, reward, and dynamics losses to balance their contributions during training (Ma et al., 2024). Several methods learn representations without reconstruction. DreamerPro jointly learns prototypes from observations and recurrent states (Deng et al., 2022), while MuDreamer uses reward, continuation, value, and action prediction (Burchi & Timofte, 2024). TD-MPC2 combines latent state prediction, reward prediction, and value learning to train representations for planning without an observation decoder (Hansen et al., 2024). Reward and transition structure also guide state abstraction. DeepMDP learns representations by predicting rewards and transition distributions in latent space (Gelada et al., 2019). Methods based on bisimulation use differences in rewards and transition distributions to constrain distances between state representations (Zhang et al., 2021). R2-Dreamer projects latent states into the image embedding space and aligns the two representations using Barlow Twins (Morihira et al., 2026; Zbontar et al., 2021). We study temporal scales by jointly aligning image and state representations at individual steps and over matching windows.

Representations across temporal scales. Dawid & LeCun (2024) describe a hierarchical joint-embedding predictive architecture (JEPA) that uses detailed representations for predictions over short intervals and more abstract representations for longer intervals. Clockwork Variational Autoencoders organize latent variables into a hierarchy whose higher levels update less frequently (Saxena et al., 2021). Hieros trains a separate world model and policy at each level of its hierarchy. Higher levels receive sequences of latent states from the level below and set subgoals for the lower policies (Mattes et al., 2024). Director trains two policies using a shared world model: a manager selects goals in latent space, and a worker chooses actions to reach them (Hafner et al., 2022). TS-Dreamer instead jointly supervises the same representations at different temporal scales within a single recurrent world model. The world model and policy architectures remain unchanged.

Temporal representation objectives. Temporal objectives use relationships among observations to encourage representations to capture environmental dynamics. Contrastive Predictive Coding and Temporal Predictive Coding learn to predict future representations from past context (Oord et al., 2018; Nguyen et al., 2021). In reinforcement learning, the self-predictive representations (SPR) method predicts target representations several steps ahead (Schwarzer et al., 2021), and EfficientZero adds a consistency loss between predicted latent states and encodings of future observations (Ye et al., 2021). Dreamer-CDP predicts continuous observation embeddings from recurrent context (Hauri & Zenke, 2026). NE-Dreamer uses a temporal Transformer to predict the next encoder embedding from latent state sequences (Bredis et al., 2026). TWISTER trains its Transformer world model to make contrastive predictions at several future offsets, conditioned on intervening actions (Burchi & Timofte, 2025). TC-LeWM uses temporal windows for regularization: it subtracts a local temporal mean from latent representations and applies the sketched isotropic Gaussian regularizer (SIGReg) to the residuals (Liu et al., 2026). TS-Dreamer's additional alignment objective compares image and state representations over matching windows, rather than predicting future representations or regularizing temporal residuals.

3 METHOD

TS-Dreamer learns latent states for prediction and control by jointly aligning image embeddings and state projections at different temporal scales, as illustrated in Figure 2. All selected scales contribute, with window lengths up to the training sequence length.

3.1 LATENT WORLD MODEL

At decision step t, the agent receives an image observation o_t and reward r_t, then selects an action a_t. The continuation flag $c_t \in \{0, 1\}$ is zero at terminal transitions, but not for time-limit truncation alone. Training uses experience sequences stored in a replay buffer, with the goal of maximizing future reward.

We use a reconstruction-free recurrent state-space model (RSSM), following Dreamer's latent dynamics and R2-Dreamer's alignment between image and state representations (Hafner et al., 2019; 2025; Morihira et al., 2026). The RSSM combines deterministic memory h_t and a discrete stochastic state z_t into the latent state $s_t = [\mathrm{vec}(z_t), h_t]$, which is used for prediction and control; vec flattens the stochastic state. With ϕ denoting all world model parameters, the components are

$$\begin{aligned}
\text{Image encoder:} \quad & e_t = f_{\mathrm{enc},\phi}(o_t), \\
\text{Recurrent dynamics:} \quad & h_t = f_{\mathrm{dyn},\phi}(h_{t-1}, z_{t-1}, a_{t-1}), \\
\text{Posterior inference:} \quad & z_t \sim q_\phi(\cdot \mid h_t, e_t), \\
\text{Prior prediction:} \quad & \hat{z}_t \sim \rho_\phi(\cdot \mid h_t), \\
\text{Reward prediction:} \quad & p_{r,\phi}(r_t \mid s_t), \\
\text{Continuation prediction:} \quad & p_{c,\phi}(c_t \mid s_t), \\
\text{State projector:} \quad & p_t = g_\phi(s_t).
\end{aligned} \tag{1}$$

Here $f_{\mathrm{enc},\phi}$, $f_{\mathrm{dyn},\phi}$, and g_ϕ denote the encoder, recurrent dynamics, and projector, respectively. The reward and continuation predictors define $p_{r,\phi}$ and $p_{c,\phi}$. Both e_t and p_t lie in $\mathbb{R}^D$, where D is the embedding dimension.

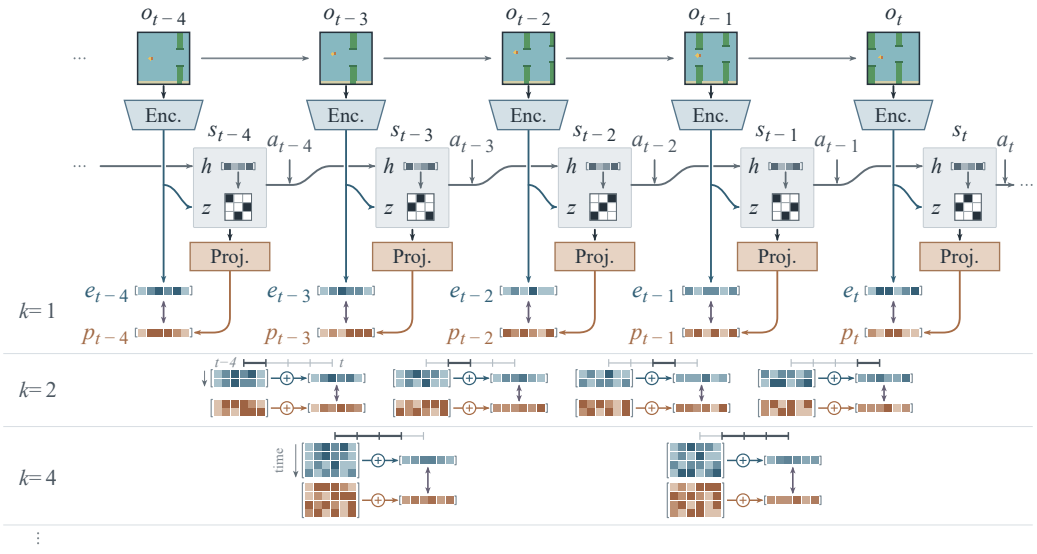


Figure 2: Overview of TS-Dreamer, illustrating alignment over windows of $k = 1, 2, 4$ decision steps. The encoder (Enc.) maps observations to image embeddings e_t. The projector (Proj.) maps states s_t, formed from memory h_t and stochastic state z_t, to p_t. Short timelines show which steps belong to each window. Rows within each window matrix correspond to successive steps, and $\oplus$ aggregates these rows into a single vector by normalized summation (Eq. 2). Purple double arrows denote Barlow Twins alignment between image and state representations at each scale. Losses over valid windows are optimized jointly across scales. The vertical ellipsis indicates additional scales. Observation thumbnails are schematic.

The posterior q_ϕ incorporates the current image into the stochastic state, whereas the prior ρ_ϕ predicts that state from memory alone. Posterior samples z_t represent observed experience; prior samples $\hat{z}_t$ are used when imagining future trajectories. Reward and continuation predictions evaluate imagined trajectories without generating images.

For alignment, e_t represents the current observation, whereas p_t is a projection of the history-dependent latent state. The policy and prediction heads operate on s_t; its projection provides a shared feature dimension for alignment.

3.2 Representations at Different Temporal Scales

Recurrent memory summarizes past observations and actions; the alignment window determines which steps are combined for alignment.

For B training sequences of length T, let $\mathcal{K} \subseteq \{1, \dots, T\}$ contain the window lengths, with $1 \in \mathcal{K}$. Each k counts agent decisions. For a window ending at t, we form

$$e_t^{(k)} = \frac{1}{\sqrt{k}} \sum_{\ell=0}^{k-1} e_{t-\ell}, \qquad p_t^{(k)} = \frac{1}{\sqrt{k}} \sum_{\ell=0}^{k-1} p_{t-\ell}. \tag{2}$$

Here ℓ is a window offset; both summaries remain D-dimensional. The factor $k^{-1/2}$ is a normalization convention and cancels under feature standardization at each scale in Barlow Twins, apart from numerical stabilization (Appendix B). States are projected before aggregation; we use every complete window within an episode, advancing one step at a time.

The additional alignment compares image and state summaries over matching windows, so its covariance includes terms from different time steps. Reordering image embeddings within a window leaves their sum unchanged. The state projections depend on the ordered history processed by the recurrent world model. The alignment objective therefore constrains how recurrent states summarize observations across the window without adding another network to process the window. The $k = 1$ term retains single-step supervision.

Before feature standardization, the window covariance is

$$\mathrm{Cov}\left(p_t^{(k)}, e_t^{(k)}\right) = \frac{1}{k}\sum_{u=0}^{k-1}\sum_{v=0}^{k-1}\mathrm{Cov}(p_{t-u}, e_{t-v})\,, \tag{3}$$

where u and v index offsets within the same sampled windows. At $k = 2$, cross-time covariance terms include $\mathrm{Cov}(p_t, e_{t-1})$ and $\mathrm{Cov}(p_{t-1}, e_t)$. These terms contribute to one alignment statistic; we do not apply a separate loss at each time lag. The factor $1/k$ belongs to this raw covariance and cancels from the standardized correlation.

3.3 Joint Alignment across Temporal Scales

We apply the Barlow Twins objective separately at each scale (Zbontar et al., 2021). Stacking the N_k valid windows at scale k gives $E^{(k)}, P^{(k)} \in \mathbb{R}^{N_k \times D}$, with matching rows corresponding to the same window. Let $\bar{E}^{(k)}$ and $\bar{P}^{(k)}$ denote the corresponding matrices with each feature centered and normalized to unit variance over these windows, using variance denominator N_k separately for images and states at each scale. Their cross-correlation matrix is

$$C^{(k)} = \frac{1}{N_k}(\bar{P}^{(k)})^\top \,\mathrm{sg}[\bar{E}^{(k)}] \in \mathbb{R}^{D\times D}, \tag{4}$$

where sg preserves values but blocks gradients through the image target branch. The scale-specific loss is

$$\mathcal{L}_{\mathrm{BT}}^{(k)} = \sum_{i=1}^{D}(1 - C_{ii}^{(k)})^2 + \lambda_{\mathrm{off}}\sum_{i\neq j}(C_{ij}^{(k)})^2. \tag{5}$$

Here $i, j \in \{1, \ldots, D\}$ index features. The diagonal term promotes correspondence between matching features, while the off-diagonal term penalizes correlations between nonmatching features. The weight $\lambda_{\mathrm{off}} \geq 0$ is shared across scales.

All selected scales jointly contribute to the alignment objective,

$$\mathcal{L}_{\mathrm{align}} = \sum_{k\in\mathcal{K}} w_k \mathcal{L}_{\mathrm{BT}}^{(k)}, \qquad w_k \geq 0, \quad \sum_{k\in\mathcal{K}} w_k = 1. \tag{6}$$

Here w_k weights scale k. The $k = 1$ term retains R2-Dreamer's single-step alignment between image and state representations (Morihira et al., 2026). Normalizing the weights keeps their sum fixed when varying $\mathcal{K}$; window lengths and weights are specified with the experimental settings.

Alignment uses posterior states inferred from stored experience. Gradients flow through $P^{(k)}$ into the projector and RSSM, and reach the image encoder through posterior inference.

3.4 World Model and Policy Learning

Predictive learning. We combine the alignment objective with reward and continuation prediction and latent dynamics regularization. The prediction loss is

$$\mathcal{L}_{\mathrm{pred}} = -\mathbb{E}[\log p_{r,\phi}(r_t \mid s_t) + \log p_{c,\phi}(c_t \mid s_t)]\,, \tag{7}$$

where the expectation averages over replayed time steps and posterior samples. For latent regularization, let q_t and ρ_t denote the posterior and prior categorical distributions before the uniform mixing used for sampling. The two Kullback–Leibler (KL) divergence terms are

$$\begin{aligned}\mathcal{L}_{\mathrm{dyn}} &= \mathbb{E}[\max(\tau_{\mathrm{KL}}, D_{\mathrm{KL}}[\mathrm{sg}[q_t] \,\|\, \rho_t])]\,,\\ \mathcal{L}_{\mathrm{rep}} &= \mathbb{E}[\max(\tau_{\mathrm{KL}}, D_{\mathrm{KL}}[q_t \,\|\, \mathrm{sg}[\rho_t]])]\,.\end{aligned} \tag{8}$$

Here D_{KL} denotes the KL divergence and τ_{KL} is a free-nats threshold below which the KL terms have zero gradient. The dynamics term trains the prior toward the posterior, while the representation term regularizes the posterior toward the prior. In each term, gradients are stopped through the target distribution. The model objective is

$$\mathcal{L}_{\mathrm{model}} = \mathcal{L}_{\mathrm{pred}} + \beta_{\mathrm{dyn}}\mathcal{L}_{\mathrm{dyn}} + \beta_{\mathrm{rep}}\mathcal{L}_{\mathrm{rep}} + \beta_{\mathrm{align}}\mathcal{L}_{\mathrm{align}}, \tag{9}$$

where the nonnegative coefficients β_{dyn}, β_{rep}, and β_{align} weight dynamics regularization, posterior regularization, and alignment.

Learning to act. We train actor $\pi_\theta(a_t \mid s_t)$ and critic $v_\psi(s_t)$ following R2-Dreamer and DreamerV3 (Morihira et al., 2026; Hafner et al., 2025); Appendix B gives the full objectives. In addition to $\mathcal{L}_{\text{model}}$, the replay value loss updates the world model through inferred states; imagined actor and critic losses do not. Temporal aggregation adds no parameters and is used only during training, leaving policy inference unchanged.

4 EXPERIMENTS

Our experiments evaluate control performance, the benefits of broader temporal coverage, and the temporal information accessible in learned states. We also compare alignment objectives and computational costs.

4.1 EXPERIMENTAL SETUP

We evaluate TS-Dreamer on 20 visual tasks from the DeepMind Control Suite (Tassa et al., 2018), the 26-game Atari 100K benchmark (Bellemare et al., 2013; Kaiser et al., 2020), and FlappyBird. DMC and Atari cover continuous-action control and discrete-action games. FlappyBird requires using motion and obstacle positions to time actions and avoid collisions, making it a useful setting for further analysis.

We use R2-Dreamer, DreamerV3, and DreamerPro as baselines (Morihira et al., 2026; Hafner et al., 2025; Deng et al., 2022). Given limited computational resources, we use small model configurations in a shared Dreamer implementation, with matched interaction budgets and evaluation protocols across methods within each task. The main FlappyBird comparison includes TS-Dreamer, R2-Dreamer, and DreamerV3. Figures and tables abbreviate these methods as R2, V3, and Pro, and TS-Dreamer as TS. Table 2 collects the model architecture and training settings. The DMC, Atari, and FlappyBird baseline comparisons use TS-Dreamer with two scales, $\mathcal{K} = \{1, 2\}$, and equal weights, $w_1 = w_2 = 1/2$.

We use three training seeds per task on DMC and Atari, and five for all FlappyBird comparisons. Final scores use the checkpoint at the fixed training budget. Each model is evaluated for 30 episodes on DMC and Atari, or ten on FlappyBird. Evaluation environment seeds or obstacle sequences are shared across methods. Training budgets and curve axes count agent decisions.

We report task scores on DMC and Atari and the number of obstacles passed in FlappyBird. Episode scores are first averaged within each trained model, then scores are averaged across training seeds. Aggregate comparisons weight tasks equally and report their mean and median. We rank the four methods within each task by their seed-averaged final scores, using average ranks for ties. We report the average rank across tasks; lower is better. Aggregate curves include bootstrap confidence intervals (Agarwal et al., 2021). Additional evaluation details and per-task scores are provided in Appendix C.

4.2 PERFORMANCE ON DEEPMIND CONTROL SUITE

Figure 3 shows the mean and median learning curves across all 20 DMC tasks. At 250k agent decisions, TS-Dreamer achieves the highest final mean and median returns among the four methods: 636.68 and 766.80, compared with 630.91 and 732.66 for R2-Dreamer. Its average task rank is also the best at 1.925, compared with 2.125 for R2-Dreamer.

4.3 PERFORMANCE ON ATARI 100K

We evaluate TS-Dreamer on all 26 Atari 100K games using three training seeds per method. In Figure 4, TS-Dreamer achieves the highest final median human-normalized score, 18.04, compared with 17.27 for R2-Dreamer, 9.32 for DreamerV3, and 4.32 for DreamerPro.

TS-Dreamer exceeds DreamerV3 and DreamerPro on 14 and 23 of the 26 games, respectively, and achieves the best average game rank of 2.115. These results show competitive visual control performance across DMC and Atari 100K. Appendix C reports the complete per-game comparison.

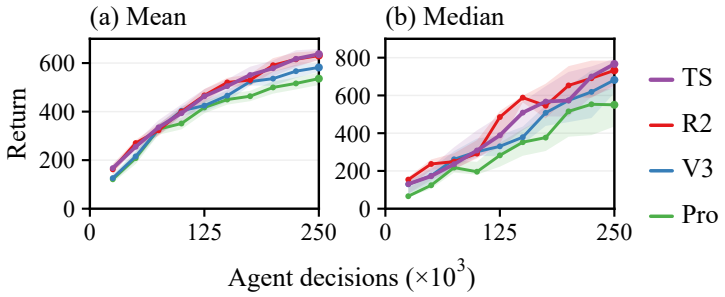


Figure 3: Mean and median returns across all 20 DMC tasks, each averaged over three training seeds. Shading shows pointwise 95% confidence intervals from bootstrapping training seeds within each task. Intermediate checkpoints use five episodes and the final checkpoint uses 30. The horizontal axis counts agent decisions; 250k decisions correspond to 500k environment frames with action repeat 2.

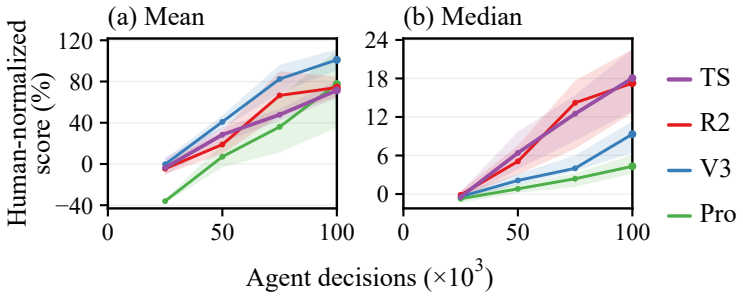


Figure 4: Mean and median human-normalized scores on all 26 Atari 100K games, each averaged over three training seeds. Shading shows pointwise 95% confidence intervals from bootstrapping training seeds within each game. Intermediate checkpoints use five episodes and the final checkpoint uses 30. The horizontal axis counts agent decisions; 100k decisions correspond to 400k environment frames with action repeat 4.

4.4 A Case Study in FlappyBird

In FlappyBird, we study baseline performance, temporal coverage, and three alignment objectives: BT, variance-invariance-covariance regularization (VICReg), and linear centered kernel alignment (CKA).

Two-scale configuration. At 110k decisions, TS-Dreamer with $\mathcal{K} = \{1, 2\}$ passes 17.3% more pipes than R2-Dreamer and 21.3% more than DreamerV3, averaged over five training seeds and ten evaluation episodes per model. Table 1 reports the scores; Appendix C.3 details the protocol.

Temporal coverage. With the same five seeds, we add windows of 4, 8, 16, and 32 decisions, holding model capacity, training budget, and total alignment weight fixed. Each scale is aligned separately, retaining $w_1 = 1/2$. Figure 5 shows gains from broader temporal coverage that depend on the scale configuration. With a longest window of $K_{\max} = 32$ decisions, the mean reaches 3621.1 pipes, 3.8× the two-scale score. Additional scale ablations on DMC and Atari tasks are reported in Appendix C.4.

Alignment objective. At $\mathcal{K} = \{1, 2\}$, Table 1 compares BT, VICReg, and linear CKA. We replace either the two-step objective alone or both objectives. BT achieves the highest mean score in both comparisons.

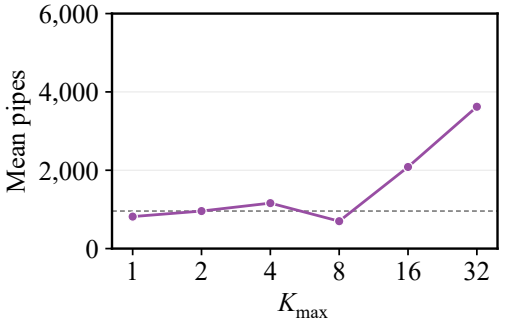


Figure 5: FlappyBird temporal coverage: $\mathcal{K} = \{1, 2, 4, \ldots, K_{\max}\}$. Points show means over five training seeds; each scale is aligned separately. $K_{\max}$ counts decisions in the longest window. The dashed line marks $\mathcal{K} = \{1, 2\}$.

Table 1: FlappyBird mean pipes at 110k decisions with ten shared schedules per model. Baseline and objective comparisons use five training seeds, as in Figure 5.

Baseline comparison

Method	Mean pipes
DreamerV3	790.2
R2-Dreamer	816.7
TS-Dreamer	**958.2**

Alignment objective

Single step	Two-step window	Mean pipes
Barlow Twins	Barlow Twins	**958.2**
Barlow Twins	VICReg	754.5
VICReg	VICReg	367.7
Barlow Twins	Linear CKA	367.0
Linear CKA	Linear CKA	261.1

BT compares the cross-correlation matrix of standardized features with the identity (Zbontar et al., 2021). VICReg combines squared differences between matching features with variance and covariance regularization (Bardes et al., 2022); linear CKA compares inner products between centered samples, regardless of scale or feature rotations (Kornblith et al., 2019). BT thus aligns corresponding features at each scale. Appendix C provides objective definitions and comparisons.

4.5 Representation Analysis

All analyzed TS-Dreamer models use $\mathcal{K} = \{1, 2\}$. Figure 6a shows correlations between image embeddings and state projections on shared held-out Walker trajectories at $k = 2$. TS also has higher mean diagonal correlations at four- and eight-step windows (Appendix C.5). Panels b and c report linear readout of physical quantities over temporal windows from latent states s and projections p.

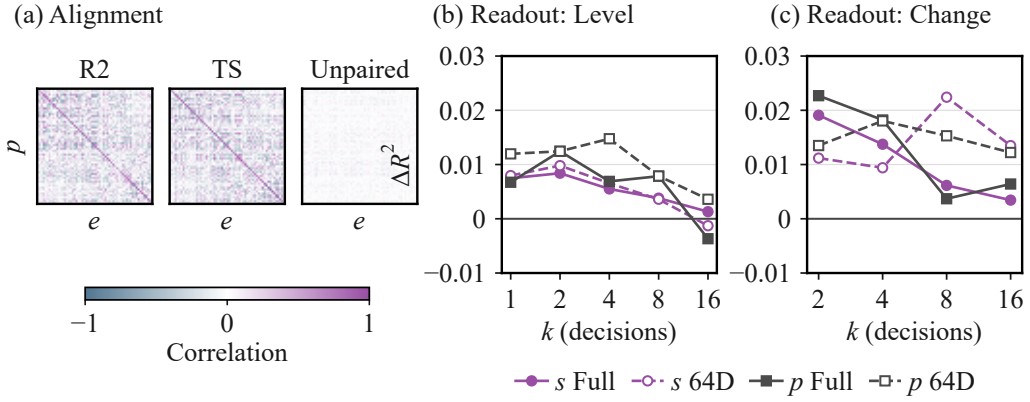


Figure 6: **Walker alignment and readout.** (a) $C^{(2)}$ for R2, TS, and unpaired TS trajectories, using the same 64 sampled features from seed 41. Rows index p and columns e. (b) Level and (c) Change readout gains, ΔR^2 (TS minus R2), against window length k in decisions. Full uses all features; 64D projects them to 64 dimensions. Correlation curves and protocols: Appendix C.5.

Physical readout. Linear readout models predict Walker height, body orientations, and velocities. We apply the same window transform to representations and physical targets, separating the overall level within a window from change between its two halves. *Full* uses all features; *64D* uses a fixed random projection to 64 dimensions for comparison at equal input size. Gains appear in both level and change components of the latent states, including after compression to 64 dimensions.

With 64-dimensional inputs, TS yields higher mean readout scores in eight of nine combinations of window length and component for full states, versus one for image embeddings. The gains are thus more consistent in recurrent states than in the per-image features used as alignment targets (Appendix C.5).

Ordered motion. Figure 7 compares continuous rotation around a fixed anchor with reversal after each lap. Paired videos contain the same frames and endings, so recognizing the motion requires temporal order. We train compact world models without motion labels, keep their weights fixed, and train a temporal convolutional classifier on their complete RSSM state sequences, using a fixed projection to 64 features per step. After 400 classifier updates, TS-Dreamer reaches 80.7% accuracy versus R2-Dreamer's 71.9%, retaining its advantage under changes in starting angle and rotation direction. Shuffling the order of encoded states reduces accuracy to 52.7% and 50.1%, respectively, near chance. The motion-readout advantage is observed in the full state sequences; the controls using state projections and image embeddings remain near chance (about 54%) under the tested readout protocol. Appendix C.6 gives further controls and protocol details.

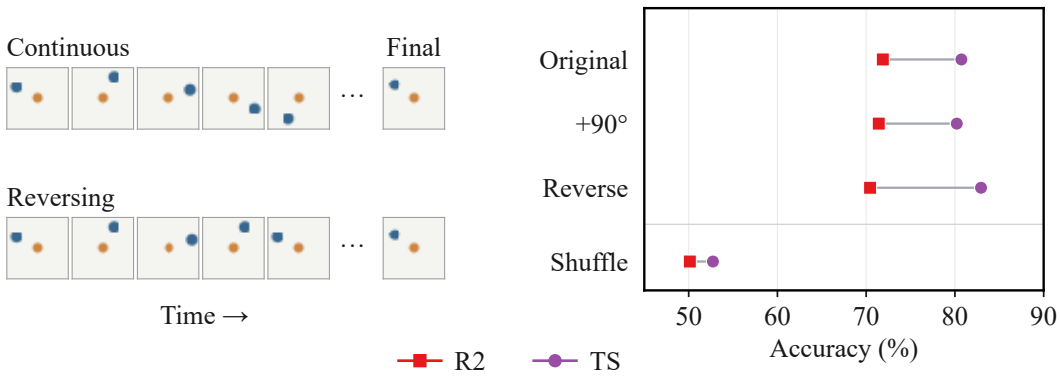


Figure 7: **Recognizing motion.** Left: continuous and reversing trajectories contain the same frames and end identically. Right: accuracy on original videos, starting angles shifted by $+90°$, reversed rotation for both classes, and shuffled encoded states. Results are averaged over two world model seeds and three classifier initializations after 400 updates. Protocols: Appendix C.6.

4.6 Computational Cost

TS-Dreamer reuses R2-Dreamer's architecture and policy inference, adding only window aggregation and alignment losses during training. In shared-GPU training on RTX 4090, TS-Dreamer records mean training times of 1.09 hours for DMC and 0.62 hours for Atari, with mean peak GPU memory allocations of 1.37 and 1.33 GiB. Appendix C.1 provides the full resource summary by method and hardware.

5 Conclusion

TS-Dreamer retains single-step alignment and additionally aligns image and state representations over matching temporal windows. The objectives at different scales jointly shape the same world model without increasing model capacity. Our experiments show stronger correspondence across temporal windows and improved readout of physical quantities and their changes. Ordered-motion analysis further shows that motion order is easier to identify from state sequences when model weights and the classifier training budget are held fixed.

TS-Dreamer achieved the highest mean and median returns in our DeepMind Control Suite comparison and competitive performance on Atari 100K. In FlappyBird, the two-scale configuration improved mean scores over the baselines. Scale ablations showed further gains from broader temporal coverage at fixed model capacity. Together, these results identify temporal scale as an important design choice for representation supervision in world models.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc Bellemare. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 29304–29320. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/f514cec81cb148559cf475e7426eed5e-Paper.pdf.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=xm6YD62D1Ub.

M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research*, 47:253–279, June 2013. ISSN 1076-9757. doi: 10.1613/jair.3912. URL https://jair.org/index.php/jair/article/view/10819.

George Bredis, Nikita Balagansky, Daniil Gavrilov, and Ruslan Rakhimov. Next Embedding Prediction Makes World Models Stronger. In *ICLR 2026 the 2nd Workshop on World Models: Understanding, Modelling and Scaling*, 2026. URL https://openreview.net/forum?id=SkAgjqPmhY.

Maxime Burchi and Radu Timofte. MuDreamer: Learning Predictive World Models without Reconstruction, May 2024. URL http://arxiv.org/abs/2405.15083. arXiv:2405.15083 [cs.AI].

Maxime Burchi and Radu Timofte. Learning Transformer-based World Models with Contrastive Predictive Coding. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 6949–6970, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/148c0aeea1c5da82f4fa86a09d4190da-Paper-Conference.pdf.

Anna Dawid and Yann LeCun. Introduction to latent variable energy-based models: a path toward autonomous machine intelligence. *Journal of Statistical Mechanics: Theory and Experiment*, 2024(10):104011, October 2024. doi: 10.1088/1742-5468/ad292b. URL https://doi.org/10.1088/1742-5468/ad292b.

Fei Deng, Ingook Jang, and Sungjin Ahn. DreamerPro: Reconstruction-Free Model-Based Reinforcement Learning with Prototypical Representations. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4956–4975. PMLR, July 2022. URL https://proceedings.mlr.press/v162/deng22a.html.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. DeepMDP: Learning Continuous Latent Space Models for Representation Learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2170–2179. PMLR, June 2019. URL https://proceedings.mlr.press/v97/gelada19a.html.

David Ha and Jürgen Schmidhuber. Recurrent World Models Facilitate Policy Evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning Latent Dynamics for Planning from Pixels. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, June 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to Control: Learning Behaviors by Latent Imagination. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

Danijar Hafner, Timothy P. Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with Discrete World Models. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=0oabwyZbOu.

Danijar Hafner, Kuang-Huei Lee, Ian Fischer, and Pieter Abbeel. Deep Hierarchical Planning from Pixels. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 26091–26104. Curran Associates, Inc., 2022. doi: 10.52202/068431-1892. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/a766f56d2da42cae20b5652970ec04ef-Paper-Conference.pdf.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, April 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-08744-2. URL https://doi.org/10.1038/s41586-025-08744-2.

Nick Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, Robust World Models for Continuous Control. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 47376–47405, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/cf73d57b6dcda32b293df7c2d5341f49-Paper-Conference.pdf.

Michael Hauri and Friedemann Zenke. Dreamer-CDP: Improving Reconstruction-free World Models Via Continuous Deterministic Representation Prediction [Tiny Paper]. In *ICLR 2026 the 2nd Workshop on World Models: Understanding, Modelling and Scaling*, 2026. URL https://openreview.net/forum?id=eDKiiHgFeq.

Lukasz Kaiser, Mohammad Babaeizadeh, Piotr Milos, Blazej Osinski, Roy H. Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. Model Based Reinforcement Learning for Atari. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=S1xCPJHtDB.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of Neural Network Representations Revisited. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 3519–3529. PMLR, May 2019. URL https://proceedings.mlr.press/v97/kornblith19a.html.

Chang Liu, Fei Suo, Yanzhou Jin, Zeyu Ping, Yusuke Iwasawa, Yutaka Matsuo, and Yaonan Zhu. Temporally Centered SIGReg Improves LeWorldModel Representations for Robot Policy Learning, August 2026. URL http://arxiv.org/abs/2607.26924. arXiv:2607.26924 [cs.LG].

Haoyu Ma, Jialong Wu, Ningya Feng, Chenjun Xiao, Dong Li, Jianye Hao, Jianmin Wang, and Mingsheng Long. HarmonyDream: Task Harmonization Inside World Models. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 33983–34007. PMLR, July 2024. URL https://proceedings.mlr.press/v235/ma24o.html.

S.G. Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989. doi: 10.1109/34.192463.

Paul Mattes, Rainer Schlosser, and Ralf Herbrich. Hieros: hierarchical imagination on structured state space sequence world models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 1476-4687. doi: 10.1038/nature14236. URL https://doi.org/10.1038/nature14236.

Naoki Morihira, Amal Nahar, Kartik Bharadwaj, Yasuhiro Kato, Akinobu Hayashi, and Tatsuya Harada. R2-Dreamer: Redundancy-Reduced World Models without Decoders or Augmentation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Je2QqXrcQq.

Tung D. Nguyen, Rui Shu, Tuan Pham, Hung Bui, and Stefano Ermon. Temporal Predictive Coding For Model-Based Planning In Latent Space. In *Proceedings of the 38th International Conference on Machine Learning*, pp. 8130–8139. PMLR, July 2021. URL https://proceedings.mlr.press/v139/nguyen21h.html.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. *CoRR*, abs/1807.03748, 2018. URL http://arxiv.org/abs/1807.03748. arXiv: 1807.03748.

Vaibhav Saxena, Jimmy Ba, and Danijar Hafner. Clockwork Variational Autoencoders. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=fU7-so5RRhW.

Max Schwarzer, Ankesh Anand, Rishab Goel, R. Devon Hjelm, Aaron C. Courville, and Philip Bachman. Data-Efficient Reinforcement Learning with Self-Predictive Representations. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=uCQfPZwRaUu.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind Control Suite, January 2018. URL http://arxiv.org/abs/1801.00690. arXiv:1801.00690 [cs.AI].

Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering Atari Games with Limited Data. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 25476–25488. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/d5eca8dc3820cad9fe56a3bafda65ca1-Paper.pdf.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International conference on machine learning*, pp. 12310–12320. PMLR, 2021. URL http://proceedings.mlr.press/v139/zbontar21a.

Amy Zhang, Rowan Thomas McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning Invariant Representations for Reinforcement Learning without Reconstruction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=-2FCwDKRREu.

AI USE STATEMENT

We used generative AI tools to polish the language and improve the readability of the manuscript. The authors take full responsibility for the final content.

A Model and Training Configurations

Table 2: Model and training configurations for the control benchmarks. Network widths and optimizer settings are shared across methods within each benchmark. Projector dimensions apply to TS-Dreamer and R2-Dreamer. SiLU denotes the sigmoid linear unit; RMSNorm denotes root mean square normalization. Motion models and classifiers follow Appendix C.6.

Setting	DMC	Atari 100K	FlappyBird
World model and policy networks			
Image input	64×64 RGB	64×64 RGB	64×64 RGB
Deterministic state width	2048	2048	512
Categorical variables $\times$ classes	32×16	32×16	16×16
Full state width	2560	2560	768
RSSM hidden width	256	256	128
Prior / posterior hidden layers	2 / 1	2 / 1	2 / 1
Recurrent blocks; initial state	8; zeros	8; zeros	8; zeros
Uniform mixture; sampling temperature	0.01; 1	0.01; 1	0.01; 1
Encoder channel widths	32, 48, 64, 64	32, 48, 64, 64	16, 24, 32, 32
Encoder stage	5×5 convolution, stride 1; 2×2 max pooling, stride 2		
Image embedding width	1024	1024	512
State projector	$2560 \to 1024$	$2560 \to 1024$	$768 \to 512$
Actor and critic hidden layers	3×256	3×256	3×128
Reward and continuation hidden layers	1×256	1×256	1×128
Hidden activation; normalization	SiLU; RMSNorm with $\epsilon = 10^{-4}$		
Actor distribution	Bounded normal	Categorical	Categorical
Reward and value output	255-bin symexp two-hot; continuation uses a Bernoulli head		
Optimization and imagination			
Optimizer for model, actor, and critic	One joint LaProp optimizer; $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-20}$		
Learning rate; linear warmup updates	4×10^{-5}; 1000	4×10^{-5}; 1000	8×10^{-5}; 1000
Adaptive gradient clipping	Threshold 0.3; parameter norm floor 10^{-3}		
Imagined transitions; discount horizon	15; 333	15; 333	50; 500
Return mixing; critic target update	$\lambda = 0.95$; exponential moving average fraction 0.02 each update		
Entropy coefficient; KL free nats	3×10^{-4}; 1	3×10^{-4}; 1	3×10^{-3}; 0.5
Interaction and replay			
Training budget in decisions	250k	100k	110k
Action repeat; nominal frames	2; 500k	4; 400k	1; 110k
Training seeds	41, 42, 43	61, 62, 63	41, 42, 43, 61, 62
Parallel collection environments	16	1	2
Batch size $\times$ sequence length	16×64	16×64	16×64
Replay capacity in transitions	500k	100k	150k
Replay transitions per decision	128	128	64
Replay storage; training precision	CPU; mixed	CPU; mixed	CPU; mixed
Representation objectives			
TS main configuration	$\mathcal{K} = \{1, 2\}$, $w_1 = w_2 = 1/2$; R2 uses $\mathcal{K} = \{1\}$		
BT loss weight; off-diagonal weight	0.05; 5×10^{-4}. Both methods use a single bias-free linear projector.		

Continued on next page

Table 2: Model and training configurations continued.

Setting	DMC	Atari 100K	FlappyBird
Baseline-specific heads	V3 uses image reconstruction. Pro uses 32-dimensional image and state projections with 2500 prototypes.		
Pro freeze duration (4% of training updates)	1250	500	Not compared

B Training Objectives

Alignment statistics. In the implementation, each feature is centered across the N_k valid windows and divided by $\sigma + 10^{-8}$ at $k = 1$ or $\max(\sigma, 10^{-3})$ at $k > 1$, where σ is the sample standard deviation computed using denominator $N_k - 1$. The fixed additive epsilon and an active standard deviation floor can break exact scale invariance. The resulting matrix product is divided by N_k and used in Eq. 5; without stabilization, this statistic is $(N_k-1)/N_k$ times the correlation matrix in Eq. 4. For longer windows, aggregation, standardization, and cross-correlation use 32-bit arithmetic. The alignment loss sums over features without division by D. At $k = 2$, the normalized sum in Eq. 2 is the two-point Haar low-pass combination (Mallat, 1989).

Scale configurations. The scale sweeps use cumulative powers of two, $\mathcal{K}_J = \{2^m : m = 0, \ldots, J-1\}$, where J is the number of scales. For $J \geq 2$, the single-step term receives half the total weight and the remaining half is distributed geometrically:

$$w_1 = \tfrac{1}{2}, \qquad w_{2^m} = \frac{2^{-m}}{2\sum_{u=1}^{J-1} 2^{-u}}, \quad m = 1, \ldots, J-1. \tag{10}$$

The single-scale configuration uses $w_1 = 1$.

Latent and prediction distributions. The KL terms in Eq. 8 use the posterior and prior distributions before uniform mixing, sum over stochastic components, and apply the free-nats threshold per sample before averaging over batch and time. Sampling uses hard Gumbel–softmax with straight-through gradients; mixture settings, temperature, and thresholds are given in Table 2. Reward and value heads use two-hot targets on 255 support points $\mathrm{symexp}(x)$ with x evenly spaced on $[-20, 20]$, where $\mathrm{symexp}(x) = \mathrm{sign}(x)(e^{|x|} - 1)$. Predictions are expectations over this support. Reward and Bernoulli continuation negative log-likelihoods each have unit weight.

Return targets. For discount horizon H, discount $\gamma = 1 - 1/H$, and return mixing λ, let b_t mark an episode boundary, including truncation, c_t denote continuation, and ν_t denote a bootstrap value. With $\eta_t = \lambda(1 - b_t)$, the detached return targets satisfy

$$R_t = r_{t+1} + \gamma c_{t+1} \left[(1 - \eta_{t+1})\nu_{t+1} + \eta_{t+1} R_{t+1} \right], \qquad R_{L-1} = \nu_{L-1}, \tag{11}$$

for a sequence of length L and $t = L-2, \ldots, 0$. Imagined trajectories use model-predicted rewards and continuations, $b_t = 0$, and $\nu_t = v_\psi(s_t)$. Replay targets use observed rewards and boundary flags, with ν_t given by the first imagined return from the posterior state at step t. Horizons and mixing parameters appear in Table 2.

Policy and value losses. The policy uses a likelihood-ratio loss with an entropy bonus and detached advantage $(R_t - v_\psi(s_t))/S$, following the actor-critic formulation in R2-Dreamer and DreamerV3 (Morihira et al., 2026; Hafner et al., 2025). Here S is the running difference between the 95th and 5th percentiles of imagined returns, bounded below by 1, with percentile estimates updated using an exponential moving average with an update fraction of 0.01. Policy and imagined value losses use continuation weights $\prod_{j=0}^{t} \gamma \hat{c}_j$, where $\hat{c}_j$ is predicted continuation. The value losses fit detached returns on imagined and replayed trajectories, together with detached scalar predictions from the slow value network; the replay loss masks episode-ending steps. Entropy and slow target update coefficients are listed in Table 2.

Joint objective. We jointly optimize the model, policy, imagined value, and replay value losses:

$$\mathcal{L}_{\mathrm{TS}} = \mathcal{L}_{\mathrm{model}} + \beta_{\pi}\mathcal{L}_{\mathrm{policy}} + \beta_{V}\mathcal{L}_{\mathrm{value}} + \beta_{\mathrm{RV}}\mathcal{L}_{\mathrm{repval}}. \tag{12}$$

The coefficients are $\beta_{\mathrm{align}} = 0.05$, $\lambda_{\mathrm{off}} = 5 \times 10^{-4}$, $\beta_{\mathrm{dyn}} = \beta_{\pi} = \beta_{V} = 1$, $\beta_{\mathrm{rep}} = 0.1$, and $\beta_{\mathrm{RV}} = 0.3$.

C Additional Results and Evaluation Protocols

Evaluation protocol. DMC and Atari use 30 evaluation episodes per model, with shared environment seeds 258000–258029 and 458000–458029, respectively. FlappyBird reports ten shared obstacle schedules, 157000–157009. Its baseline comparison and ablations all use training seeds 41, 42, 43, 61, and 62. Evaluation episodes are averaged within each model; training seeds are the statistical repetitions. Intermediate evaluations use five episodes, with seeds 257000–257004 for DMC and 457000–457004 for Atari.

C.1 Computational Cost

Table 3 reports computational resources for DMC, Atari 100K, and FlappyBird. R2-Dreamer and TS-Dreamer have matching parameter counts within each task: 11.840 to 11.848 million in DMC and 1.685 million in FlappyBird. Of 78 Atari runs per method, 41 TS-Dreamer, 28 R2-Dreamer, 11 DreamerV3, and three DreamerPro runs use an RTX 3060 Laptop GPU; the remaining runs use RTX 4090 GPUs. Costs are grouped by GPU type.

Table 3: Computational resources during shared-GPU training. Time is mean ± sample standard deviation (SD) in hours, excluding logged evaluations; GPU-specific averages reflect each method's task and seed allocation. Memory is mean peak PyTorch allocation in GiB. SD is omitted for a single run. V3, Pro, R2, and TS denote DreamerV3, DreamerPro, R2-Dreamer, and TS-Dreamer.

Benchmark / GPU	Method	Time / h	Peak / GiB
DMC / RTX 4090	V3	1.42 ± 0.14	1.89
	Pro	1.83 ± 0.23	2.86
	R2	0.99 ± 0.07	1.26
	TS	1.09 ± 0.11	1.37
Atari 100K / RTX 4090	V3	0.74 ± 0.07	1.87
	Pro	0.97 ± 0.12	2.85
	R2	0.56 ± 0.05	1.25
	TS	0.62 ± 0.08	1.33
Atari 100K / RTX 3060 Laptop	V3	1.38 ± 0.15	1.92
	Pro	1.90 ± 0.21	2.83
	R2	0.98 ± 0.07	1.21
	TS	1.05 ± 0.12	1.32
FlappyBird / RTX 3060 Laptop	R2	1.90 ± 0.02	2.25
	TS	1.90 ± 0.02	2.24

C.2 Control Benchmarks

Figures 8–11 show learning curves for all 20 DMC tasks and 26 Atari 100K games. Each panel compares the same four methods using means across three training seeds. Table 4 summarizes their final performance.

For the Atari summary, human-normalized scores are $100(R - R_{\mathrm{random}})/(R_{\mathrm{human}} - R_{\mathrm{random}})$, where R is the task return and the reference scores come from DeepMind's Atari data (Mnih et al., 2015).[1] Individual game curves show episode returns.

[1] https://raw.githubusercontent.com/google-deepmind/dqn_zoo/master/dqn_zoo/atari_data.py

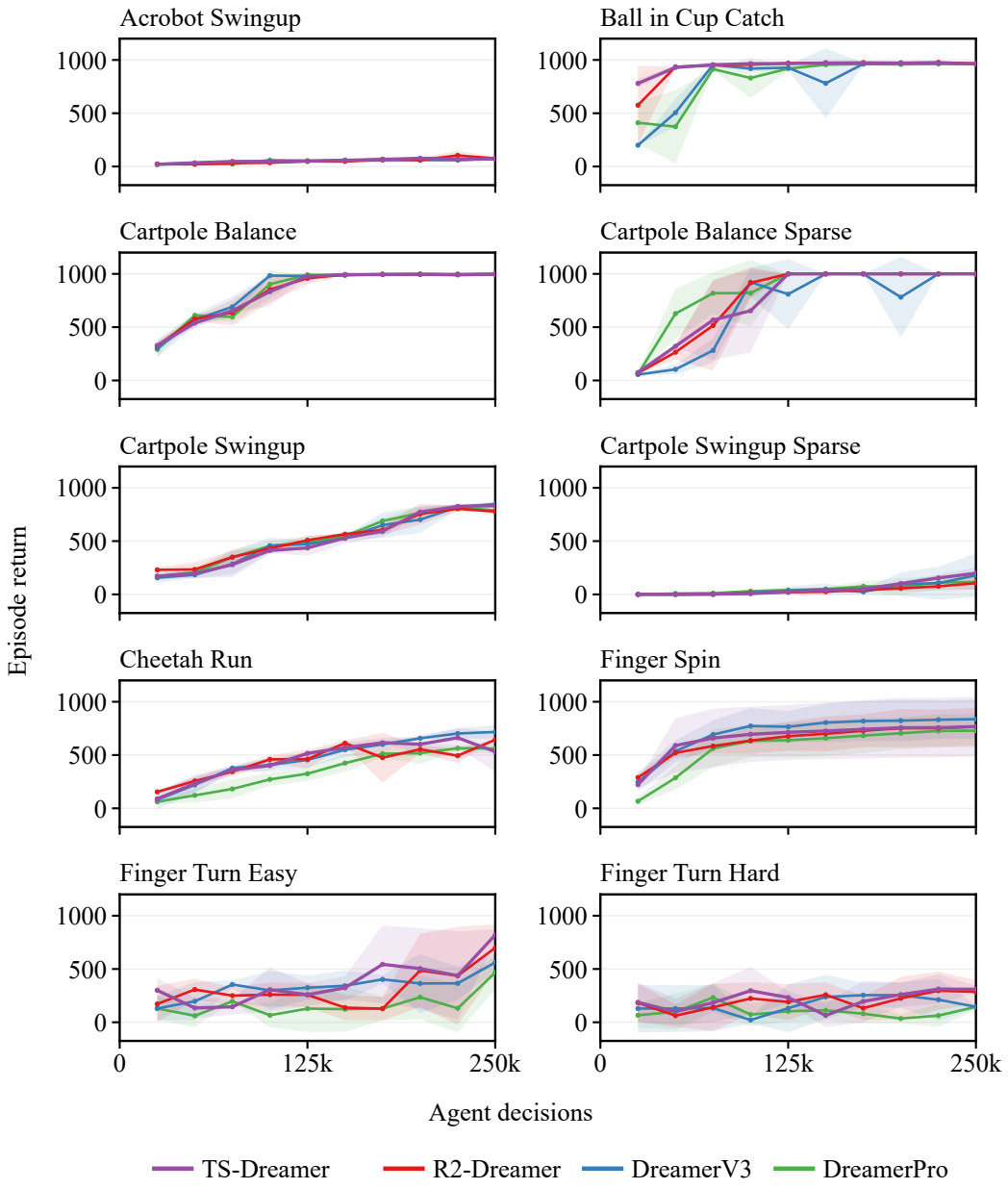


Figure 8: DMC learning curves (1/2). Lines show means and shading shows sample standard deviations across three training seeds. Intermediate checkpoints use five evaluation episodes and the final checkpoint uses 30. The horizontal axis counts agent decisions, with action repeat 2 and a budget of 250k decisions.

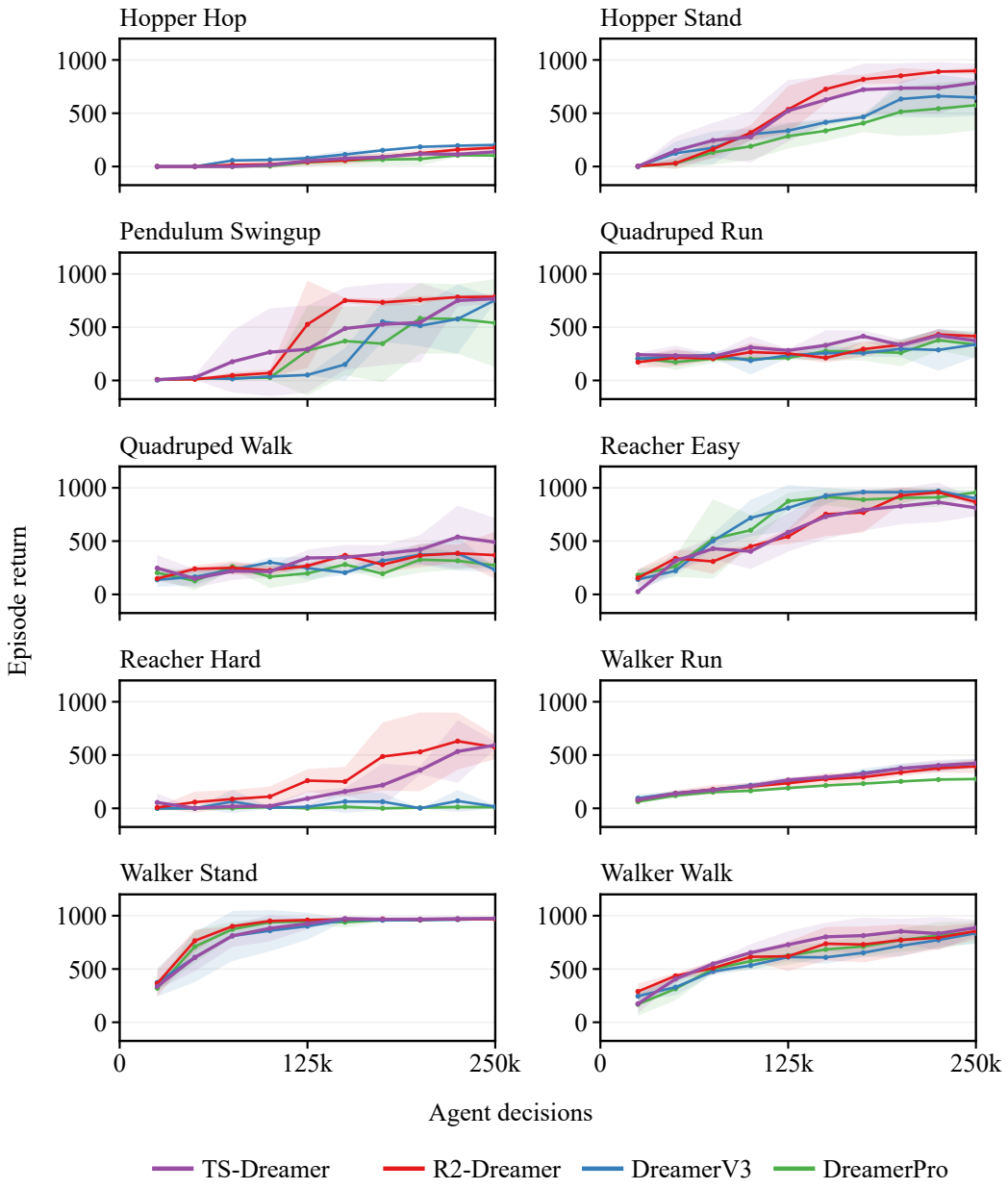


Figure 9: DMC learning curves (2/2). Lines show means and shading shows sample standard deviations across three training seeds. Intermediate checkpoints use five evaluation episodes and the final checkpoint uses 30. The horizontal axis counts agent decisions, with action repeat 2 and a budget of 250k decisions.

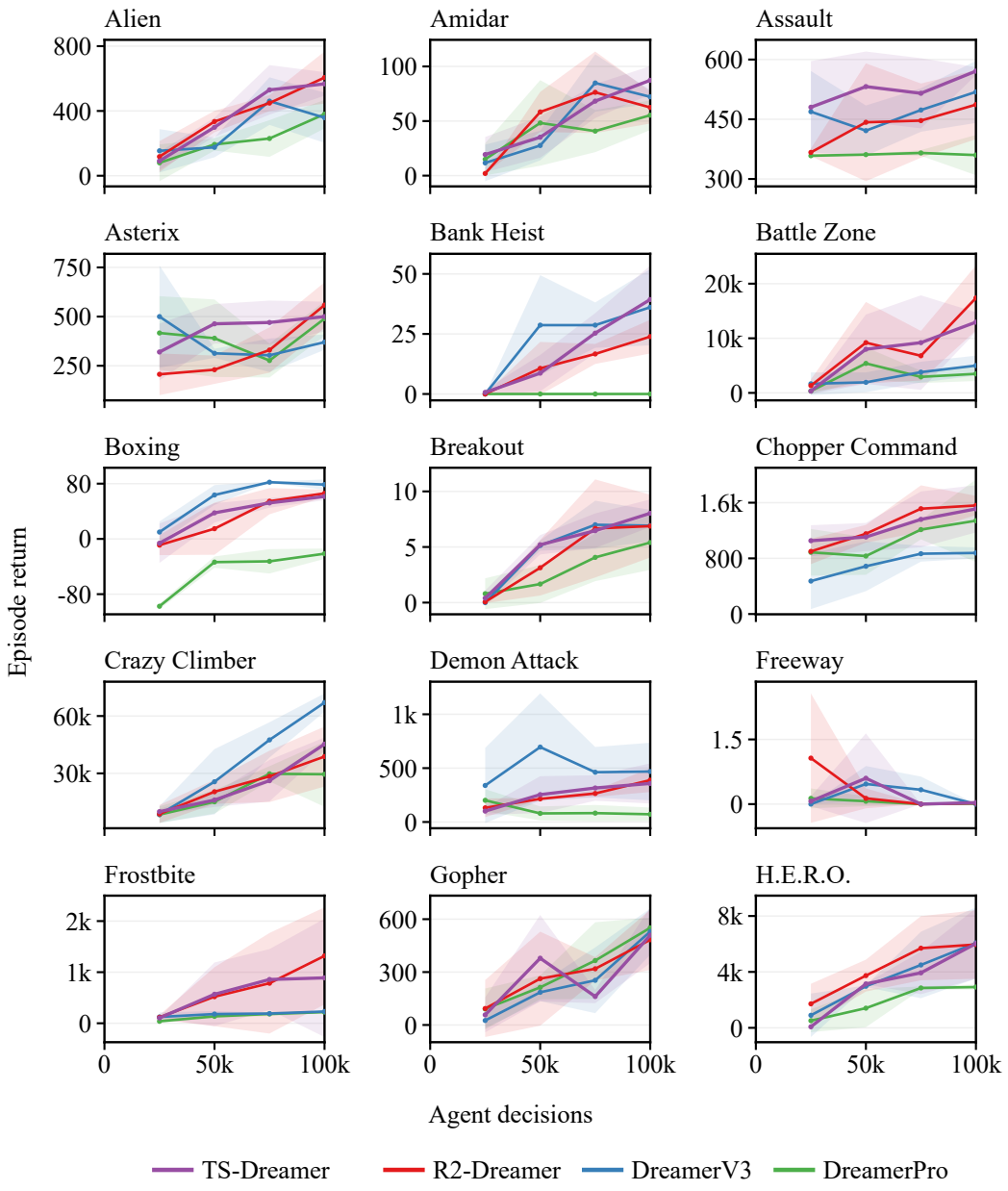


Figure 10: Atari 100K learning curves (1/2). Lines show means and shading shows sample standard deviations across three training seeds. Intermediate checkpoints use five evaluation episodes and the final checkpoint uses 30. The horizontal axis counts agent decisions, with action repeat 4 and a budget of 100k decisions.

Table 4: Final benchmark summaries across all tasks. DMC uses episode returns and Atari uses human-normalized percentages. Task scores average three training seeds, with 30 evaluation episodes per model. Lower average rank is better.

Benchmark	Metric	DreamerV3	DreamerPro	R2-Dreamer	TS-Dreamer
DMC	Mean	582.00	535.95	630.91	**636.68**
	Median	682.11	550.32	732.66	**766.80**
	Average rank	2.475	3.475	2.125	**1.925**
Atari 100K	Mean	**101.03**	77.64	74.14	71.41
	Median	9.32	4.32	17.27	**18.04**
	Average rank	2.231	3.385	2.269	**2.115**

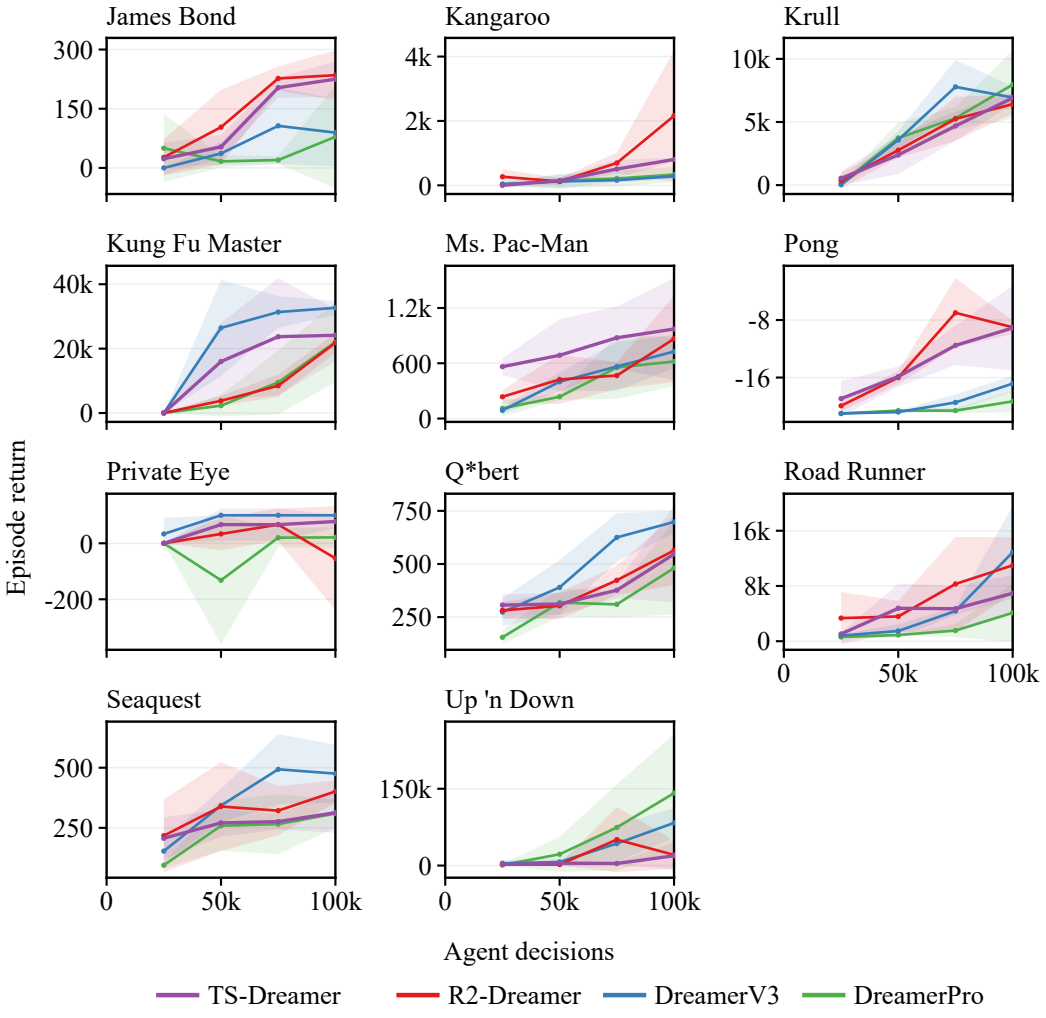


Figure 11: Atari 100K learning curves (2/2). Lines show means and shading shows sample standard deviations across three training seeds. Intermediate checkpoints use five evaluation episodes and the final checkpoint uses 30. The horizontal axis counts agent decisions, with action repeat 4 and a budget of 100k decisions.

C.3 FLAPPYBIRD TEMPORAL COVERAGE AND ALIGNMENT OBJECTIVES

Environment. Our FlappyBird game implementation uses 64×64 RGB observations and two actions, flap or no flap. All methods share the same environment and reward. In normalized screen coordinates, pipes have width 0.10, spacing 0.52, gap height 0.30, and horizontal speed 0.012 per frame; gap centers are sampled uniformly from $[0.39, 0.61]$. The bird has radius 0.025, downward acceleration 0.0038 per frame squared, maximum downward speed 0.030, and upward flap velocity 0.025, with an 11-frame flap cooldown. Collisions with pipes or screen boundaries end the episode. Training rewards are $+5$ per pipe passed, -5 per collision, and a proximity bonus that decreases linearly from 0.1 at the next gap center to zero at a vertical distance of 0.5. Reported scores count passed pipes only. Evaluation uses fixed obstacle schedules with the same geometry and dynamics.

Each model is evaluated at 110k decisions on the first ten shared obstacle schedules, 157000–157009.

Temporal coverage. Figure 5 reports the cumulative scale sets with longest windows of 1, 2, 4, 8, 16, and 32 decisions on the same five training seeds.

Alignment geometry. Let $X, Y \in \mathbb{R}^{N \times D}$ contain centered image embeddings and state projections, with rows indexing matching samples. Linear CKA is the normalized Frobenius inner product of their Gram matrices,

$$\mathrm{CKA}(X, Y) = \frac{\langle XX^\top, YY^\top \rangle_F}{\|XX^\top\|_F \, \|YY^\top\|_F}. \tag{13}$$

For nonzero Gram matrices, CKA equals one when they are proportional. It matches geometry of sample inner products up to a common scale while allowing independent orthogonal rotations of feature coordinates (Kornblith et al., 2019).

Alternative alignment objectives. Alternative alignment objectives use detached image targets and 32-bit floating-point arithmetic. Linear CKA uses $D(1 - \mathrm{CKA})$, differentiating through state normalization. VICReg weights invariance, variance, and covariance by 25, 25, and 1, respectively, then scales their sum by $D/25$. Variance penalties are averaged over image and state representations; covariance penalties are summed (Bardes et al., 2022).

Table 1 compares replacing the two-step objective or both objectives, with shared interaction budgets, model capacity, and evaluation schedules. All objective comparisons use $\mathcal{K} = \{1, 2\}$ with $w_1 = w_2 = 1/2$.

C.4 ADDITIONAL SCALE ABLATIONS

Table 5 reports the best tested $K_{\max}$ for three DMC tasks, two Atari games, and FlappyBird. Scale sets and weights follow Eq. 10, with model capacity, training budget, and total alignment weight fixed. DMC and Atari use three training seeds (41–43 and 61–63, respectively) and 30 shared evaluation episodes per model. Their single-step R2 controls are independently retrained with the main benchmark hyperparameters and seeds; the main results retain the original checkpoints. FlappyBird follows the five-seed, ten-episode protocol in Appendix C.3.

Table 5: Best tested extended scale sets. For each task, $K_{\max}$ is selected by the highest mean return among $\{4, 8, 16, 32\}$; each set retains all smaller powers of two. Scores are means across training seeds. R2 uses single-step alignment.

Task	R2	Best $K_{\max}$	TS-Dreamer
Walker Walk	778.2	8	945.2
Cheetah Run	517.7	4	638.6
Finger Turn Hard	290.1	32	343.4
Seaquest	389.1	16	390.7
Pong	-4.5	16	-1.8
FlappyBird	816.7	32	3621.1

C.5 Physical Readout and Alignment

Representation analysis protocol. The Walker analyses use six frozen checkpoints: R2-Dreamer and TS-Dreamer, each with training seeds 41–43. Alignment uses eight held-out trajectories after 64 initial steps to update the recurrent state, with 3,376 common endpoints across five window lengths. Image embeddings e_t and state projections $p_t = g_\phi(s_t)$ are aggregated using Eq. 2, then separately centered and standardized using population standard deviations floored at 0.001. Within each model, $C_{ij}^{(k)}$ correlates feature i of state projections with feature j of image embeddings. Figure 12 reports $D^{-1}\sum_{i=1}^{D} C_{ii}^{(k)}$ for $D = 1{,}024$; Table 6 gives the diagonal mean squared deviation from one and the off-diagonal root mean square. The mismatch control pairs different trajectories. Metrics are averaged over training seeds. Figure 6a shows seed 41 matrices at the same 64 evenly spaced original feature indices. Window lengths count agent decisions.

Table 6: Walker alignment at two steps. Diagonal error measures squared deviation from one. RMS denotes root mean square.

Metric	R2	TS
Diagonal correlation ↑	0.7870	**0.7998**
Diagonal error ↓	0.0534	**0.0477**
Off-diagonal RMS ↓	0.2477	**0.2444**
Mismatched correlation	−0.0229	−0.0235

Table 7: Physical readout. Counts of component/window combinations where TS has higher mean R^2 than R2, out of nine.

Representation	Full	64D
Image e	4/9	1/9
State s	**9/9**	8/9
Projection p	8/9	**9/9**

Physical readout protocol. Linear readout uses 12 training, four validation, and eight test trajectories. States s and projections p share the same posterior draw within each model and are paired with image embeddings e from corresponding observations. Representations and physical targets use the same causal Haar transform: the level component L sums the window and divides by $\sqrt{k}$; the change component D subtracts the second-half sum from the first-half sum and divides by $\sqrt{k}$. We use L at lengths $1, 2, 4, 8, 16$ and D at $2, 4, 8, 16$, giving nine component/window combinations. Test R^2 is averaged over target coordinates, then training seeds. Walker targets comprise height, 14 orientation components, and nine velocity components. Normalization uses training trajectories only. Validation mean squared error on standardized targets selects ridge regularization from $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$. We compare all features with a fixed orthogonal random projection to 64 dimensions. Differences subtract R2-Dreamer from TS-Dreamer at the same training seed index.

Table 8: Walker physical readout: differences in mean test R^2, TS-Dreamer minus R2-Dreamer. Positive values favor TS-Dreamer. L denotes the level component and D the change component; subscripts give window lengths in decisions.

	Level component					Change component			
Input	L_1	L_2	L_4	L_8	L_{16}	D_2	D_4	D_8	D_{16}
Full features									
Image e	-0.0013	-0.0019	-0.0015	-0.0025	0.0010	0.0022	0.0008	-0.0006	0.0004
State s	0.0075	0.0084	0.0055	0.0038	0.0013	0.0191	0.0137	0.0061	0.0034
Projection p	0.0067	0.0124	0.0069	0.0079	-0.0037	0.0227	0.0182	0.0037	0.0064
64 dimensions									
Image e	-0.0016	-0.0012	-0.0038	-0.0036	-0.0025	-0.0020	0.0037	-0.0061	-0.0057
State s	0.0080	0.0098	0.0065	0.0036	-0.0013	0.0112	0.0094	0.0224	0.0135
Projection p	0.0120	0.0124	0.0148	0.0079	0.0036	0.0135	0.0180	0.0153	0.0122

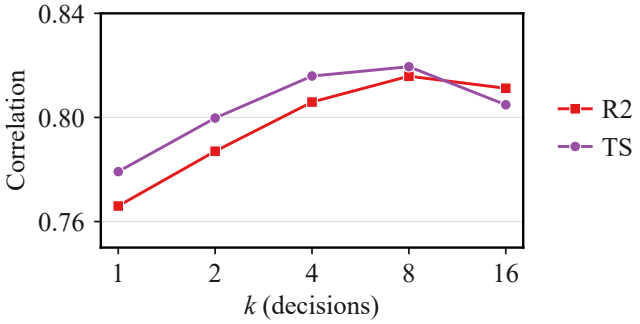


Figure 12: **Walker correlations across temporal windows.** Mean diagonal correlation between aggregated image embeddings and state projections on shared held-out trajectories, averaged across three training seeds. Window length k counts agent decisions.

C.6 Ordered-Motion Classification

Ordered-motion task and controls. The task contrasts continuous rotation with direction reversal after each lap around a stationary anchor. Matched videos have identical RGB frames, repetition counts, and final 32 frames, differing only in earlier frame order. Training, validation, and test sets contain 256, 96, and 128 scene pairs, respectively, each containing both classes. Training periods are 40, 48, and 56 frames. Transfer tests shift the initial angle by $\pi/2$, reverse rotation, change the circle center and scale its radius by 0.7, or change object/background colors. Held-out periods 44 and 52 test intermediate speeds. Time-shuffling reorders valid encoded states before classification.

Motion classification protocol. Compact world models train for 2,000 updates with KL and alignment losses, seeds 41 and 42, and no motion labels. The frozen image encoder and RSSM process frames with zero actions to obtain s_t, concatenating flattened stochastic z_t (64 dimensions) and deterministic h_t (256 dimensions). Features are standardized on training sequences, then a fixed, label-independent Gaussian projection maps 320 to 64 dimensions at every step. For T frames, the classifier receives the full $T \times 64$ sequence, excluding padding. Labels train only the classifier; all other components and preprocessing remain fixed. Controls apply the same preprocessing to projections p and embeddings e. The classifier uses three 32-channel convolutions (widths 1, 5, 5), rectified linear units, averaging over valid steps, and a linear output. Training uses batch size 64 and 400 updates, with an 800-update comparison. Validation binary cross-entropy on the original condition selects the learning rate from 0.001 and 0.003; selected classifiers are reused across tests. Figure 7 averages three paired classifier initializations per world model seed.

Table 9: Motion classification accuracy (%). Initializations are paired across methods. Scores average repetitions within each world model seed, then across the two seeds. Additional controls use 400 updates.

Test / representation	R2-Dreamer	TS-Dreamer	Untrained
State readout: three classifier initializations per world model seed			
Original, 400 updates	71.88	**80.73**	50.20
Start angle $+90°$, 400 updates	71.42	**80.21**	49.93
Reversed direction, 400 updates	70.44	**82.94**	49.74
Shuffled states, 400 updates	50.13	52.73	–
Original, 800 updates	94.60	94.60	50.52
Additional controls: one classifier initialization per world model seed			
Geometry / state s	60.74	**74.61**	49.41
Held-out speed / state s	57.23	**83.59**	50.98
Original / state projection p	51.17	54.49	50.00
Original / image embedding e	53.52	54.30	51.17