Rollout-Decoded Reconstruction for Latent World Models

Rishi Shah
Machine Learning Engineer
E3A Healthcare
rishishah994@gmail.com

Rishav Shrestha
Chief Technical Officer
E3A Healthcare
rishav@e3ahealth.com

Preprint. DOI: 10.5281/zenodo.21894547

Abstract

The decoder of a latent world model is trained on latents anchored to observations: encoder outputs, and the one-step predictions teacher forcing makes from them. At deployment it is applied to the model's own free-running rollout, which by then has run hundreds of steps since the last observation. We propose Rollout-Decoded Reconstruction (RDR), a single loss term that free-runs the model during training, decodes every rollout latent, and penalizes reconstruction error against ground truth. RDR adds no parameters and requires no architectural changes, and setting its weight to zero recovers the standard objective exactly, so every comparison is a one-flag A/B at an identical parameter count. On the chaotic Kuramoto–Sivashinsky equation, RDR raises valid prediction time from 3.87 to 6.97 time units, a 1.80× improvement at an identical 193,568 parameters, confirmed on seeds never used in selection and in 10 of 10 preregistered configurations at ratios of 1.71–2.50×. The improvement grows with latent width, from 1.41× at dimension 16 to 2.50× at dimension 48, while the standard objective loses horizon as capacity is added. In model-based control, ensembles of RDR world models reach useful behavior in fewer optimizer steps, winning 20 of 20 paired episodes at every reduced-data rung on two tasks, and are robust to a planner–training rollout mismatch that costs the standard objective more return in 4 of 4 measured rows. Both results come from the same single term. RDR is an objective rather than an architecture: it composes with any world model that retains a decoder.

1 Introduction

Latent world models learn three components: an encoder that maps observations to a compact state, a transition that advances the state, and a decoder that maps states back to observation space [9, 8, 10, 11]. The standard training objective supervises the decoder on posterior latents, states computed while observations are available, and on predictions one teacher-forced step from them. Multi-step training signals, where present, act in latent space only.

At deployment the model free-runs. The transition iterates on its own outputs, and the decoder is applied to latents that have drifted for tens or hundreds of steps without being conditioned on an observation. The decoder's training distribution and its deployment distribution therefore diverge exactly where long-horizon behavior is decided. The gap is named but unmeasured. PlaNet [9] identified the direct fix, observation overshooting, which decodes the multi-step rollout latents, and

set it aside as too expensive in image domains, so what it is worth was never established; the Dreamer line [8, 10, 11] trains its decoder on posterior latents only and never decodes an imagined state as a training loss. The gap has stood open in every latent world model since.

We propose Rollout-Decoded Reconstruction (RDR), a loss term that closes this gap directly. During training, the model is rolled free-running from an encoded initial state, exactly as evaluation will roll it; every rollout latent is decoded; and the reconstruction error against ground truth is penalized. The term adds no parameters, changes no architecture, and reduces to the standard objective when its weight λ is zero. Every comparison in this paper is therefore a controlled A/B at fixed data, seeds, budget, and parameter count, differing in one flag.

On the Kuramoto–Sivashinsky (KS) equation [14, 20], a chaotic PDE with exact ground truth and a standard long-horizon metric, RDR raises valid prediction time (VPT) from 3.87 ± 0.23 to 6.97 ± 0.42 time units, a 1.80× improvement at an identical 193,568 parameters, confirmed on fresh seeds and at both evaluation horizons; the direction holds in 10 of 10 preregistered configurations at ratios of 1.71–2.50× (Section 4.2). The effect is attributable to the decoder's training distribution rather than capacity: a variant that spends +25.7% more parameters on the same objective performs worse in 7 of 10 configurations, and the advantage grows with latent width while the standard objective anti-scales (Sections 4.3, 4.4). On two control tasks, ensembles of RDR world models reach useful behavior in fewer optimizer steps and are robust to a planner–training rollout mismatch; both properties are reported with the controls that discipline them (Section 5). At matched budget an observation-space predictor reaches the same horizon on this fully observed system, which bounds the bottleneck rather than the objective: the RDR contrast is within-latent throughout, and an observation-space predictor is unavailable in the settings latent world models exist to serve (Section 6.1).

RDR is a training objective, so it does not compete with the architectural work in this area. Symmetry reduction, discrete latents, KL balancing, and latent overshooting all leave a decoder in place and all leave its training distribution unchanged, and RDR applies on top of each. Any world model with an encoder, a transition, and a decoder can adopt it by adding one term and setting one weight.

Every quantitative claim traces to archived evaluation artifacts produced under a fixed, preregistered protocol; the figures are rendered by a script that re-asserts each quoted number against those artifacts and fails the build on drift.

2 Related Work

World-model decoders. The RSSM lineage [9, 8, 10, 11] trains encoder, latent transition, and decoder jointly, with the decoder supervised on posterior latents. Multi-step signals exist in latent space (latent overshooting in PlaNet, KL balancing in its successors), but the decode-the-rollout variant was considered by PlaNet and not pursued, and the Dreamer line does not revisit it. Decoder-free models remove reconstruction entirely, TD-MPC2 [12] by value prediction and MuDreamer [3] by deleting the reconstruction loss from Dreamer; this is the one family RDR does not reach, having removed the component it trains. Wherever a decoder is retained, it can be trained into a substantially better long-horizon instrument at zero parameter cost.

Training on model-generated outputs. The exposure-bias literature addresses train/deployment mismatch at the input: scheduled sampling [1] mixes model predictions into teacher-forced inputs, and professor forcing [15] aligns hidden-state distributions adversarially. In observation-space PDE surrogates, the pushforward trick [2] and solver-in-the-loop training [23] unroll the model and train on its own outputs, but these models have no latent space and hence no decoder subject to the mismatch. In video generation, Self Forcing [13], Diffusion Forcing [4], and Next Forcing [24] train autoregressive diffusion models on their own rollouts in pixel space, without an encoder–latent–decoder triplet. Two concurrent 2026 preprints are closest. NeuroWorld [6] decodes a frozen dynamics model's rollout latents with a separately trained subject decoder for fMRI prediction; Koopman Dreamer [16] includes an open-loop observation-prediction term among several auxiliaries around a spectrally constrained transition. Neither isolates the decoder's training distribution as the object of study.

Rollout training for latent reduced-order models. The same construction has been reached from the reduced-order modeling side. Rollout-LaSDI [22] adds a rollout term to latent space dynamics identification that encodes a state, integrates the latent dynamics to a sampled future time, decodes it, and penalizes field error against the reference solution, with gradients reaching the decoder; on a parameterized 2D Burgers family it cuts maximum relative error threefold against an otherwise identical model. The setting differs from ours in the ways that decide what the term is doing. Their latent evolution is a per-parameter linear system whose coefficients are fit by regression and interpolated across parameters, where ours is a learned nonlinear transition applied recursively to its own output, which is the regime in which the decoder's inputs drift off the distribution it was fit on. Their system is a smooth viscous PDE, where ours is chaotic and amplifies error exponentially. They report no action-conditioned or control result, and they hold the latent width fixed, so the capacity behavior of Section 4.4 has no counterpart. Rollout-decoded training is established for latent reduced-order models and, to our knowledge, absent from world models; this work carries it there and measures what it is worth.

Long-horizon forecasting of KS. We adopt the standard conventions: VPT is the time until normalized forecast error first crosses a threshold, and horizons are calibrated in Lyapunov times. For the L=22 domain we use the literature value $\lambda_{\max} \approx 0.043$ [7], so one Lyapunov time is 23.26 time units. The strongest published latent-space results on this domain, manifold learning with symmetry reduction [17], reach roughly 1.4–2.0 Lyapunov times, and full-state reservoir computers [18] go further. Both are architectural results and are orthogonal to the training objective studied here. The symmetry-reduced model is itself an encoder–dynamics–decoder triplet whose decoder is fit by reconstruction while its dynamics is learned separately in the reduced coordinates, so its decoder is subject to the same mismatch and RDR applies to it directly; the reservoir is full-state and has no latent to decode from. Our contribution is a training-objective effect measured within one architecture at fixed parameters, data, and budget, and Appendix C places every arm on the absolute scale.

3 Rollout-Decoded Reconstruction

3.1 Model and base objective

The model is a deliberately plain instance of the encoder, transition, decoder triplet. An MLP encoder E_ϕ maps a field snapshot $u_t \in \mathbb{R}^{64}$ to a latent $z_t \in \mathbb{R}^d$ (two hidden layers of 256, GELU). A state-space predictor f_θ, a four-layer S5 stack [21] with state size 64, advances the latent one step. A decoder D_ψ, an MLP with one hidden layer of 512, maps latents back to field space. A target encoder $\bar{E}$, the exponential moving average of E_ϕ with decay 0.999, provides latent regression targets; we write $\bar{z}_t = \bar{E}(u_t)$ and use *posterior*, by analogy with RSSM practice, for latents computed from observed data.

Four standard terms train the triplet. With sg the stop-gradient and $\hat{z}_{t+k}$ the free-running rollout ($\hat{z}_t = E_\phi(u_t)$, $\hat{z}_{t+k+1} = f_\theta(\hat{z}_{t+k})$, gradients flowing through the whole chain):

$$
\begin{aligned}
L_{\mathrm{TF}} &= \tfrac{1}{T}\textstyle\sum_t \left\| f_\theta(\bar{z}_t) - \mathrm{sg}(\bar{z}_{t+1}) \right\|^2 && \text{teacher-forced one-step latent prediction,} \\
L_{\mathrm{R}} &= \tfrac{1}{K}\textstyle\sum_{k=1}^{K} \left\| \hat{z}_{t+k} - \mathrm{sg}(\bar{z}_{t+k}) \right\|^2 && \text{multi-step latent rollout consistency,} \\
L_{\mathrm{OBS}} &= \tfrac{1}{T}\textstyle\sum_t \left\| D_\psi(f_\theta(\bar{z}_t)) - u_{t+1} \right\|^2 && \text{decode the teacher-forced prediction,} \\
L_{\mathrm{recon}} &= \tfrac{1}{T}\textstyle\sum_t \left\| D_\psi(E_\phi(u_t)) - u_t \right\|^2 && \text{reconstruction anchor on the online encoder.}
\end{aligned}
$$

In all four terms, the decoder is supervised only on posterior latents or on predictions one teacher-forced step from them; no term exposes it to a multi-step rollout latent.

3.2 The RDR objective

RDR decodes the same free-running rollout that L_{R} constrains, and that evaluation will score, against the ground-truth fields:

$$
L_{\mathrm{RDR}} = \frac{1}{K}\sum_{k=1}^{K} \left\| D_\psi(\hat{z}_{t+k}) - u_{t+k} \right\|^2, \qquad L = L_{\mathrm{TF}} + \alpha_e\, L_{\mathrm{R}} + L_{\mathrm{OBS}} + L_{\mathrm{recon}} + \lambda\, L_{\mathrm{RDR}}. \tag{1}
$$

The gradient reaches the decoder directly and, through the rollout chain, the predictor and the encoder, so all three components are shaped by the trajectory the model will actually produce. The curriculum is standard for rollout losses: α_e ramps linearly from 0 to 1 between epochs 2 and 5, the rollout branch (and with it the RDR term) opens after the two warm-up epochs, and the rollout is pure free-running throughout. Setting λ=0 recovers the baseline exactly; we call that arm *posterior-only*. The operating weight is λ=0.3, and the result is flat across $\lambda \in \{0.1, 0.3, 0.6, 1.0\}$ (Section 4.3).

3.3 A split-decoder variant

A single decoder serves two roles under RDR, sharp on-manifold reconstruction and robust off-manifold rollout decoding, and these could in principle conflict. We therefore also evaluate a variant that gives the rollout term its own decode head of identical shape, +25.7% total parameters, with two evaluation policies: the sharp head everywhere (variant B), or a ramp from sharp to rollout head across the horizon (variant C, fixed by preregistration as the primary arm before any result existed). Section 4.3 shows that at the operating point the split spends parameters without improving on the shared decoder, and that its usefulness depends on latent size.

4 Forecasting Experiments

4.1 Setup

We integrate KS on L=22 with 64 grid points by ETDRK4 at dt=0.1, discarding a 2000-step transient, and generate 512 training, 64 validation, and 64 test trajectories of 256 snapshots each from disjoint seeds. Training draws one fixed 160-snapshot window per trajectory and unrolls the rollout losses over K=128 steps; batch size 32, Adam at 3×10^{-4}, three seeds per arm. The headline configuration is latent 32, decoder width 512, λ=0.3, 320 epochs.

Evaluation free-runs each model from the first snapshot of every held-out trajectory through the same rollout path used in training, decodes, and scores normalized RMSE against the climatological standard deviation of the scored window. VPT is the time of the first crossing of 0.5, in time units (tu). The canonical horizon is 200 steps (20 tu); the long horizon is 1300 steps (130 tu) on separately generated, dealiased records. Appendix A gives the metric in full, the long-record generation procedure, the pinned-estimator construction behind the canonical/long pair, and a validation-split re-ranking confirming that the reported winner is selectable on validation.

4.2 Main result

Table 2 (Appendix B) and Figure 1 summarize the preregistered round. At fresh seeds (10–12), none used during selection, the posterior-only arm reaches 3.87 ± 0.23 tu and the RDR arm reaches 6.97 ± 0.42 tu: a 1.80× improvement at an identical 193,568 trainable parameters, both counts measured from the checkpoints with the EMA target encoder excluded identically. The same checkpoints score 3.77 vs. 6.90 tu on the long horizon, so the effect is not a short-horizon artifact, and Figure 2 shows the behavior on a single held-out trajectory. Across the ten preregistered rows, nine distinct configurations plus the winner's fresh-seed rerun, RDR wins 10 of 10 at capacity-matched ratios between 1.71× and 2.50×.

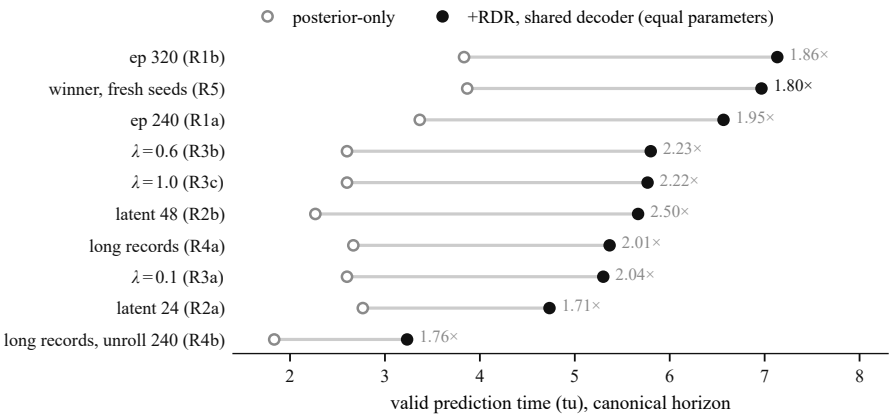

Figure 1: Every preregistered configuration, canonical horizon. Each row is one configuration scored at three seeds; open circles are the posterior-only arm, filled circles the capacity-matched +RDR arm, with the per-row ratio at right. RDR wins 10 of 10, ratio 1.71–2.50×. The fresh-seed confirmation row (R5) is the headline number; rows above it were selection rows.

The preregistered gate. The gate, locked before any sweep number existed, required the preregistered primary arm (variant C) to exceed 5.77 tu, the observation-space baseline measured at an earlier training budget. It does: 6.47 ± 0.40 tu at fresh seeds, every seed above the bar (per-seed 6.4/6.9/6.1); the selection-seed value is 6.90 ± 0.50 and is reported only under that label. The bar itself reproduces at 5.80 ± 0.44 on re-measurement (Appendix A). The bar was measured at a smaller training budget than the winner; Section 6.1 reports the matched-budget comparison, which does not affect the within-latent A/B.

Training-time scaling. Training length is the one lever that keeps paying: the primary arm moves from 5.27 to 6.37 to 6.90 tu across 160, 240, and 320 epochs at selection seeds, still unsaturated at the longest budget tested. Longer training is left to future work.

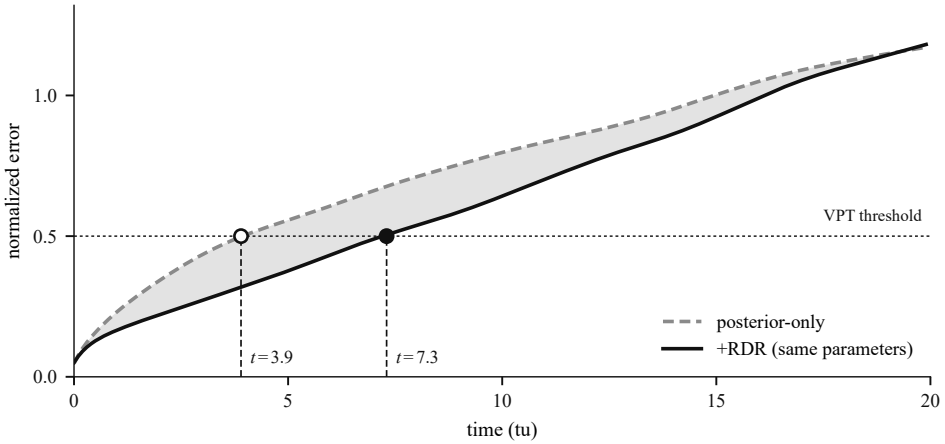

Figure 2: Forecast error against ground truth on one held-out KS trajectory, for two world models at an identical parameter count of 193,568 free-running from the same initial condition. RDR holds lower error at every horizon (shaded), and VPT is the first crossing of the 0.5 threshold: t=3.9 for the posterior-only arm against t=7.3 for RDR. Those crossings stand for an arm-level result: mean VPT over all 64 test trajectories at three fresh seeds is 3.87 ± 0.23 against 6.97 ± 0.42 tu, and Figure 1 gives every configuration.

4.3 Ablations: decoder sharing and the loss weight

Zero added parameters, full effect. The headline arm is the shared-decoder form of RDR: one decoder supervised on both posterior and rollout latents. It matches the posterior-only arm's parameter count exactly and is the strongest arm.

Added capacity does not reproduce the effect. The split-decoder variant (243,296 parameters against 193,568) performs worse than the shared decoder in 7 of 10 configurations on both horizons, including the winner and its fresh-seed confirmation (6.47 vs. 6.97). The supervision, rather than the capacity, carries the effect. The preregistration fixed variant C as the primary arm before these numbers existed, and variant C is the lower of the two RDR arms, so the headline gate was cleared by the conservative choice. The primary arm's own ratio, 1.67×, is capacity-confounded and is reported as such.

The split is latent-size dependent. At latent 128 with a linear decoder the head split is decisive: VPT 0.43 without it, 0.90 with it, under the sharp-head evaluation. At latent 32 with a 512-wide decoder the ordering reverses. When the latent is wide and the decoder thin, one head cannot be simultaneously sharp on-manifold and robust off-manifold; with a stronger decoder at moderate latent size, one head absorbs both roles and the split only spends parameters.

Insensitivity to the loss weight. The sweep over λ (0.1, 0.6, 1.0 against the default 0.3) keeps the capacity-matched ratio between 2.0× and 2.3×. Every tested weight roughly doubles VPT; the term has one hyperparameter and is insensitive to it at this operating point.

4.4 Scaling with latent width

Re-scoring earlier checkpoints under the final protocol completes a four-point latent bracket at one recipe (Figure 3; full numbers in Appendix B).

The posterior objective anti-scales; RDR converts capacity into horizon. As the latent grows from 16 to 48, the posterior-only arm's VPT falls at every rung, from 3.00 tu to 2.77, 2.60, and 2.27, in 3 of 3 seeds at every step. The RDR arm rises through 4.20, 4.73, and 5.90, then holds at 5.67. This is the mechanism restated: a wider latent has more directions the decoder never sees under posterior-only training, free-running rollouts wander into exactly those directions, and RDR trains the decoder there. RDR wins at every rung and in every seed; all 36 per-seed paired ratios exceed 1, with a minimum of 1.32 on the shared arm.

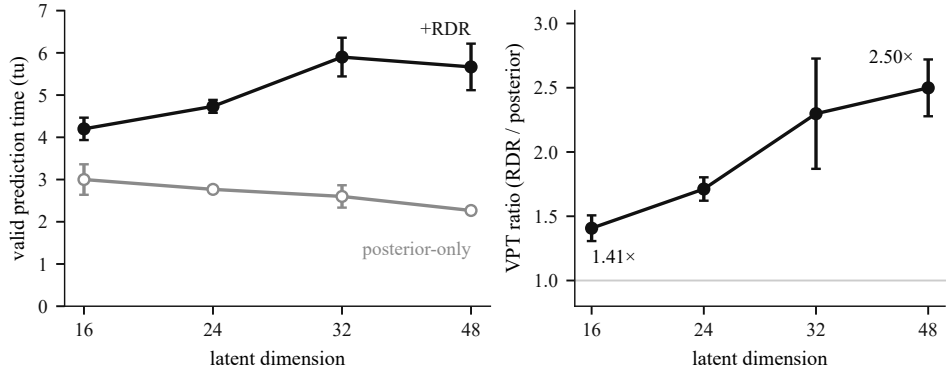

Figure 3: The latent-size bracket at the matched recipe (decoder 512, 160 epochs, λ=0.3, final protocol, three paired seeds per rung). Left: VPT by arm. The posterior-only arm *falls* as the latent widens, from 3.00 to 2.27 tu (3/3 seeds at every step), while RDR rises through 4.20, 4.73, and 5.90 tu, then 5.67 at latent 48 (that last step within noise). Right: the per-rung paired ratio rises from 1.41× to 2.50×; endpoints are separated by roughly 3.4 combined standard deviations, and the top step, 32 to 48, holds in only 2 of 3 seeds. Error bars are ±1 sd over seeds.

Calibrated strength. The ratio increases across the bracket: the 16-to-24 and 24-to-32 steps hold in 3 of 3 paired seeds, and the endpoints (1.41 ± 0.10 vs. 2.50 ± 0.22) are cleanly separated. The top step is noisy: 32 to 48 holds in 2 of 3 seeds and reverses under leave-one-seed-out. At the rung level, the level actually manipulated, n=4, and the observed ordering attains the smallest cluster-permutation p available, $2/24 \approx 0.083$; a per-seed Spearman correlation (0.907 over 12 points) overstates certainty if quoted alone and is not relied on. Three facts accompany the bracket. Predictor width covaries with latent size by construction, so the axis is model latent width rather than a pure bottleneck size; the

per-rung arm contrast is unaffected, since both arms share the width. The posterior arm falls below the persistence baseline (2.40 tu) at latent 48 in 3 of 3 seeds, so part of the top-rung ratio growth is baseline collapse rather than RDR gain. And VPT is quantized at 0.1 tu, so the smallest reported seed spreads are quantization-limited. The bracket is descriptive: selection seeds 0–2, post hoc, no preregistered gate. Long-horizon ratios track the canonical values throughout (1.40/1.74/2.30/2.46).

The rung nearest the system's inertial manifold, dimension near 8 for L=22 KS, is where RDR's edge is weakest, and the edge grows away from it. Latent ≈8 was not run; behavior below 16 is untested, and nothing in $[16, 48]$ suggests a turnover.

5 Control Experiments

We carry the objective to two classic control tasks, pendulum swing-up and a cartpole swing-up variant, with action-conditioned world models. Ensembles of ten models per arm drive a cross-entropy-method MPC planner [5, 19] with 128 sampled action sequences per step and no disagreement bonus, scored over 20 paired episodes with identical episode seeds for both arms. The planner rolls the world model with the same stateful rollout used in training; Section 5.2 additionally evaluates a per-step-reset variant. Two properties are established, each by a matched control.

5.1 Step-efficiency

We first vary the training-data budget. With training epochs held fixed as the dataset shrinks to 50/25/10%, RDR wins 20 of 20 paired episodes at every reduced rung on both tasks, with pendulum mean-return margins of +745, +1072, and +396 (Figure 4, left). Because fixed epochs give reduced rungs proportionally fewer optimizer steps, 480/240/60 against the full-data 960, this protocol varies data volume and optimization length together. We therefore pair it with a second ladder in which optimizer steps are matched at every rung. Under matched steps both arms train fully at every rung, the posterior arm at 25% exceeding its own full-data score, and the margins narrow to +3/+13/−18 (Figure 4, right). The pair locates the benefit precisely: **RDR reaches useful control in fewer optimizer steps, and given equal steps, reduced data limits neither arm.** The advantage is one of optimization rather than of data volume.

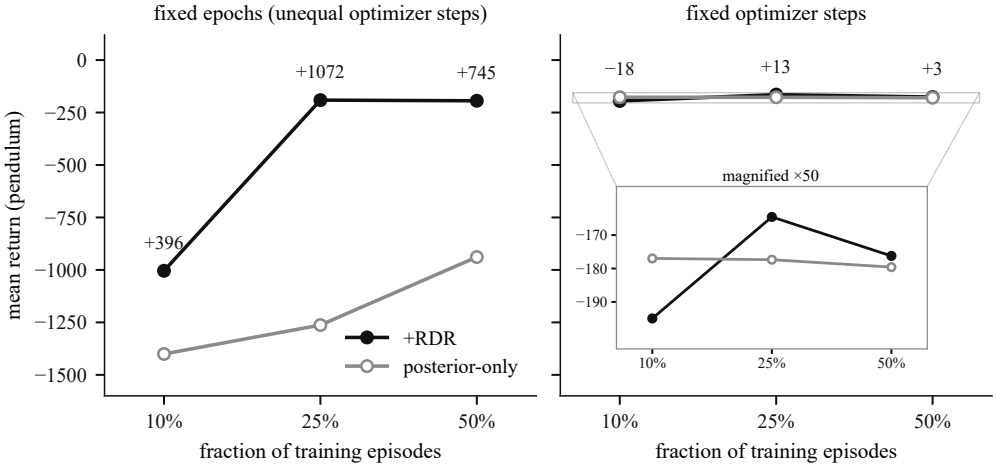

Figure 4: The reduced-data ladder on pendulum, mean return over 20 paired episodes, annotations giving the RDR-minus-posterior margin at each rung. Left: epochs held fixed, so reduced rungs receive proportionally fewer optimizer steps (480/240/60 at the 50/25/10% rungs against the full-data 960). RDR reaches useful control at every rung, 20/20 paired wins per rung on both tasks, and the posterior arm degrades badly. Right: the same recipe with optimizer steps matched at every rung. The margins narrow (+745/+1072/+396 become +3/+13/−18), and on cartpole the sign reverses at two rungs (6/20 paired wins at 25% and 50%). At the shared scale the two arms coincide; the inset magnifies the marked sliver, across which the entire spread between them is about 30 return.

5.2 Robustness to planner–training rollout mismatch

A deployed planner can roll the world model with an implementation that differs from the one used in training; a common variant re-initializes the recurrent state at each planning step rather than carrying it. We evaluate both arms under the stateful rollout, which matches training, and under the per-step-reset variant, on both tasks and both seed sets. Moving from the per-step-reset to the stateful planner improves both arms in all four measured rows, and the posterior arm gains more in every row: +9.0, +18.1, +17.1, +12.7 return against RDR's +1.8, +3.6, +12.8, +10.6 (Table 5). The asymmetry follows from the objective. RDR trains its decoder on free-running rollouts and is largely indifferent to the planner's rollout function; the posterior-only decoder, trained exclusively on encoder posteriors and teacher-forced latents, absorbs the mismatch as lost return. **RDR is robust to planner–training rollout mismatch**, an implementation class that is easy to ship and that the standard objective silently converts into lost return. Under the stateful planner the arms perform equivalently, margins +0.61 and +1.24 on the primary rows, which bounds the property: RDR insulates against mismatch rather than planning better under matched conditions.

5.3 Scope of the oracle comparison

Under the stateful planner, ensemble MPC over the learned world models matches planning with the true dynamics at the same search budget: on pendulum the RDR ensemble returns −177.45 against the oracle planner's −183.67, on cartpole 158.57 against 151.01. The posterior-only ensemble shares the property (−178.06 and 157.33 on the same rows). This is a statement about ensembles of learned models, quoted as scope for the control experiments, and not an RDR result.

6 Scope

6.1 Comparison to observation-space prediction

A fair reading of the main result requires the strongest non-latent control. An observation-space pushforward predictor [2], trained on its own unrolled outputs at the winner's budget, reaches 7.00 ± 0.26 tu (Table 1): parity with the capacity-matched RDR arm, every gap far inside one seed-level sd, and ahead of the preregistered primary arm. Both model classes scale with training at nearly the same rate, the latent stack from 5.27 to 6.90 against observation-space from 5.80 to 7.00 over the same budget change, which is why the 5.77 bar, measured at the smaller budget, is not the matched comparison. On a fully observed, 64-dimensional system, the latent bottleneck buys nothing over a direct observation-space predictor at matched budget.

This result bounds the bottleneck rather than the objective. The RDR contrast is within-latent throughout, and every arm in Table 1 that carries a latent is unaffected by the class comparison. It does establish what this system can and cannot demonstrate: a 64-dimensional fully observed field is small enough to predict directly, so the architecture RDR improves is not the one a practitioner would choose here. The settings that force a latent, partial observability, pixel observations, or a planner that requires a compact state, are also the settings in which an observation-space predictor is unavailable. Measuring the effect in one of those is the natural next experiment.

Table 1: Model classes at the winner's training budget (320 epochs), canonical and long horizons, mean ± sd over three seeds. The pushforward baseline sits between the two latent arms in parameter count. The within-latent RDR effect (rows 1, 2 vs. row 4) is untouched by this comparison. On the class comparison, the headline arm reaches parity with the pushforward baseline (row 1 vs. row 3, a gap of 0.06 combined sd), while the preregistered primary arm sits below it (row 2 vs. row 3, 1.11 combined sd).

model	params	VPT canonical (tu)	VPT long (tu)
latent + RDR, shared decoder (headline)	193,568	6.97 ± 0.42	6.90 ± 0.44
latent + RDR, split head (preregistered primary)	243,296	6.47 ± 0.40	6.40 ± 0.40
observation-space pushforward	230,464	7.00 ± 0.26	6.90 ± 0.26
latent, posterior-only	193,568	3.87 ± 0.23	3.77 ± 0.23

6.2 Limitations

The forecasting claim rests on one dynamical system at one domain size, three seeds per cell, and a metric quantized at 0.1 tu; the uniform 10-of-10 direction and the fresh-seed confirmation mitigate but do not remove this. Configuration selection during the sweep compared results on the test split; validation-split re-ranking selects the same winner with rank agreement $\rho = 0.979$ (Appendix A), and the headline is in any case reported at fresh seeds never used in selection. The control results are two low-dimensional classic tasks. Absolute horizons for every arm, the published results on this domain, and the caveats that cross-class comparison requires are given in Appendix C. The case for the latent stack over observation-space training is not made on this system; RDR is measured within-latent throughout, and the settings that make a latent necessary are left to future work.

7 Conclusion

A latent world model's decoder is deployed on the model's own free-running rollout and trained, almost universally, on something else. Rollout-Decoded Reconstruction closes that gap with one loss term and no new parameters. On a chaotic PDE it nearly doubles valid prediction time at an identical parameter count, in 10 of 10 preregistered configurations, with an advantage that grows as the latent widens while the standard objective anti-scales. In control it yields step-efficiency and robustness to rollout mismatch, each established by a matched control. On a fully observed low-dimensional system an observation-space predictor reaches parity at matched budget, which marks where a latent bottleneck earns its place and leaves the within-latent effect standing.

The architectures this reaches are already in use. Every model in the Dreamer lineage trains its decoder on posterior latents and free-runs it at inference, and adopting RDR in any of them means adding one term and choosing one weight; on the system measured here that is worth almost a doubling of usable horizon. The scaling behavior indicates where it repays most: over the range measured, the wider the latent, the more the standard objective leaves on the table, and deployed world models run latents far wider than any rung tested here. Measuring the effect where a latent is required, under partial observability, pixel observations, or planning over compact states, is the direct next step.

A Evaluation Protocol Details

Metric. For each arm, the model free-runs from the first snapshot of each of the 64 held-out trajectories; the decoded rollout is scored by RMSE over (trajectory, space) at each horizon step, normalized by the climatological standard deviation of the scored window. VPT@0.5 is the time at which this normalized error first crosses 0.5, reported in time units on the 0.1-tu sampling grid; a curve that never crosses is right-censored at the horizon. Canonical horizon: 200 steps on the held-out test split. Long horizon: 1300 steps on 64 separately generated long records. The split-head ramp evaluation blends its two heads with a denominator pinned to 200 steps regardless of the evaluated horizon, so the canonical and long evaluations of one checkpoint are a single estimator scored at two horizons; the canonical/long pair is a censoring comparison on one pinned forecaster.

Long-record generation. Long-horizon ground truth is generated with 2/3-rule dealiasing and a per-step Hermitian projection, and the climatological normalization is computed on the scored window.

Validation-split re-ranking. All sweep configurations were re-scored on the validation split (120 evaluations, no retraining; no trainer ever selected weights on validation). Validation selects the same winner, rank agreement with the test ranking is Spearman $\rho = 0.979$ over the nine configurations (0.985 with the fresh-seed rerun included), and the epoch-scaling verdict holds on validation (6.73 at 240 epochs rising to 7.13 at 320, the top two). The reported test numbers are therefore honest held-out estimates for a validation-selectable configuration.

The 5.77 bar. The preregistered gate's threshold of 5.77 tu re-measures at 5.80 ± 0.44 tu under the final protocol, a 0.03 tu match; at the winner's budget the same observation-space baseline reaches 7.00 ± 0.26 (Section 6.1). The locked threshold did not move.

Compute. All training ran on short-lived cloud GPU instances for a total of roughly $155; every evaluation and figure in this paper runs on a laptop CPU from the archived checkpoints.

B Full Numerical Results

Table 2: The preregistered sweep, all four arms, canonical horizon, VPT mean ± sd (tu) over three seeds (sd suppressed where the archived summary reports arm means only). Long-horizon columns track canonical throughout (every arm within 0.15 tu of its canonical value) and are omitted for width; the winner's long-horizon values appear in Section 4.2. Rows sorted by shared-arm VPT. R5 is the winner configuration at fresh seeds 10–12; all other rows are selection rows at seeds 0–2.

config	posterior	+RDR shared	+RDR split, sharp (B)	+RDR split, ramp (C)
R1b: 320 epochs	3.83 ± 0.21	7.13 ± 0.50	6.53	6.90 ± 0.50
R5: winner, fresh seeds	3.87 ± 0.23	6.97 ± 0.42	6.13	6.47 ± 0.40
R1a: 240 epochs	3.37	6.57 ± 0.71	6.10	6.37 ± 0.67
R3b: λ=0.6	2.60	5.80 ± 0.70	5.10	5.57 ± 0.76
R3c: λ=1.0	2.60	5.77 ± 0.15	5.43	6.07 ± 0.71
R2b: latent 48	2.27	5.67 ± 0.55	5.47	5.80 ± 1.35
R4a: long records	2.67	5.37 ± 0.12	4.57	4.73 ± 0.29
R3a: λ=0.1	2.60	5.30 ± 0.66	4.40	4.47 ± 0.40
R2a: latent 24	2.77	4.73 ± 0.15	4.53	4.73 ± 0.40
R4b: long records, unroll 240	1.83	3.23 ± 0.64	2.27	2.37 ± 0.15

The window dataset draws one fixed 160-of-256-snapshot window per trajectory for the entire run, so 37.5% of each training record is never seen at any epoch; this is why the longer-record rows (R4a, R4b) do not improve on their shorter-record counterparts.

Table 3: The latent-size bracket (Figure 3), canonical horizon, matched recipe (decoder 512, 160 epochs, λ=0.3), three paired seeds per rung. Ratios are per-seed paired means. All four rungs are scored under the final protocol at the same recipe.

latent	posterior	+RDR shared	+RDR split, ramp (C)	ratio shared/posterior
16	3.00 ± 0.36	4.20 ± 0.27	4.03 ± 0.21	1.41 ± 0.10
24	2.77 ± 0.06	4.73 ± 0.15	4.73 ± 0.40	1.71 ± 0.09
32	2.60 ± 0.27	5.90 ± 0.46	5.27 ± 0.40	2.30 ± 0.43
48	2.27 ± 0.06	5.67 ± 0.55	5.80 ± 1.35	2.50 ± 0.22

Table 4: The reduced-data ladder, both regimes, both tasks: mean return of each arm with the RDR-minus-posterior paired margin and RDR's paired win count over 20 episodes. Fixed-epoch rungs receive 480/240/60 optimizer steps as the data shrinks; step-matched rungs all receive the full-data 960 and were evaluated under the stateful planner, whose full-data rows appear in Table 5. Rungs draw episode windows independently, so smaller rungs are not strict subsets of larger ones.

regime	data	pend. post / +RDR	Δ (wins)	cartp. post / +RDR	Δ (wins)
fixed epochs	10%	−1400 / −1004	+396 (20/20)	−28 / +21	+48 (20/20)
	25%	−1263 / −191	+1072 (20/20)	−21 / +110	+132 (20/20)
	50%	−939 / −194	+745 (20/20)	+49 / +114	+64 (20/20)
fixed steps	10%	−177.0 / −194.9	−18.0 (12/20)	90.3 / 95.6	+5.4 (17/20)
	25%	−177.4 / −164.6	+12.8 (12/20)	122.9 / 115.6	−7.4 (6/20)
	50%	−179.6 / −176.2	+3.3 (18/20)	129.2 / 124.8	−4.3 (6/20)

Table 5: The planner rollout comparison, all four measured rows: mean return per arm under the per-step-reset and stateful planner rollouts, with the paired margin and RDR's win count over 20 episodes. The stateful rollout matches the one used in training. The last two columns give each arm's gain from moving to the stateful planner: the posterior arm gains more in 4 of 4 rows.

row	planner	posterior	+RDR	Δ (wins)	post. gain	RDR gain
pendulum, seeds 0–19	per-step-reset	−187.10	−179.27	+7.83 (19/20)		
	stateful	−178.06	−177.45	+0.61 (9/20)	+9.04	+1.82
pendulum, seeds 100–119	per-step-reset	−263.02	−247.28	+15.74 (18/20)		
	stateful	−244.90	−243.69	+1.22 (13/20)	+18.12	+3.59
cartpole, seeds 0–19	per-step-reset	140.25	145.77	+5.52 (15/20)		
	stateful	157.33	158.57	+1.24 (11/20)	+17.08	+12.80
cartpole, seeds 100–119	per-step-reset	141.32	142.81	+1.49 (17/20)		
	stateful	154.03	153.45	−0.59 (13/20)	+12.71	+10.64

Parameter accounting. All counts exclude the 90,656-parameter EMA target encoder, a non-gradient shadow copy unused at evaluation, from every arm identically (full checkpoint totals: 284,224 shared/posterior, 333,952 split). The observation-space pushforward's 230,464 sits between the two latent arms' counts, so the class comparison of Table 1 is not a capacity artifact in either direction.

C The Absolute Scale

This appendix places every arm on the absolute VPT scale, so that the within-architecture effects reported above can be read against the published results on this domain. The comparison classes differ in structure and in evaluation protocol, and the caveats following the table are load-bearing.

Table 6: Every arm on the absolute VPT scale for KS at L=22, with $\lambda_{\max}$=0.043 [7] (one Lyapunov time = 23.26 tu). The published band and the reservoir row are different settings and carry the caveats below; neither is class-comparable to the latent arms.

tier	VPT (tu)	Λt	status
persistence baseline	2.4	0.10	
latent, posterior-only (winner config)	3.87	0.17	measured, fresh seeds
observation-space pushforward, small budget	5.77	0.25	the preregistered gate's bar
latent + RDR, split head (primary arm)	6.47	0.28	fresh-seed confirmed
latent + RDR, shared decoder (headline)	6.97	0.30	fresh-seed confirmed
observation-space pushforward, matched budget	7.00	0.30	Section 6.1
published latent-ROM band [17]	33–47	1.4–2.0	symmetry-reduced
full-state reservoir (ESN), this work	73.0	3.14	see caveats

The published band uses symmetry reduction (the slice trick), an architectural technique our models do not employ. It is orthogonal to the objective studied here: that model is itself an encoder–dynamics–decoder triplet whose decoder is fit by reconstruction, so RDR composes with it rather than competing against it. The ESN, implemented to calibrate what this data and metric support, is full-state with no latent bottleneck, receives a 50-snapshot teacher-forced spin-up that the latent arms do not (their forecasts cold-start from a single snapshot), and is retuned per horizon; the canonical-horizon tuning right-censors at the 20-tu window, and the quoted 73.0 comes from the long-horizon tuning. It demonstrates that the data and metric support far longer horizons than any bottlenecked arm here reaches, and it is never class-compared against them.

References

[1] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015. arXiv:1506.03099.

[2] Johannes Brandstetter, Daniel Worrall, and Max Welling. Message passing neural PDE solvers. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2202.03376.

[3] Maxime Burchi and Radu Timofte. MuDreamer: Learning predictive world models without reconstruction. *arXiv preprint arXiv:2405.15083*, 2024.

[4] Boyuan Chen, Diego Martí Monsó, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2407.01392.

[5] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. arXiv:1805.12114.

[6] Zijian Dong, Jianxiong Zhou, Kwun Kei Ng, Jan Paolo Macapinlac Balagtas, Zhizhou Li, Zijiao Chen, and Juan Helen Zhou. NeuroWorld: A latent brain world model for stimulus-conditioned human brain dynamics. *arXiv preprint arXiv:2608.01773*, 2026.

[7] Russell A. Edson, Judith E. Bunder, Trent W. Mattner, and Anthony J. Roberts. Lyapunov exponents of the Kuramoto–Sivashinsky PDE. *The ANZIAM Journal*, 61(3):270–285, 2019. arXiv:1902.09651.

[8] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1912.01603.

[9] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning (ICML)*, 2019. arXiv:1811.04551.

[10] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering Atari with discrete world models. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2010.02193.

[11] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640:647–653, 2025. arXiv:2301.04104.

[12] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.16828.

[13] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2506.08009.

[14] Yoshiki Kuramoto and Toshio Tsuzuki. Persistent propagation of concentration waves in dissipative media far from thermal equilibrium. *Progress of Theoretical Physics*, 55(2):356–369, 1976.

[15] Alex Lamb, Anirudh Goyal, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. Professor forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016. arXiv:1610.09038.

[16] Jiaqi Li, Xinglong Zhang, Haibin Xie, Yixing Lan, Wei Pan, and Xin Xu. Koopman dreamer: Spectrally constrained latent dynamics for stable world-model imagination. *arXiv preprint arXiv:2607.19719*, 2026.

[17] Alec J. Linot and Michael D. Graham. Deep learning to discover and predict dynamics on an inertial manifold. *Physical Review E*, 101(6):062209, 2020. arXiv:2001.04263.

[18] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Physical Review Letters*, 120:024102, 2018.

[19] Cristina Pinneri, Shambhuraj Sawant, Sebastian Blaes, Jan Achterhold, Joerg Stueckler, Michal Rolinek, and Georg Martius. Sample-efficient cross-entropy method for real-time planning. In *Conference on Robot Learning (CoRL)*, 2020. arXiv:2008.06389.

[20] Gregory I. Sivashinsky. Nonlinear analysis of hydrodynamic instability in laminar flames—I. Derivation of basic equations. *Acta Astronautica*, 4(11–12):1177–1206, 1977.

[21] Jimmy T. H. Smith, Andrew Warrington, and Scott W. Linderman. Simplified state space layers for sequence modeling. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2208.04933.

[22] Robert Stephany and Youngsoo Choi. Rollout-LaSDI: Enhancing the long-term accuracy of latent space dynamics. In *NeurIPS Workshop on Machine Learning and the Physical Sciences*, 2025. arXiv:2509.08191.

[23] Kiwon Um, Robert Brand, Yun Fei, Philipp Holl, and Nils Thuerey. Solver-in-the-loop: Learning from differentiable physics to interact with iterative PDE-solvers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. arXiv:2007.00016.

[24] Gangwei Xu, Qihang Zhang, Jiaming Zhou, Xing Zhu, Yujun Shen, Xin Yang, and Yinghao Xu. Next forcing: Causal world modeling with multi-chunk prediction. *arXiv preprint arXiv:2606.11187*, 2026.