NEXT EMBEDDING PREDICTION MAKES WORLD MODELS STRONGER

Anonymous authors
Paper under double-blind review

ABSTRACT

Dreamer-style world models process trajectories sequentially, yet their representation losses are usually anchored to the current observation through pixel reconstruction or same-timestep latent alignment. This mismatch is limiting under partial observability, where successful control depends on retaining compact information about cues, landmarks, and spatial context long after they disappear. We introduce NE-Dreamer, a decoder-free world model that moves the representation target one step into the future. Given the history-conditioned latent state, a lightweight causal predictor forecasts the next encoder embedding, and training aligns this prediction to a stop-gradient target with a redundancy-reduction objective. This converts representation learning from same-timestep description into history-conditioned prediction, directly pressuring the latent state to preserve information useful beyond the current frame. Across bsuite diagnostics, MiniGrid Memory, and DMLab, NE-Dreamer improves most strongly in long-delay memory and navigation settings, while matching strong decoder-based and decoder-free baselines on the DeepMind Control Suite. These results identify temporal target placement as a distinct and complementary lever to architectural memory for improving partially observable model-based reinforcement learning.

1 INTRODUCTION

Model-based reinforcement learning (MBRL) (Ha & Schmidhuber, 2018; Hafner et al., 2024; Hansen et al., 2024; Schrittwieser et al., 2020) from high-dimensional observations relies on learning a compact latent state that supports long-horizon prediction and control (Hafner et al., 2020). This requirement is especially challenging under partial observability, where the information needed for decision-making may not be present in the current observation and must instead be inferred and retained from the agent's history.

Modern world models such as Dreamer (Hafner et al., 2024) incorporate temporal context through recurrent latent dynamics (Hafner et al., 2019) and sequence training. However, temporal capacity alone does not determine what information the representation is trained to preserve. Pixel reconstruction supplies dense supervision, but can allocate capacity to visually detailed aspects of the observation that are irrelevant for control (Kim et al., 2025). Decoder-free (Morihira et al., 2026; Hansen et al., 2024; Okada & Taniguchi, 2022; Deng et al., 2022; Burchi & Timofte, 2025) objectives reduce this burden but commonly emphasize same-timestep agreement (Morihira et al., 2026; Deng et al., 2022; Okada & Taniguchi, 2022), or induce temporal structure with contrastive learning on future representations (Burchi & Timofte, 2025). As a result, a latent state may encode the current observation well while failing to preserve information that remains important across time. This suggests that, in partially observable control, the critical design choice is not merely whether to remove reconstruction or add self-supervised representation learning, but where the representation target is placed in time and how latents are trained to carry decision-relevant information over time. This distinction is critical in memory- and navigation-intensive environments, where cues, landmarks, and spatial context may disappear long before they become behaviorally relevant. Prior work has often addressed such settings by strengthening temporal aggregation (Samsami et al., 2024; Zhang et al., 2023), for example through improved recurrent memory (Samsami et al., 2024) or effective tokenization (Micheli et al., 2024). These approaches are complementary to our method. Rather than only increasing the capacity to process history, we ask whether the representation objective itself can train the history-conditioned latent state to preserve information useful beyond the current

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.

observation. We study whether predicting the next embedding from latent history provides a useful alternative to same-timestep alignment, and examine alternative combinations of target placement and predictor architecture.

We introduce NE-Dreamer, a decoder-free world model that learns through next-embedding prediction. Instead of reconstructing pixels or aligning representations at the same timestep, NE-Dreamer temporally shifts the representation target to train latent history to preserve information needed for future prediction: a lightweight causal sequence predictor maps the history-conditioned latent state to a prediction of the next encoder embedding. The prediction is aligned with the actual next-step embedding (Xu et al., 2025) using a redundancy-reduction objective (Zbontar et al., 2021; Bardes et al., 2022; Balestriero & LeCun, 2025). This temporally shifted target supervises predictive information in the latent history, without requiring image reconstruction (Hafner et al., 2020), data augmentation (Deng et al., 2022), contrastive samples (Burchi & Timofte, 2025), or additional signals.

NE-Dreamer retains the Dreamer-style actor–critic control pipeline while replacing pixel reconstruction with temporal predictive alignment. Its strongest evidence comes from eight DMLab memory and navigation tasks (Beattie et al., 2016) and expanded reconstruction-free comparisons: NE-Dreamer mostly exceeds Dreamer, DreamerPro and R2Dreamer on subset of memory-heavy DMLab tasks. MiniGrid visual control (Chevalier-Boisvert et al., 2023), Behavioral Suite diagnostics (Osband et al., 2020), and continuous-control evaluations establish the objective's broader scope. Design studies examine predictor architecture, prediction horizon, and target construction to characterize practical choices across these settings.

Our main contributions are as follows:

1. We introduce next-embedding prediction from RSSM history as a decoder-free representation objective, using a training-only causal predictor while retaining Dreamer's imagination and actor–critic pipeline.
2. We evaluate the method across diagnostic, visual-navigation, and continuous-control tasks, with expanded reconstruction-free baseline comparisons demonstrating its practical value in memory and navigation heavy environments.
3. We characterize future targets and latent-history readout through design comparisons, and investigate practical horizon, loss, and target-branch choices.

2 Related Work

Decoder-free and predictive world models. Dreamer-like (Hafner et al., 2024; Chen et al., 2022; Burchi & Timofte, 2025; Zhang et al., 2023; Micheli et al., 2023) agents learn recurrent latent dynamics for imagination-based control, often using pixel reconstruction (Hafner et al., 2024) as dense supervision in combination with reward and continuation signals. Decoder-free world models train latent states using reward/value and previous-action prediction (Hutson et al., 2024; Burchi & Timofte, 2024), planning consistency (Hansen et al., 2024), contrastive objectives (Burchi & Timofte, 2025; Nguyen et al., 2021), redundancy reduction (Morihira et al., 2026), prototypical representations (Deng et al., 2022), or latent prediction (Lim et al., 2026). Next-embedding prediction also appears in vision pretraining: NEPA (Xu et al., 2025) predicts subsequent patch embeddings with a causal transformer. TWISTER (Burchi & Timofte, 2025) uses action-conditioned contrastive predictive coding in a transformer world model, while R2-Dreamer (Morihira et al., 2026) applies redundancy reduction to same-timestep representations. NE-Dreamer combines next-observation embedding targets, RSSM-history readout, and redundancy reduction in a training-only predictor. Our focus is the representation objective within a Dreamer-style RSSM: how can latent history be trained to remain useful for future prediction and delayed control?

Memory and navigation under partial observability. Partial observability has motivated methods that improve memory and navigation by increasing temporal capacity, including stronger recurrent models (Deng et al., 2023; Mattes et al., 2024; Samsami et al., 2024), attention mechanisms (Micheli et al., 2023; Wang et al., 2025), tokenization schemes (Micheli et al., 2024) and architecture-level Dreamer variants (Samsami et al., 2024). These approaches improve how the agent processes history. Our focus is complementary: we keep the Dreamer-style actor–critic and imagination pipeline fixed, and instead change what the latent history is trained to preserve. In memory- and navigation-heavy

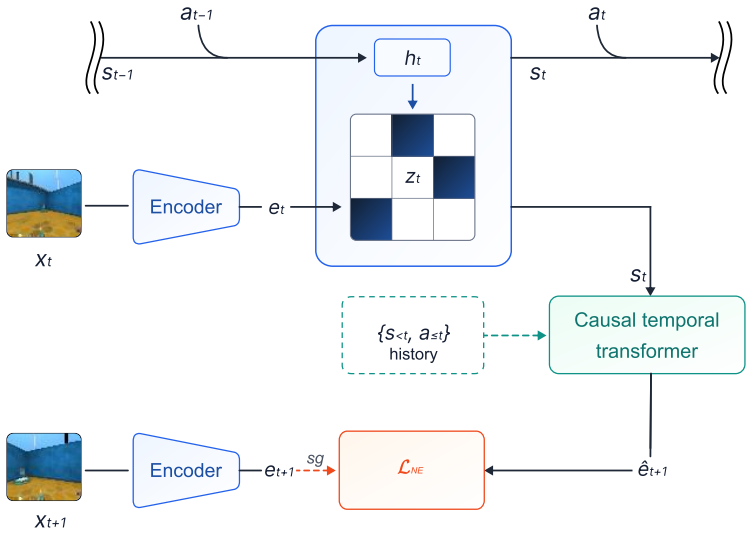

Figure 1: **Method overview.** NE-Dreamer keeps Dreamer's RSSM (Hafner et al., 2019) and imagination-based actor–critic, but replaces reconstruction with next-embedding alignment: a lightweight causal predictor (Vaswani et al., 2017) receives only latent history and actions and predicts the next encoder embedding. The predictor is used only during training.

tasks, cues, landmarks, object identities, and spatial context may disappear before they become behaviorally relevant. NE-Dreamer addresses this at the objective level by aligning the history-conditioned latent state with future representation targets, rather than merely matching the current observation or relying on additional temporal capacity.

A broader discussion is provided in Appendix E.2.

3 Method

3.1 Problem setup

We consider partially observable reinforcement learning from pixel observations. At timestep t, the agent receives observation $x_t \in \mathbb{R}^{H\times W\times C}$ and the reward r_t from the incoming transition under a_{t-1}, then selects action a_t. The continuation indicator $c_t \in \{0, 1\}$ is zero when that incoming transition terminates the episode and one otherwise; time-limit truncation is tracked separately.

Following Dreamer, we learn a latent world model that maintains a history-conditioned state s_t with a recurrent state-space model (RSSM). This state summarizes past observations and actions and is used both for world-model prediction and actor–critic learning through imagined latent rollouts.

3.2 Latent world model (RSSM)

We build on the recurrent state-space model (RSSM) (Hafner et al., 2019) used in Dreamer. The RSSM maintains a history-conditioned latent state $s_t = (h_t, z_t)$, where h_t is a deterministic recurrent state and z_t is a stochastic latent variable.

Encoder and latent inference An encoder maps observations to embeddings, $e_t = f_{\text{enc}}(x_t)$. Given the previous latent state and action, the deterministic state is updated recurrently:

$$h_t = f_{\text{rec}}(h_{t-1}, z_{t-1}, a_{t-1}).$$

Conditioned on h_t, the model defines a prior and posterior over the stochastic latent:

$$p_\phi(z_t \mid h_t), \qquad q_\phi(z_t \mid h_t, e_t).$$

During world-model training, we sample $z_t \sim q_\phi(z_t \mid h_t, e_t)$. During imagination, observations are unavailable, so future states are generated from the prior, $\hat{z}_t \sim p_\phi(z_t \mid h_t)$.

Reward and continuation heads The world model predicts reward and continuation from the latent state:

$$p_\phi(r_t \mid s_t), \qquad p_\phi(c_t \mid s_t).$$

Standard Dreamer also predicts observations through a pixel decoder $p_\phi(x_t \mid s_t)$. NE-Dreamer removes this decoder and replaces reconstruction with the future-predictive bottleneck defined in Sec. 3.3.

World-model loss The world model is trained with reward and continuation likelihoods, a prior–posterior regularizer, and the proposed next-embedding objective:

$$\mathcal{L}_{\text{wm}} = \mathcal{L}_{\text{rew}} + \mathcal{L}_{\text{cont}} + \beta_{\text{kl}}\mathcal{L}_{\text{kl}} + \beta_{\text{ne}}\mathcal{L}_{\text{NE}}.$$

The reward and continuation losses are negative log-likelihoods:

$$\mathcal{L}_{\text{rew}} = -\mathbb{E}_{q_\phi}\left[\log p_\phi(r_t \mid s_t)\right], \mathcal{L}_{\text{cont}} = -\mathbb{E}_{q_\phi}\left[\log p_\phi(c_t \mid s_t)\right].$$

The KL term regularizes the posterior toward the prior:

$$\mathcal{L}_{\text{kl}} = \mathbb{E}_{q_\phi}\left[\text{KL}\left(q_\phi(z_t \mid h_t, e_t) \parallel p_\phi(z_t \mid h_t)\right)\right].$$

Reward, continuation, and KL losses are averaged over valid timesteps and minibatch sequences; the alignment loss below uses a batch cross-correlation of embeddings. We use standard Dreamer stabilizers for $\mathcal{L}_{\text{kl}}$, including KL balancing and free-nats.

3.3 Next-embedding predictive alignment

NE-Dreamer replaces reconstruction with a future-predictive objective on latent history. The goal is not merely to predict another embedding, but to train the history-conditioned state to contain information useful beyond the current observation. The target is grounded in the next observation rather than an internal RSSM variable, providing predictive supervision without reconstructing pixels. Appendix E.1 develops the information-bottleneck interpretation.

Causal next-embedding predictor Given latent states $s_t = (h_t, z_t)$ inferred by the RSSM, a lightweight causal predictor maps latent history and actions to a prediction of the next encoder embedding:

$$\hat{e}_{t+1} = f_\theta(s_{\leq t}, a_{\leq t}),$$

where f_θ is a small causal temporal transformer (Xu et al., 2025) used only during training. It observes RSSM latents through time t and incoming actions through a_t, without pixels or future latents. Prediction uses the action-token position paired with s_t. This incoming-action convention defines primary NE-Dreamer.

The target is the encoder embedding of the next observation:

$$e^\star_{t+1} = \text{sg}(e_{t+1}) = \text{sg}(f_{\text{enc}}(x_{t+1})),$$

where $\text{sg}(\cdot)$ denotes stop-gradient. Gradients flow through $\hat{e}_{t+1}$ into the predictor, RSSM, and the encoder features used to infer the context latents $s_{\leq t}$, but not through the target embedding $e^\star_{t+1}$. Thus, the objective shapes the representation available to the RSSM while stopping gradients through the target branch. The shared encoder continues to learn through the prediction branch.

Alignment objective We instantiate the alignment loss with Barlow Twins (Zbontar et al., 2021), but the framework is not tied to this specific choice. Any collapse-resistant or task-appropriate embedding loss can be used, such as MSE when stable targets are available, VICReg-style variance/covariance regularization (Bardes et al., 2022), or SIGReg-style (Balestriero & LeCun, 2025) isotropy regularization. In our main experiments, Barlow Twins provides a simple redundancy-reduction objective between the prediction and the stop-gradient target.

Let $\tilde{\hat{e}}_{t+1}$ and $\tilde{e}^\star_{t+1}$ denote embeddings normalized per dimension over valid transitions in the minibatch. We define the cross-correlation matrix

$$C_{ij} = \mathbb{E}\left[\tilde{\hat{e}}_{t+1,i}\tilde{e}^\star_{t+1,j}\right],$$

where the expectation is over non-terminal transitions. Normalization uses the sample standard deviation and a numerical stabilizer. The next-embedding loss is

$$\mathcal{L}_{\mathrm{NE}} = \sum_i (1 - C_{ii})^2 + \lambda_{\mathrm{BT}} \sum_{i \neq j} C_{ij}^2.$$

The diagonal term aligns each predicted feature with its target counterpart, while the off-diagonal term discourages redundant dimensions. Applied to future targets, this loss asks whether the latent history contains information that predicts the next representation. It therefore changes what the RSSM is trained to preserve, while the reward, continuation, and KL terms retain their roles in learning the world model.

We use the same latent-imagination actor–critic objective as Dreamer, which optimizes behavior from imagined rollouts in the learned world model; full details are provided in Appendix A.4.

4 Experiments

4.1 Experimental setup

We evaluate NE-Dreamer across bsuite diagnostics, MiniGrid visual control, DMLab memory and navigation, and DeepMind Control. The comparisons assess capability profiles and performance across increasingly complex visual settings; design studies examine future targets, predictor architecture, and target construction. Baselines include DreamerV3 (Hafner et al., 2024), R2-Dreamer (Morihira et al., 2026), DreamerPro (Deng et al., 2022), TD-MPC2 (Hansen et al., 2024), and DrQ-v2 (Kostrikov et al., 2020) where applicable.

The evaluation addresses five questions:

1. Which capabilities change under next-embedding prediction?
2. Does the objective support strong visual-control performance?
3. Does it scale to demanding 3D memory and navigation tasks?
4. How does it perform on standard continuous-control benchmarks?
5. Which design choices lead to improved performance?

Dreamer-family comparisons retain the RSSM and imagination-based actor–critic pipeline, with task-specific budgets and configurations. The predictor is absent from policy execution. MiniGrid uses a 500k-step budget and five reported training seeds per method. DMLab comparisons use five seeds per method/task. Detailed hyperparameters and setup can be found in Appendix A.

Table 1: **bsuite capability profile.** Family scores; parentheses give the difference from Dreamer.

Task family	NE-Dreamer	Dreamer
Deep Sea	0.029 (+0.009)	0.020
Umbrella Length	0.783 (+0.200)	0.583
Memory Length	0.826 (+0.200)	0.626
Memory Size	0.376 (−0.024)	0.400

4.2 Behavioral Suite diagnostics: which abilities improve?

We first use Behavioral Suite (Osband et al., 2020) to diagnose the capability profile induced by NE-Dreamer. These tasks separate challenges that are often entangled in larger benchmarks: long-delay retention, memory capacity, exploration, and delayed credit assignment. Under our predictive-learning view, the expected gain is selective: preserving compact future-relevant information need not improve every temporal capability.

Long-delay memory retention. We begin with Memory Length, which consists of 23 task variants with increasing retention horizon. The agent must remember a compact cue across a delay. Figure 2 compares NE-Dreamer with Dreamer, R2-Dreamer, and DreamerPro. NE-Dreamer sustains strong performance over longer delays and achieves the highest aggregate score across all methods.

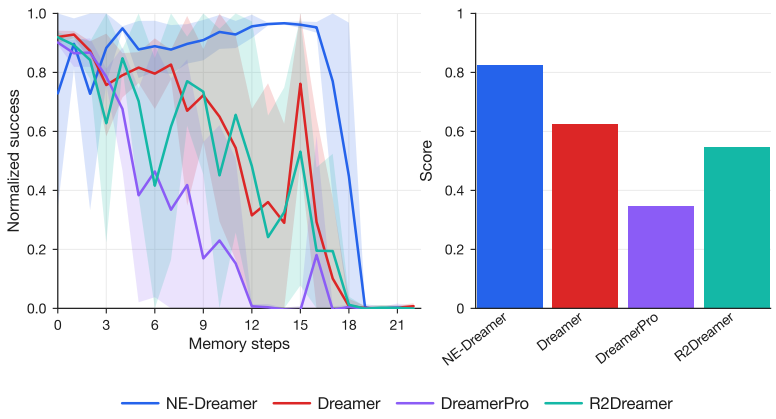

Figure 2: **BSuite Memory Length. Left**: Quality curve of per-method performance on each horizon variant; **Right**: Normalized score across 23 task variants with highest value achieved by NE-Dreamer.

Capability profile across bsuite. We next compare NE-Dreamer and Dreamer on the broader profile in Table 1. Gains are strongest on Memory Length and Umbrella Length, where performance depends on retention or decisions with delayed consequences. Deep Sea primarily stresses exploration and shows a smaller gain. Memory Size varies the amount of information to remember, rather than the retention horizon, and NE-Dreamer is slightly weaker on this family.

This distinction helps locate the method's practical benefit: predictive supervision supports long-delay behavior without solving exploration or memory capacity in general. An additional predictor sweep also favors next-target variants, with the current-state next-target configuration achieving its highest consistently aggregated score.

4.3 MiniGrid: visual control

We next test whether the representation objective remains useful in a higher-dimensional visual domain. MiniGrid Memory (Chevalier-Boisvert et al., 2023) provides compact visual-control tasks, allowing comparison with both reconstruction-based and decoder-free agents before moving to larger 3D environments. We evaluate MemoryS13Random and MemoryS17Random over 500k environment steps.

Table 2: **MiniGrid visual control.** Scores at 500k steps; five training seeds per method.

Method	S13	S17
NE-Dreamer	$\mathbf{0.993} \pm 0.001$	$\mathbf{0.995} \pm 0.001$
Dreamer	$\mathbf{0.993} \pm 0.001$	0.921 ± 0.167
DreamerPro	0.992 ± 0.001	0.793 ± 0.290
R2-Dreamer	0.720 ± 0.309	0.821 ± 0.325
TD-MPC2	0.211 ± 0.290	0.174 ± 0.390

Table 2 shows strong performance on both tasks. S13 is nearly saturated by several baselines: NE-Dreamer and Dreamer share the highest displayed mean. S17 separates the methods more clearly, with NE-Dreamer reaching highest value within std intervals. These results support the usefulness of next-embedding prediction for visual control while retaining the Dreamer actor–critic pipeline.

4.4 DMLab: scaling to 3D memory and navigation

We next scale evaluation to harder visual environments. DMLab (Beattie et al., 2016) combines egocentric observations, object interactions, spatial exploration, and delayed decisions. It tests whether predictive supervision remains useful when spatial context and control interact over extended trajectories, beyond the compact diagnostics above.

Because DMLab experiments are computationally expensive, we run each task for up to 50M interactions and focus the eight-task comparison on Dreamer. The RSSM and imagination-based actor–critic remain unchanged while next-embedding alignment replaces pixel reconstruction. Figure 3

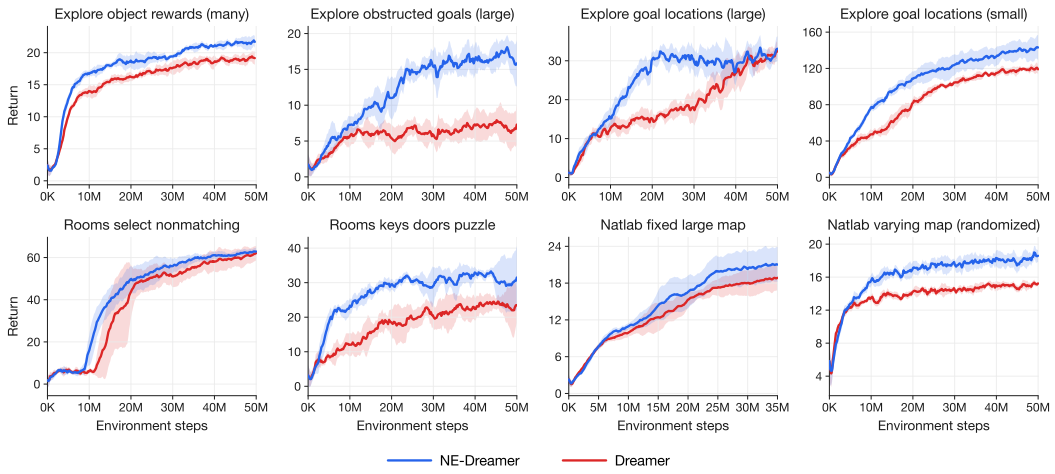

Figure 3: **DMLab memory and navigation.** NE-Dreamer improves over Dreamer across eight 3D tasks under a 50M-step budget. Gains are largest when successful control requires retaining spatial, object-level, or goal-relevant information beyond the current observation.

shows improvements across object selection, goal-directed exploration, and navigation, providing the broadest evidence for the objective's utility.

Additionally we run comparison with reconstruction-free agents to test that gains extend beyond removing the decoder. NE-Dreamer exceeds DreamerPro and R2-Dreamer in chosen subset of memory-heavy tasks. Detailed results can be found in Appendix B.1.

4.5 DeepMind Control: applicability beyond memory tasks

NE-Dreamer is motivated by memory and navigation, but the representation objective should also support standard visual control. We therefore evaluate on 20 DeepMind Control tasks (Tunyasuvunakool et al., 2020). These tasks emphasize continuous-action learning rather than delayed cue retention, making them a useful calibration of the objective's broader scope.

Unlike bsuite Memory, MiniGrid Memory, and DMLab, DMC tasks are primarily driven by continuous control rather than delayed cue retention. This makes them a useful calibration setting: if NE-Dreamer only helped memory tasks by over-specializing the representation, performance should degrade here. Instead, the desired outcome is to preserve strong control while improving the regimes where predictive belief states matter most.

Figure 4 shows that NE-Dreamer supports effective continuous-control learning across 20 tasks. Its aggregate returns exceeds DrQ-v2, while remaining within range of Dreamer-family baselines; relative performance varies across tasks. All 20 task curves appear in Appendix C.

Together with the memory and navigation results, DMC clarifies the scope of NE-Dreamer. Its main advantage appears when compact information must be retained for future prediction and delayed decisions, while standard continuous-control performance remains strong.

4.6 Ablations: future targets and latent-history readout

Which configurations perform well? We examine two design choices: moving the embedding target to the next timestep and using a causal transformer to read latent history. Starting from current-target alignment without the transformer, Table 3 adds each component separately and then combines them. The comparison characterizes practical configurations.

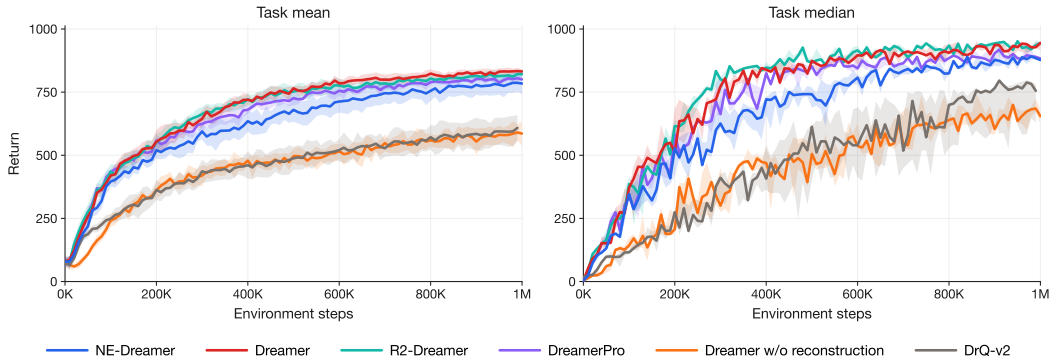

Figure 4: **DMC performance.** Across 20 standard continuous-control tasks, NE-Dreamer matches DreamerV3 and decoder-free baselines, showing that the future-predictive bottleneck improves memory-focused settings without degrading conventional latent-control performance.

Table 3: **Future targets and latent-history readout.** Next target predicts the next encoder embedding; the transformer reads RSSM history. Bold: best; underline: second-best. NE is strongest on bsuite, MiniGridS17 and Natlab Fixed, while achieving top-two competitive quality on Rooms Select.

Cell	Context	Target	BSuite	MiniGrid S17	Rooms Select	NatLab Fixed
A	self	e_t	0.548	0.924 ± 0.110	36.8 ± 15.3	14.4 ± 0.5
B	self	e_{t+1}	0.800	0.848 ± 0.151	27.9 ± 19.3	$\underline{17.5} \pm 2.3$
C	full	e_t	0.600	0.900 ± 0.100	$\mathbf{64.2} \pm 2.5$	12.5 ± 1.5
D (NE-Dreamer)	full	e_{t+1}	**0.826**	$\mathbf{0.988} \pm 0.018$	$\underline{55.8} \pm 4.0$	$\mathbf{17.7} \pm 2.1$

The current-target transformer leads on Rooms Select while obtaining low quality on NatLab fixed. Combining a temporal transformer with next-embedding prediction achieves the highest mean performance on 3 of 4 tasks. This consistency supports the combined design as a practical choice, with highest gains with combined setup.

Which target construction works? We compare four target branches to understand which parts of the predictive signal matter. Our default uses a stop-gradient embedding from the current online encoder (Online in Table 4). EMA targets instead use a moving-average encoder; shuffled targets break temporal correspondence; removing stop-gradient permits gradients through the target branch.

Table 4: **Target construction.** MiniGrid AUC and final score over three training seeds. Our detached online target has similar reported quality to EMA: higher AUC, lower final score. Bold: best; underline: second-best.

Target branch	AUC	Final score
Online (ours)	$\mathbf{0.744} \pm 0.076$	$\mathbf{0.995} \pm 0.001$
EMA encoder	$\underline{0.730} \pm 0.110$	$\mathbf{0.996} \pm 0.009$
Shuffled target	0.225 ± 0.132	0.365 ± 0.265
No stop-gradient	0.286 ± 0.191	0.372 ± 0.211

Table 4 favors temporally corresponding, detached targets over both shuffled targets and the no-stop-gradient control. The online target has slightly higher AUC than EMA, while EMA has slightly higher final score. This supports the simple detached-online construction as a practical choice without requiring a separate target encoder. The controls measure task performance, rather than directly diagnosing representational collapse. Protocol and auxiliary measurements are given in Appendix D.5.

What cue information is represented? The predictor can read earlier states, whereas the policy acts on the current RSSM state. We therefore probe both representations to ask where cue information remains decodable. Table 5 compares current-state versus history context and current versus next targets. The current RSSM state itself already summarizes past observations.

Table 5: **Cue information in RSSM and predictor features.** Linear-probe accuracy over delays $k \leq 16$, mean $\pm$ standard deviation over five training seeds. Gap: RSSM minus predictor accuracy. Bold highlights the mean gap closest to zero, comparing cue decodability across the two representations, not overall representation quality. Lower gap is better.

Configuration	Probe gap
A	-0.104 ± 0.109
B	-0.037 ± 0.049
C	-0.051 ± 0.048
D (NE-Dreamer)	$\mathbf{-0.002} \pm 0.054$

NE-Dreamer has the mean RSSM–predictor probe gap closest to zero, indicating similar cue-decoding accuracy in its current state and training-only readout. This is the comparison emphasized in Table 5.

Table 6: **Cue separability.** Identical scripted actions; three training seeds per configuration.

Variant	Separability
A	0.874
C	0.937
D (NE)	**0.989**
Shuffled target	0.461

The scripted-action comparison asks a complementary question: how well can states associated with different cues be separated? NE has the highest mean separability among the reported configurations (Table 6). Together, the measurements show decodable cue information. Incompletely controlled cue visibility and layout matching prevent a causal retention interpretation, and decodability alone does not demonstrate policy use. Appendix D.4 describes the measurement protocol.

Why one-step prediction? One-step prediction performs best among horizons 1, 2, 3, and 5 on Rooms Select, supporting the selected horizon for that task. Longer-horizon prediction remains an open design direction; the experiment does not identify the source of its performance shortfall. The horizon plot, loss-choice study, and alignment-weight sensitivity are in Appendix D.

5 Conclusion

We introduced NE-Dreamer, a decoder-free world model that trains Dreamer-style latent dynamics through a future-predictive bottleneck. Instead of reconstructing pixels or aligning latents only with the current observation, NE-Dreamer uses a lightweight causal predictor to predict the next encoder embedding from RSSM latent history and actions. This turns next-embedding prediction into direct pressure on the belief state: the latent history must preserve compact information that remains useful for future prediction and delayed control.

Empirically, this objective improves the regimes it is designed for. bsuite diagnostics show stronger long-delay retention, MiniGrid Memory confirms the effect in controlled visual cue-retention tasks, and DMLab demonstrates that the same mechanism scales to harder 3D navigation and memory environments. At the same time, DeepMind Control results show that these gains do not come at the cost of standard continuous-control performance. Ablations further show that future targets and causal latent-history readout are complementary: the strongest results arise when the model both has access to latent history and is explicitly trained to make that history predictive.

Overall, our results suggest that decoder-free world models need more than compact representations; they need predictive belief states. NE-Dreamer provides a simple way to train such states without changing the Dreamer actor–critic pipeline, adding reconstruction losses, or introducing a separate memory architecture. This points to future work on richer predictive bottlenecks, adaptive alignment losses, and scaling future-predictive latent-history training to broader embodied and long-horizon control settings.

REPRODUCIBILITY STATEMENT

Section 3 specifies the world-model architecture, next-embedding prediction objective, and gradient flow through the prediction and target branches. Appendix A provides reference hyperparameters, benchmark-specific training settings, predictor architecture, actor–critic objectives, parameter counts, and training-time measurements. The main MiniGrid and DMLab comparisons use five training seeds per method and task. Additional baseline comparisons and per-task continuous-control results appear in Appendices B and C. Appendix D reports design studies covering alignment weights, loss functions, prediction horizons, target construction, and representation diagnostics. We distinguish empirical observations from mechanistic interpretations and discuss the limitations of the representation probes in Section 4.6 and Appendix E.1.

AI USE STATEMENT

We used AI assistants to edit and polish author-written text. AI tools were not used to generate data, and the method and its derivations are the authors' own. The authors take full responsibility for the content of this paper.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael G. Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. *CVPR*, pp. 15619–15629, 2023. doi: 10.1109/cvpr52729.2023.01499. URL https://doi.org/10.1109/cvpr52729.2023.01499.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. *ICML*, pp. 1298–1312, 2022. URL https://proceedings.mlr.press/v162/baevski22a.html.

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 *International Conference on Learning Representations (ICLR)*, 2022. URL https://arxiv.org/abs/2105.04906.

Charles Beattie, Joel Z. Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, Julian Schrittwieser, Keith Anderson, Sarah York, Max Cant, Adam Cain, Adrian Bolton, Stephen Gaffney, Helen King, Demis Hassabis, Shane Legg, and Stig Petersen. DeepMind Lab, 2016. URL https://arxiv.org/abs/1612.03801.

Maxime Burchi and Radu Timofte. MuDreamer: Learning predictive world models without reconstruction, 2024. URL https://arxiv.org/abs/2405.15083.

Maxime Burchi and Radu Timofte. Learning transformer-based world models with contrastive predictive coding. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=YK9G4Htdew.

Chang Chen, Yi-Fu Wu, Jaesik Yoon, and Sungjin Ahn. TransDreamer: Reinforcement learning with transformer world models, 2022. URL https://arxiv.org/abs/2202.09481.

Xinlei Chen and Kaiming He. Exploring simple Siamese representation learning. *CVPR*, pp. 15750–15758, 2021. doi: 10.1109/cvpr46437.2021.01549. URL https://doi.org/10.1109/cvpr46437.2021.01549.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo Perez-Vicente, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & Miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. In *Advances in Neural Information Processing Systems*, volume 36, pp. 73383–73394, 2023. doi: 10.52202/075280-3209.

Fei Deng, Ingook Jang, and Sungjin Ahn. DreamerPro: Reconstruction-free model-based reinforcement learning with prototypical representations. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4956–4975. PMLR, 2022. URL https://proceedings.mlr.press/v162/deng22a.html.

Fei Deng, Junyeong Park, and Sungjin Ahn. Facing off world model backbones: RNNs, transformers, and S4. In *Advances in Neural Information Processing Systems*, volume 36, pp. 72904–72930, 2023. doi: 10.52202/075280-3188.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *CoRR*, abs/2006.07733, 2020. URL https://arxiv.org/abs/2006.07733.

David Ha and Jürgen Schmidhuber. World models. *CoRR*, abs/1803.10122, 2018. URL http://arxiv.org/abs/1803.10122.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels, 2019. URL https://arxiv.org/abs/1811.04551.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination, 2020. URL https://arxiv.org/abs/1912.01603.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models, 2024. URL https://arxiv.org/abs/2301.04104.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Miles Richard Hutson, Isaac Kauvar, and Nick Haber. Policy-shaped prediction: avoiding distractions in model-based reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=hgdh4foghu.

Kyungmin Kim, JB Lanier, and Roy Fox. Make the pertinent salient: Task-relevant reconstruction for visual control with distractions. In *Reinforcement Learning Conference (RLC)*, 2025. URL https://openreview.net/forum?id=o16C3Pky9u.

Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *CoRR*, abs/2004.13649, 2020. URL https://arxiv.org/abs/2004.13649.

Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Jing Yu Lim, Rushi Shah, Zarif Ikram, Samson Yu, Haozhe Ma, Tze-Yun Leong, and Dianbo Liu. Performance asymmetry in model-based reinforcement learning, 2026. URL https://arxiv.org/abs/2505.19698.

Paul Mattes, Rainer Schlosser, and Ralf Herbrich. Hieros: Hierarchical imagination on structured state space sequence world models. In *International Conference on Machine Learning (ICML)*, 2024. URL https://openreview.net/forum?id=IUBhvyJ9Sr.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models, 2023. URL https://arxiv.org/abs/2209.00588.

Vincent Micheli, Eloi Alonso, and François Fleuret. Efficient world models with context-aware tokenization, 2024. URL https://arxiv.org/abs/2406.19320.

Naoki Morihira, Amal Nahar, Kartik Bharadwaj, Yasuhiro Kato, Akinobu Hayashi, and Tatsuya Harada. R2-Dreamer: Redundancy-reduced world models without decoders or augmentation. In *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 Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8130–8139. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/nguyen21h.html.

Masashi Okada and Tadahiro Taniguchi. DreamingV2: Reinforcement learning with discrete world models without reconstruction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 985–991, 2022. doi: 10.1109/IROS47612.2022.9981405.

Ian Osband, Yotam Doron, Matteo Hessel, John Aslanides, Eren Sezener, Andre Saraiva, Katrina McKinney, Tor Lattimore, Csaba Szepesvari, Satinder Singh, Benjamin Van Roy, Richard Sutton, David Silver, and Hado Van Hasselt. Behaviour suite for reinforcement learning, 2020. URL https://arxiv.org/abs/1908.03568.

Mohammad Reza Samsami, Artem Zholus, Janarthanan Rajendran, and Sarath Chandar. Mastering memory tasks with world models. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=1vDArHJ68h.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588 (7839):604–609, December 2020. ISSN 1476-4687. doi: 10.1038/s41586-020-03051-4. URL http://dx.doi.org/10.1038/s41586-020-03051-4.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. ISSN 2665-9638. doi: 10.1016/j.simpa.2020.100022. URL https://www.sciencedirect.com/science/article/pii/S2665963820300099.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding, 2018. URL https://arxiv.org/abs/1807.03748.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Lingyi Wang, Rashed Shelim, Walid Saad, and Naren Ramakrishnan. DMWM: Dual-mind world model with long-term imagination. In *Advances in Neural Information Processing Systems*, volume 38, pp. 5613–5647, 2025. doi: 10.52202/085713-0174.

Sihan Xu, Ziqiao Ma, Wenhao Chai, Xuweiyi Chen, Weiyang Jin, Joyce Chai, Saining Xie, and Stella X. Yu. Next-embedding prediction makes strong vision learners. *arXiv preprint arXiv:2512.16922*, 2025. URL https://arxiv.org/abs/2512.16922.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. Barlow Twins: Self-supervised learning via redundancy reduction. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12310–12320, 2021. URL https://proceedings.mlr.press/v139/zbontar21a.html.

Weipu Zhang, Gang Wang, Jian Sun, Yetian Yuan, and Gao Huang. STORM: Efficient stochastic transformer based world models for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 27147–27166, 2023. doi: 10.52202/075280-1182.

A IMPLEMENTATION AND EVALUATION

A.1 HYPERPARAMETERS

Table 7 lists the reference configuration, with benchmark-specific settings and the representation-loss scales used by each method. Available run configurations accompany the evidence package.

Table 7: **Implementation settings.** Reference configuration and benchmark-specific hyperparameters.

Parameter	Symbol	Setting
General		
Replay Buffer Capacity	—	5×10^5
Batch Size	B	16
Batch Length	T	64
Optimizer	—	LaProp
Activation	—	RMSNorm + SiLU
Model Size	—	`size12M`
Input Image Resolution	—	64×64 RGB
Replayed Steps per Policy Step	—	1024 (BSuite) / 512 (MiniGrid) / 512 (DMC) / 32 (DMLab)
Environment Instances	—	1 (Bsuite) / 16 (MiniGrid) / 16 (DMC) / 16 (DMLab)
Action Repeat	—	1 (Bsuite) / 1 (MiniGrid) / 2 (DMC) / 4 (DMLab)
World Model		
Number of Latents	—	32
Classes per Latent	—	16
Prediction Loss Scale	β_{pred}	1.0
Dynamics Loss Scale	β_{dyn}	1.0
Representation Loss Scale	β_{rep}	0.1
Learning Rate	α	4×10^{-5}
Optimizer Moments	β_1, β_2	0.9, 0.999
Optimizer Epsilon	ϵ	1×10^{-20}
Gradient Clipping	AGC	0.3
Slow Value Momentum	—	0.02
Model Return Lambda	λ	0.95
Actor Critic		
Imagination Horizon	H	15
Discount	γ	$332/333$
Return Lambda	λ	0.95
Critic EMA Decay	—	0.98
Critic EMA regularizer	—	1.0
Return Normalization Percentiles	—	5th and 95th
Return Normalization Decay	—	0.99
Actor Entropy Scale	η	3×10^{-4}
Learning Rate	α	4×10^{-5}
Optimizer Moments	β_1, β_2	0.9, 0.999
Optimizer Epsilon	ϵ	1×10^{-20}
Gradient Clipping	AGC	0.3
Per method loss weight		
Dreamerv3 reconstruction loss	—	1.0
NE-Dreamer next-embedding loss	—	1.0
R2-Dreamer barlow-twins loss	—	0.05
DreamerPro SwAV loss	—	1.0
NE-Dreamer transformer configuration		

Table 7 (continued)

Parameter	Symbol	Setting
hidden dim	—	256
num layers	—	2
num heads	—	4
Additional hyperparameters	—	
BT redundancy λ	—	5×10^{-4}

A.2 TRAINING TIME

Training time on Select Non-matching task Table 8 compares total experiment wall time on Rooms Select. NE-Dreamer and Dreamer have similar mean wall times, while R2-Dreamer is faster in this comparison. Jobs used one H100 and the same node configuration. The predictor is absent from policy execution.

Table 8: **Training time on Rooms Select.** Total wall time, mean $\pm$ standard deviation; relative time is normalized to Dreamer.

Method	Wall time (h) mean $\pm$ std	Rel. vs Dreamer mean $\pm$ std
Dreamer	86.0 ± 5.1	1.000 ± 0.000
NE-Dreamer	87.5 ± 2.7	1.019 ± 0.051
R2-Dreamer	78.0 ± 4.8	0.911 ± 0.096

A.3 PARAMETER COUNTS

Table 9 separates the shared RSSM and control components from each representation module. Counts omit the duplicate frozen world model and actor–critic used at inference. Total training parameter counts differ across methods.

Table 9: **Parameter counts by component.** Configuration: `size12M`, 64×64 RGB, eight one-hot actions.

Component	Dreamer	R2Dreamer	NE-Dreamer	DreamerPro
Encoder	220,416	220,416	220,416	220,416
RSSM	5,982,208	5,982,208	5,982,208	5,982,208
Decoder	1,272,003	0	0	0
Reward	721,407	721,407	721,407	721,407
Continue	656,129	656,129	656,129	656,129
Actor	790,024	790,024	790,024	790,024
Critic	853,503	853,503	853,503	853,503
Critic target	853,503	853,503	853,503	853,503
R2 - Projector	0	2,621,440	0	0
NE transformer	0	0	2,239,744	0
Prototypes	0	0	0	80,000
Obs projection	0	0	0	32,800
Feat projection	0	0	0	81,952
EMA encoder	0	0	0	220,416
EMA obs projection	0	0	0	32,800
Total	**11,349,193**	**12,698,630**	**12,316,934**	**10,525,158**

A.4 ACTOR–CRITIC LEARNING

NE-Dreamer uses the standard Dreamer-style actor–critic pipeline. After training the world model, the agent improves its policy by rolling out imagined trajectories in latent space. We denote the imagined latent state by

$$s_t = (h_t, \hat{z}_t), \qquad \hat{z}_t \sim p_\phi(z_t \mid h_t).$$

At each imagined step, the policy samples an action and the world model predicts reward and continuation:

$$a_t \sim \pi_\theta(a_t \mid s_t), \qquad \hat{r}_t, \hat{c}_t \sim p_\phi(r_t, c_t \mid s_t).$$

The critic is trained to predict λ-returns computed along imagined rollouts:

$$R_t^\lambda = \hat{r}_{t+1} + \gamma \hat{c}_{t+1} \left((1-\lambda) V_\psi(s_{t+1}) + \lambda R_{t+1}^\lambda \right). \tag{1}$$

We optimize the critic with a negative log-likelihood objective:

$$\mathcal{L}_{\text{critic}}(\psi) = -\mathbb{E}_{p_\phi, \pi_\theta} \left[\sum_{t=1}^{H} \log p_\psi(R_t^\lambda \mid s_t) \right]. \tag{2}$$

The actor is trained to maximize imagined returns, using the same normalized advantage objective as DreamerV3:

$$\mathcal{L}_{\text{actor}}(\theta) = -\mathbb{E}_{p_\phi, \pi_\theta} \left[\sum_{t=1}^{H} \left[\text{sg}\left(\frac{R_t^\lambda - V_\psi(s_t)}{\max(1, S)} \right) \log \pi_\theta(a_t \mid s_t) + \eta \, \mathcal{H}[\pi_\theta(\cdot \mid s_t)] \right] \right], \tag{3}$$

where S is the return normalization scale, η is the entropy coefficient, and $\text{sg}(\cdot)$ denotes stop-gradient.

All actor–critic components, imagination horizon, return estimation, and optimization hyperparameters are retained from the shared implementation; the reference settings are listed in Table 7. Thus, NE-Dreamer changes only the world-model representation objective, replacing pixel reconstruction with temporally shifted next-embedding prediction.

B ADDITIONAL DMLAB RESULTS

B.1 EXPANDED BASELINE COMPARISONS

Table 10: **DMLab reconstruction-free baselines on selected tasks.** Final return and normalized learning AUC, mean ± sample standard deviation.

Metric	NE-Dreamer (ours)	R2-Dreamer	DreamerPro
Rooms Select			
Final return	$\mathbf{58.56} \pm 2.84$	37.54 ± 28.95	5.80 ± 0.63
AUC	$\mathbf{40.34} \pm 4.65$	29.07 ± 21.30	5.78 ± 0.05
NatLab Regrowth			
Final return	7.54 ± 0.36	$\mathbf{7.89} \pm 0.89$	3.37 ± 0.20
AUC	$\mathbf{6.48} \pm 0.22$	6.21 ± 0.20	4.01 ± 0.08

Normalized learning AUC integrates each observed curve through the common 49.9M cutoff without extrapolation. Table 10 reports both metrics with sample standard deviations across training seeds. Tables 10 and 11 display Rooms Select and NatLab Regrowth. In comparison with reconstuction-free baselines NE-Dreamer achieves higher quality on Rooms Select task and performs comparably to R2-Dreamer on Natlab Regrowth task.

Table 11: **DMLab comparison with TD-MPC2 on selected tasks.** Returns at the 20M-step cut using the discrete-action TD-MPC2 port.

Task	NE-Dreamer	TD-MPC2
Rooms Select Nonmatching	39.04 ± 15.45	11.00 ± 7.07
NatLab Varying Map Regrowth	6.62 ± 1.14	5.96 ± 1.36

TD-MPC2 uses a different planning architecture and a discrete/episodic environment port. NE-Dreamer achieves higher mean returns than TD-MPC2.

C DeepMind Control task results

Figure 5 compares all 20 tasks for NE-Dreamer, Dreamer, R2-Dreamer, DreamerPro, Dreamer without reconstruction, and DrQ-v2. The aggregate performance in the main paper combines task-dependent outcomes, including shortfalls on Cartpole Swingup, Finger Spin, and Hopper Hop. These task-dependent results accompany effective continuous-control learning; NE-Dreamer remains competetive to Dreamer-family baselines in aggregate.

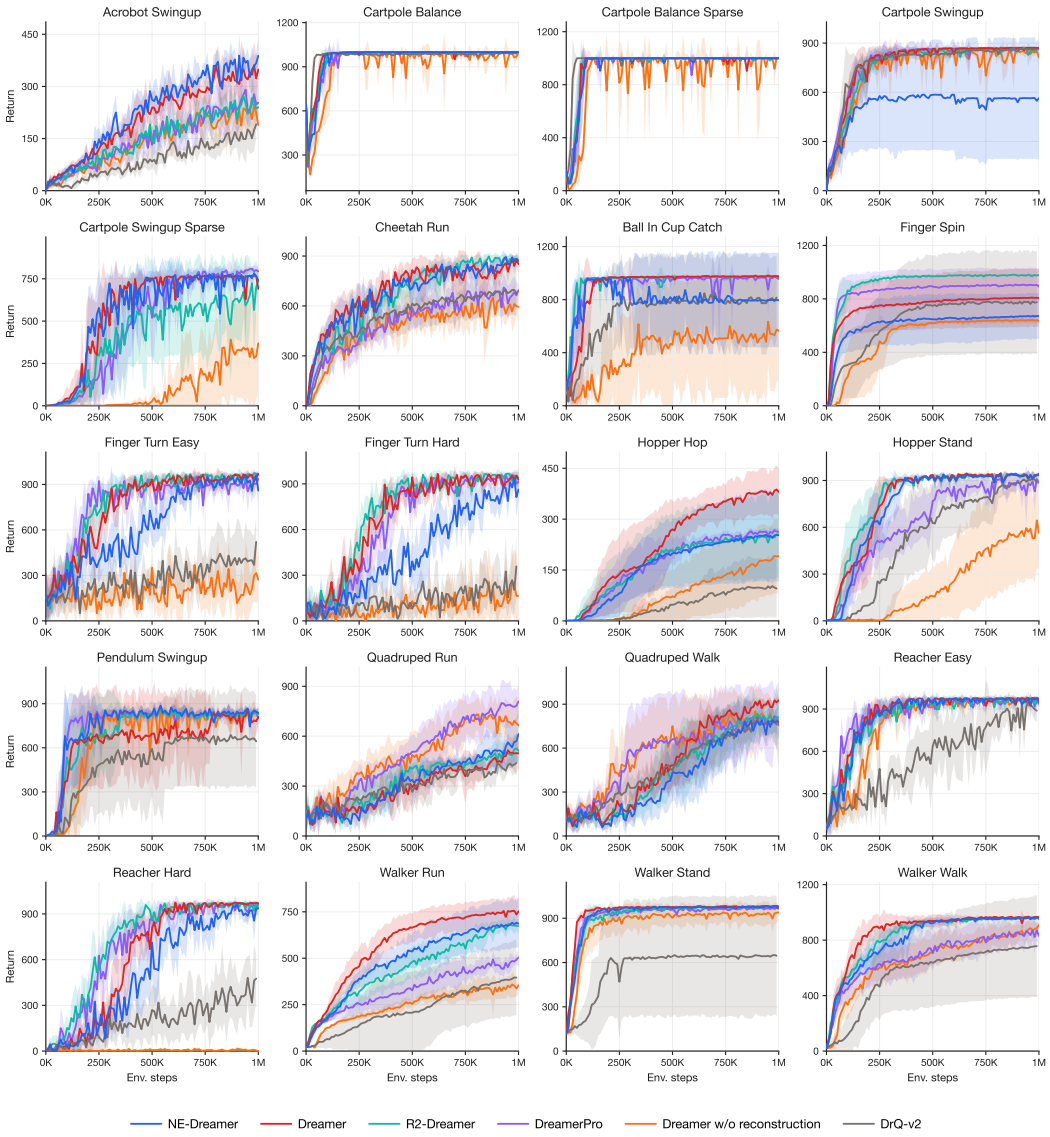

Figure 5: **Continuous-control task results.** Aggregate learning curves for all 20 DMC tasks.

D REPRESENTATION AND DESIGN DIAGNOSTICS

D.1 HYPERPARAMETER STABILITY

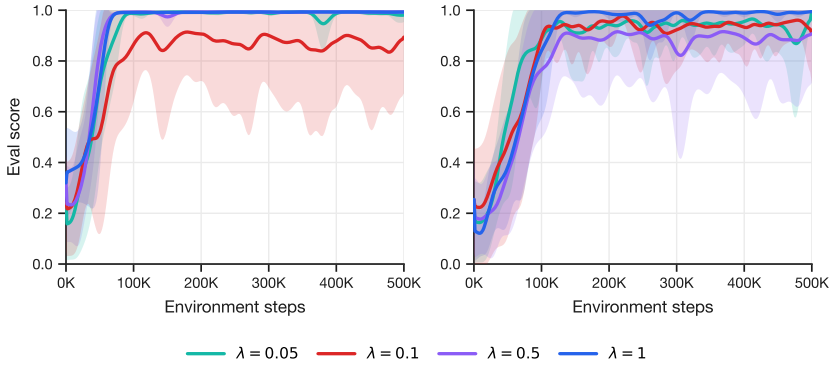

Figure 6: **Alignment-weight sensitivity.** MiniGrid alignment-weight comparison.

Figure 6 studies the weight of next-embedding alignment on both MiniGrid variants. Varying this coefficient changes the balance between predictive supervision and the reward, continuation, and KL terms. The curves show how this balance affects learning speed and stability across the tested weights.

D.2 LOSS CHOICE

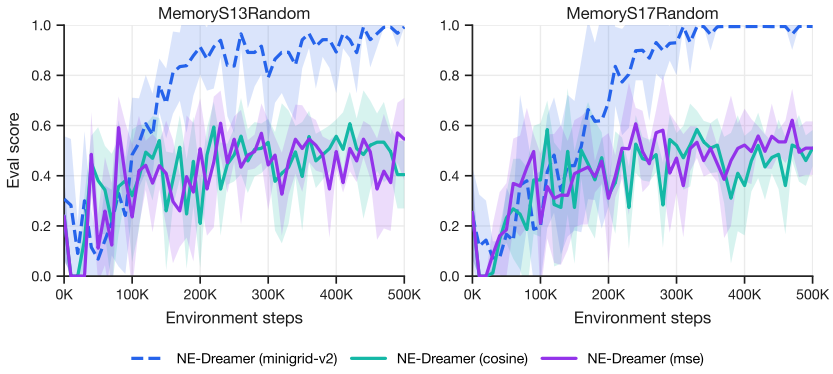

Figure 7: **Next-embedding loss choice.** Redundancy reduction, plain MSE, and cosine alignment.

Figure 7 compares different losses for next-embedding alignment. Plain MSE and cosine alignment produce weaker and less stable learning than redundancy reduction in these tasks. Bringing predicted features close to their targets is therefore not the only practical consideration; controlling feature redundancy also matters in this comparison. The framework is not tied to Barlow Twins, but these results support its use here.

D.3 Prediction horizon

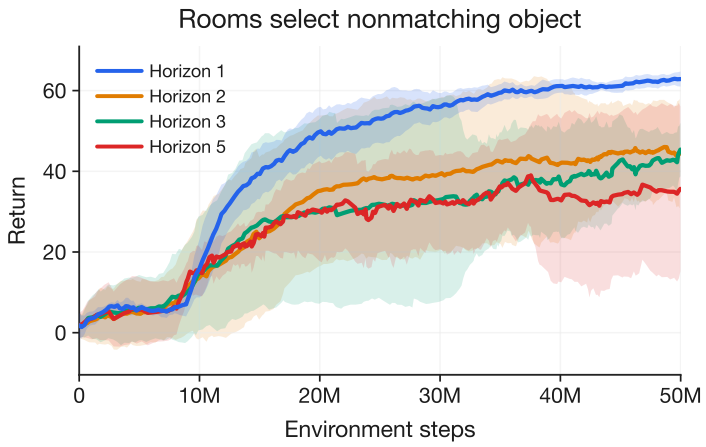

Figure 8: **Prediction horizon on Rooms Select.** One-step prediction performs best among horizons 1, 2, 3, and 5.

Figure 8 compares one-step prediction with horizons 2, 3, and 5 on Rooms Select. One-step prediction performs best, supporting a local predictive target in this setting. Longer horizons may introduce harder or more uncertain targets, but the curves do not distinguish that explanation from optimization difficulty. Extending prediction through latent overshooting or temporally abstract targets remains a promising direction.

D.4 Representation and cue measurements

We use the reported frozen linear probes of current RSSM features s_t and training-time predictor features u_t, averaged over delays $k \leq 16$ after the stated cue offset. Table 5 gives means and sample standard deviations over five training seeds, computed after averaging delays within each seed. The configuration labels specify predictor context and target time; current-state context still includes the RSSM's recurrent history. The probe gap compares two representations.

For cue separability, the procedure reset two environments with different cues and applied identical actions.

D.5 Target-branch controls

Additional measurements give an EMA probe-accuracy gap of $+0.023 \pm 0.077$ (RSSM minus predictor), shuffled-target cue recovery 0.452 ± 0.081 and effective ranks 94.2 for the ordinary target and 51.3 for the shuffled target. These measurements describe cue decodability and feature variation under the target controls.

E Interpretation and additional related work

E.1 Information-bottleneck interpretation

Dreamer can be interpreted as a variational information bottleneck on the RSSM state. At each step, the posterior receives the current observation through the encoder,

$$q_\phi(z_t \mid h_t, e_t), \qquad e_t = f_{\text{enc}}(x_t),$$

while the prior

$$p_\phi(z_t \mid h_t)$$

is predicted only from history. The KL term limits how much new information from the current observation can enter the stochastic state:

$$I(X_t; Z_t \mid H_t) \;\le\; \mathbb{E}\left[\mathrm{KL}\big(q_\phi(z_t \mid h_t, e_t) \,\|\, p_\phi(z_t \mid h_t)\big)\right].$$

This bound concerns the stochastic posterior channel conditioned on the deterministic state. It does not bound information in the entire deterministic state or the predictor's full history. KL balancing and free-nats further distinguish the implemented regularizer from an exact information constraint.

In reconstruction-based Dreamer, this bottleneck is trained with an observation likelihood:

$$\mathcal{J}_{\text{Dreamer}} = \mathbb{E}_{b,t,q_\phi}\Big[\log p_\phi(x_t \mid s_t) + \log p_\phi(r_t \mid s_t) + \log p_\phi(c_t \mid s_t) \\ - \beta\,\mathrm{KL}\big(q_\phi(z_t \mid h_t, e_t) \,\|\, p_\phi(z_t \mid h_t)\big)\Big],$$

where $s_t = (h_t, z_t)$ and the expectation averages minibatch sequences b, valid timesteps t, and posterior samples. This schematic objective omits the implementation's KL balancing and free-nats. The reconstruction term trains the compressed state to predict the current observation. This provides dense supervision, but it can also encourage the state to retain visual detail that is not needed for control.

NE-Dreamer keeps the same RSSM bottleneck but changes the relevance variable. We remove the pixel likelihood and instead train the latent history to predict a future encoder representation:

$$\hat{e}_{t+1} = T_\theta(s_{\le t}, a_{\le t}), \qquad e^\star_{t+1} = \mathrm{sg}(f_{\text{enc}}(x_{t+1})).$$

The corresponding objective can be viewed as

$$\mathcal{J}_{\text{NE}} = \mathbb{E}_{b,t,q_\phi}\Big[\log p_\phi(r_t \mid s_t) + \log p_\phi(c_t \mid s_t) \\ - \beta\,\mathrm{KL}\big(q_\phi(z_t \mid h_t, e_t) \,\|\, p_\phi(z_t \mid h_t)\big)\Big] - \beta_{\text{ne}}\,\mathcal{L}_{\text{NE}}.$$

Here $\mathcal{L}_{\text{NE}}$ is the single batch-level cross-correlation loss over matched predictor/target pairs defined in the main text. As an information-bottleneck interpretation, NE-Dreamer trades off compression of observation information through the RSSM KL with predictability from the latent history:

$$\max \sum_t \mathcal{R}\left(S_{\le t}, A_{\le t}; E_{t+1}\right) - \beta \sum_t I(X_t; Z_t \mid H_t),$$

where $\mathcal{R}$ denotes a surrogate for future-representation predictability, implemented with Barlow Twins in our main experiments. This is a conceptual interpretation rather than an equality between the implemented loss and mutual information, or a guarantee of a sufficient belief state or absence of collapse.

This differs from reconstruction in two ways. First, the target is an encoder representation rather than pixels, so the model is not trained to reproduce all observation-level detail. Second, the predictor T_θ is lightweight and causal: it receives only RSSM latents and actions, not pixels, future latents, or x_{t+1}. The auxiliary objective therefore encourages the RSSM history $s_{\le t}$ to contain information useful for predicting the future representation.

The loss is defined in encoder space rather than directly on h_{t+1} or z_{t+1} because h_t and z_t are internal model variables, not observation-grounded targets. Supervising one latent state with another would mainly impose self-consistency inside the model and could be satisfied by co-adapting both sides. In contrast, $e^\star_{t+1}$ is grounded in the next observation while being stop-gradient, providing an observation-grounded predictive target without adding a pixel decoder. The target uses the same trainable encoder: stop-gradient blocks this loss path, but does not fix the encoder across updates or prevent co-adaptation through other paths.

Importantly, the deterministic state is still central to the bottleneck. The predictor consumes the full latent history,

$$T_\theta(s_{\le t}, a_{\le t}) = T_\theta(h_{\le t}, z_{\le t}, a_{\le t}),$$

so the alignment loss encourages future-relevant information in the latent history. Direct predictor access to earlier states need not force the current RSSM state alone to retain that information. Gradients flow through the prediction branch into the predictor, RSSM, and the encoder features used to infer the context latents $s_{\le t}$, but not through the target branch $e^\star_{t+1}$. Thus, the encoder is shaped to provide information that helps the RSSM construct a predictive latent history.

E.2 Additional related work

Predictive representation learning and collapse prevention Predicting learned representations is a common alternative to pixel reconstruction in self-supervised learning. CPC (van den Oord et al., 2018) predicts future latent targets contrastively, while masked-prediction and joint-embedding methods such as data2vec (Baevski et al., 2022), I-JEPA (Assran et al., 2023), and next-embedding prediction approaches operate directly (Xu et al., 2025) in representation space. These methods, however, are typically developed outside control-oriented world modeling or are not tied to training a recurrent latent state for imagination-based control. A central challenge in decoder-free learning is avoiding representational collapse. Contrastive methods use negative samples; bootstrapping and redundancy-reduction methods such as BYOL (Grill et al., 2020), SimSiam (Chen & He, 2021), Barlow Twins (Zbontar et al., 2021), and VICReg (Bardes et al., 2022) use stop-gradient targets, cross-correlation constraints, variance preservation, EMA or covariance regularization. In reinforcement learning, RAD (Laskin et al., 2020) and DrQ (Kostrikov et al., 2020) stabilize visual representations through augmentation, usually between paired views or same-timestep embeddings.