The Content–Geometry Tradeoff in World Models: When JEPA Learns Latent Maps

Mengye Ren[1], Ying Wang[*1], Christopher Hoang[*1], Yilun Kuang[1], and Yann LeCun[12]
[1]New York University [2]AMI Labs
{mengye,yw3076,ch3451,yk2516,yann.lecun}@nyu.edu
https://agenticlearning.ai/content-geometry-tradeoff/

Abstract

A latent world model for planning needs two properties: latent distances that smoothly track progress toward a goal (*geometry*), and task-relevant factors that remain decodable from the latent representation (*content*). Common approaches prevent representation collapse with a stop-gradient (SG) target or a distributional regularizer, but we find that both often yield poor latent geometry. In this paper, we introduce sliced variance regularization (SVReg), which imposes a variance floor on latent projections along random directions. SVReg learns a smooth map of the agent's state. Its latent trajectories are smooth, and its latent distances track progress in navigation and reaching. These geometric benefits, however, can come with a tradeoff in content. SVReg can discard slowly changing, hard-to-predict factors, such as the pose of a manipulated object, that SG variants tend to preserve. To address this tradeoff, we propose Slow SVReg, which favors slowly changing factors with a slowness term and a gated residual predictor. It keeps these factors and improves planning. This simple recipe has the highest planning success in the hard settings of five visual control tasks, and it learns latent maps from first-person mazes and city walks.

1 Introduction

An autonomous agent inhabits a temporal world, observing a stream of high-dimensional sensory data from which it builds an internal model of how the world evolves. Learning latent dynamics from sequential observations is a natural path to such world models. An encoder compresses observations into a compact latent space, and a predictor captures temporal transitions, enabling planning and control without operating in the observation space. The Joint Embedding Predictive Architecture (JEPA) (LeCun, 2022) instantiates this idea. The JEPA objective alone, however, admits a trivial solution where the encoder maps everything to a constant. A popular fix today in modern self-supervised learning (SSL) is the *stop-gradient* trick, which passes the prediction target through the encoder but blocks gradients on that branch. This is used by many JEPA implementations (Assran et al., 2023; Bardes et al., 2024), and the closely related image SSL methods (Grill et al., 2020; Chen and He, 2021). Another alternative to stop-gradient is distributional regularizers (Zbontar et al., 2021; Bardes et al., 2022; Balestriero and LeCun, 2025). While the choice between the two is usually made for training stability, it also decides the structure of latent rep-

Agentic Learning AI Lab Technical Report AGL-TR-2026-04. doi:10.5281/zenodo.24356797
[*] Equal contribution.

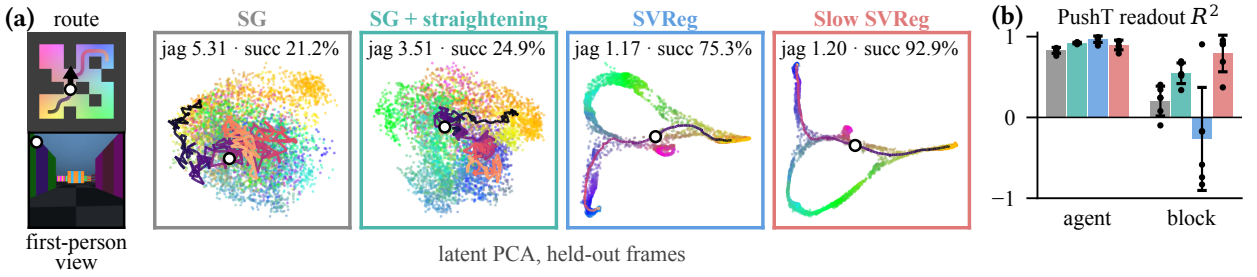


Figure 1: ***SVReg learns a smooth map (geometry) but can lose the pushed block (content), and Slow SVReg keeps both.*** *(a) One first-person route through the Medium maze in each latent, with the first-person view at the marked point. "jag" = the route's jaggedness (1 is smooth) and "succ" = planning success on hard goals. (b) Held-out readout R^2 of the PushT agent and resting block, colored as in (a) (mean over 6 seeds).*

resentations. A planner that searches in latent space (Zhou et al., 2025; Wang et al., 2026b; Maes et al., 2026) requires both *geometry*, where latent distance smoothly tracks progress toward the goal, and *content*, where task-relevant factors, such as an object to be manipulated, stay decodable.

We find that many common choices lead to poor geometry. With stop-gradient (SG), the target is held fixed during each update, so the encoder is trained only to predict its current latents. This one-sided objective does not ask consecutive latents to lie on a smooth path. In our experiments, a smooth route often appears jagged in the SG latent. Latent distance then becomes a poor guide toward far goals, and SG falls behind on hard maze goals. The one-sided objective is also fragile when data are scarce. In contrast to SG, common distributional regularizers fall short in a different way. VICReg and SIGReg were designed for i.i.d. multi-view batches and constrain the whole latent distribution.

In contrast, a regularizer with fewer constraints yields a better latent map. In this paper, we introduce sliced variance regularization (SVReg), a one-sided floor on the spread of the latent along random directions, trained without SG. The target then moves with the encoder, so the whole latent is trained to be predictable from its past, and directions the predictor cannot explain, such as noise, are suppressed. In our experiments, a smooth route then stays smooth in the latent. The resulting latent is a smooth map of the agent's state. Its distances track progress in navigation and reaching, and in first-person mazes its leading principal components trace the agent's position.

However, the predictability pressure can erase task-relevant content. Factors are penalized for their unpredictable variance, so slow, hard-to-predict factors, such as manipulated objects, can be lost. In comparison, SG variants tend to keep these factors, but at the cost of the jagged geometry described above. Together, the two failures form what we call the *content–geometry tradeoff*.

To achieve the best of both worlds, we propose Slow SVReg. It adds 1) a *slowness* term, which encourages a small portion of the latent to change slowly across frames, and 2) a *gated residual predictor*, which copies the current latent by default and predicts the change. Both favor slowly changing factors. Slow SVReg keeps these factors in manipulation and improves planning, especially in first-person mazes, where the agent's position must be carried across frames. Slow SVReg plans best on hard maze goals, Reacher, Cube and PushT. Moreover, a model trained on Wanderland city walk scans (Liu et al., 2026) learns a latent map of the streets from first-person video and actions.

Our contributions are as follows. First, we show that SG and common distributional regularizers learn poor latent geometry for planning, while SVReg learns smooth latent maps in navigation and reaching. Second, we identify the content–geometry tradeoff, analyze its direction in the linear encoder objective,

show how modes compete for latent capacity in controlled experiments on latent linear dynamics, and find the same pattern in visual control. Finally, we introduce the slowness regularizer and the gated residual predictor, which favor slowly changing factors and keep content more reliably. The resulting model performs strongly across navigation, mapping, reaching, and manipulation.

2 Analysis

This section explains where the content–geometry tradeoff comes from and motivates our main method in Section 3, through the lens of linear latent dynamics. Without stop-gradient, the encoder is penalized for unpredictable variance (Section 2.1). This suppresses noise, but it also shrinks hard-to-predict factors, which can then be dropped when the latent has too few dimensions to hold every factor (Section 2.3). With stop-gradient, finite data can lead the encoder to read noise while its latent statistics look healthy (Section 2.2).

We analyze T observations with a linear encoder $\Phi(\boldsymbol{o}) = W\boldsymbol{o}$ and an action-conditioned linear predictor $\hat{\boldsymbol{z}}_{t+1} = A\boldsymbol{z}_t + B\boldsymbol{a}_t$, where $\boldsymbol{z}_t = \Phi(\boldsymbol{o}_t)$. The JEPA loss (LeCun, 2022) is

$$\mathcal{L} = \frac{1}{T-1}\sum_{t=0}^{T-2}\left\|A\boldsymbol{z}_t + B\boldsymbol{a}_t - \boldsymbol{z}_{t+1}\right\|^2.$$

This loss admits a trivial collapse in which Φ maps every observation to the same constant ($\Phi \equiv 0$ for the linear encoder). The remedies are a stop-gradient on the target (Assran et al., 2023; Bardes et al., 2024), or a regularizer on the latent distribution (Zbontar et al., 2021; Bardes et al., 2022; Balestriero and LeCun, 2025; Kuang et al., 2026).

2.1 Two JEPA Objectives

For mean-centered observations, define the lag-0 ($\hat{\Sigma}_{00}$), next-step ($\hat{\Sigma}_{11}$), and lag-1 ($\hat{\Sigma}_{01}$) sample covariances

$$\hat{\Sigma}_{00} = \frac{\sum_t \boldsymbol{o}_t\boldsymbol{o}_t^\top}{T-1}, \quad \hat{\Sigma}_{11} = \frac{\sum_t \boldsymbol{o}_{t+1}\boldsymbol{o}_{t+1}^\top}{T-1}, \quad \hat{\Sigma}_{01} = \frac{\sum_t \boldsymbol{o}_t\boldsymbol{o}_{t+1}^\top}{T-1},$$

with sums over $t = 0, \ldots, T-2$, latent statistics $\Sigma_z = W\hat{\Sigma}_{00}W^\top$ and $C_z = W\hat{\Sigma}_{01}W^\top$, and Σ_z invertible (no latent direction is constant).

Theorem 2.1 (Encoder objective after eliminating the predictor). *For a linear encoder $\Phi(o) = Wo$ and a predictor without the action term, substituting the optimal linear predictor gives:*

1. ***Without stop-gradient**, $A^\star = C_z^\top\Sigma_z^{-1}$ and*

$$\mathcal{L}^\star_{\text{no-sg}}(W) \;=\; \operatorname{tr}\!\left(W\hat{\Sigma}_{11}W^\top\right) \;-\; \operatorname{tr}\!\left(C_z^\top\Sigma_z^{-1}C_z\right).$$

2. ***With stop-gradient and a frozen target** W_{tgt}, writing $\hat{\Sigma}_{zz'} = W\hat{\Sigma}_{01}W_{\text{tgt}}^\top$, $A^\star = \hat{\Sigma}_{zz'}^\top\Sigma_z^{-1}$ and*

$$\mathcal{L}^\star_{\text{sg}}(W;\, W_{\text{tgt}}) \;=\; \operatorname{tr}\!\left(W_{\text{tgt}}\hat{\Sigma}_{11}W_{\text{tgt}}^\top\right) \;-\; \operatorname{tr}\!\left(\hat{\Sigma}_{zz'}^\top\Sigma_z^{-1}\hat{\Sigma}_{zz'}\right).$$

The proof is in Section A, and Section A.3 extends the theorem to the action-conditioned predictor. Without stop-gradient, the loss also penalizes unexplained target variance. Removing unpredictable directions, including observation noise, therefore reduces the loss and can give a smoothly moving state a smooth latent path. With stop-gradient (SG), target variance is constant during an update, so noise directions are unpenalized and perfectly predictable rotations allow curved paths even at zero prediction error.

2.2 Finite Data

Finite-sample noise has a nonzero lag-1 cross-covariance, of order $\sigma^2\sqrt{d/T}$ for observation dimension $d \ll T$ and noise level σ. When this spurious lag structure exceeds the signal's, the fixed-target objective can select it, and the encoder keeps healthy latent variance while reading noise instead of state. We call this failure *signal collapse*. In-sample statistics cannot detect it, since the selected noise directions mimic signal lag correlations and a variance regularizer rescales them to meet its floor. A held-out state probe does detect it (Proposition A.1 in Section A.4).

2.3 Content

While good geometry captures latent dynamics, task-relevant content such as the state of a manipulated object can still be lost. Consider one scalar component of the environment state that evolves as $s_{t+1} = \varphi s_t + \kappa a_t + \epsilon_t$. Its persistence φ measures how much of it carries over to the next step, so a mode with φ near 1 changes slowly. Its one-step residual δ is the share of its variance that is not predicted by the current state and action. Without stop-gradient, a less predictable mode adds more prediction loss, and SVReg shrinks its latent coordinate.

The variance floor keeps the mode's scale away from zero, so shrinkage alone keeps the mode readable (Section A.5). Content is lost when modes compete for too few latent dimensions. We analyze this competition in the linear case, where two modes share one coordinate (Proposition A.2 in Section A.5). For persistent modes, with or without stop-gradient, a new mode enters the latent more easily when it is more persistent than the mode already there. Without stop-gradient, it also enters more easily when the action explains more of its variance, and the cost of one coordinate carrying two dynamics can protect the incumbent, so whichever factor fills the latent first can keep it. A manipulated object rests most of the time and moves only through contact, so the action explains little of its variance, and the latent can trade it for an action-driven factor such as the arm. SVReg loses the cube's height early in training, as this competition predicts (Figure 5b). In Section B we also test the competition of capacity with a linear experiment (Figure 7).

3 Learning Geometry and Content

In this section, we introduce Slow SVReg. It trains without stop-gradient with sliced variance regularization (SVReg) for geometry, and adds *slow channels* and a *gated residual predictor* to favor slowly changing factors (Figure 2). We also define measures of content and geometry. The baselines are described in Section 5.1.

Sliced variance regularization. VICReg and SIGReg (Section 5.1) were designed for i.i.d. multi-view batches and constrain the whole latent distribution, toward decorrelated coordinates or an isotropic Gaussian. World model data are temporally correlated trajectories whose states need not follow such a distribution, even with ample data, which motivates a less restrictive regularizer. Sliced variance regularization (SVReg) puts a one-sided floor on the standard deviation of the latent along random directions:

$$\mathcal{R}_{\text{SVReg}}(z) = \mathbb{E}_{\boldsymbol{u}}\left[\max(0,\ 1 - \text{std}_t(\boldsymbol{z}_t \cdot \boldsymbol{u}))^2\right], \qquad \boldsymbol{u} \sim \text{Unif}(\mathbb{S}^{r-1}), \tag{1}$$

where $\boldsymbol{z}_t \cdot \boldsymbol{u}$ projects each latent onto the direction $\boldsymbol{u}$ and std_t is taken over the estimation sample. The squared hinge is active only when the standard deviation along the direction falls below 1. Restricting $\boldsymbol{u}$ to the coordinate axes gives the element-wise floor, a squared form of the variance term of VICReg (Bardes et al., 2022). A latent can satisfy the element-wise floor while two coordinates copy each other, leaving no

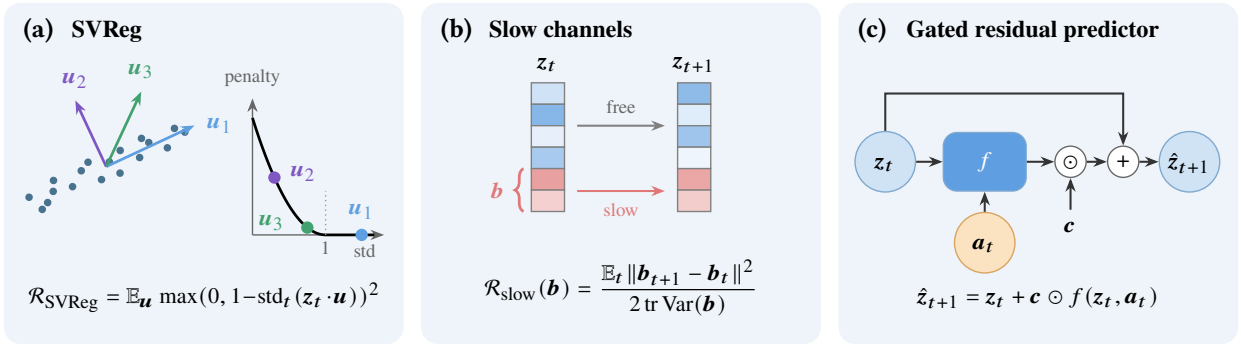


Figure 2: *The three components of Slow SVReg. (a) SVReg penalizes random directions* $\boldsymbol{u}$ *whose latent standard deviation* $\mathrm{std}_t(\boldsymbol{z}_t \cdot \boldsymbol{u})$ *falls below* 1 *(*$\boldsymbol{u}_2$*,* $\boldsymbol{u}_3$*). (b) The slow channels* $\boldsymbol{b}_t$*, a fixed set of latent dimensions (red), are penalized by the variance-normalized slowness term (Equation (2)). The other dimensions change freely. (c) The gated residual predictor copies* $\boldsymbol{z}_t$ *and adds the change predicted by* f *from the latents and the action* $\boldsymbol{a}_t$*, scaled by the learned gate* $\boldsymbol{c}$ *(Equation (3)).*

variance along their difference. SVReg also checks such mixed directions, without requiring the coordinates to be decorrelated. We add SVReg with weight λ to the JEPA prediction loss without stop-gradient, just like VICReg and SIGReg.

Slow channels. SVReg alone can lose content that rarely changes and moves only through contact, such as a resting cube. We therefore train a small fixed set of latent channels to change slowly between frames. For the slow channels $\boldsymbol{b}_t$, we add the slowness term with weight β,

$$\mathcal{R}_{\text{slow}}(\boldsymbol{b}) = \frac{\mathbb{E}_t \|\boldsymbol{b}_{t+1} - \boldsymbol{b}_t\|^2}{2 \operatorname{tr} \operatorname{Var}(\boldsymbol{b})}. \tag{2}$$

This is a variance-normalized form of the slowness objective of slow feature analysis (Wiskott and Sejnowski, 2002), so shrinking the slow channels cannot reduce the term, while SVReg covers the whole latent. It resembles the invariance term of VICReg applied to consecutive frames, but acts on a subset of channels and is normalized by their variance. Prior world models apply slowness along trajectories to the whole latent, in JEPA objectives such as PLDM (Sobal et al., 2025; Hauri et al., 2026) and as a regularizer that keeps consecutive latents close (Xia et al., 2026). The closest split designs constrain a fixed part of a representation (Garrido et al., 2023b; Denton and Birodkar, 2017).

Gated residual predictor. The gated residual predictor is a second way to favor slow factors. It copies the current latent by default and learns the change, so a factor that does not change between frames costs nothing to predict. The linear dynamics of Section 2.3 already have this form, $s_{t+1} = s_t + \big((\varphi - 1)s_t + \kappa a_t\big) + \epsilon_t$. For a persistent mode, φ near 1 makes the change without action small, as for a resting object or the agent's position between moves. We use

$$\hat{\boldsymbol{z}}_{t+1} = \boldsymbol{z}_t + \boldsymbol{c} \odot f(\boldsymbol{z}_{\leq t}, \boldsymbol{a}_t), \tag{3}$$

where $\boldsymbol{a}_t$ is the action and $\boldsymbol{c}$ is a learned per-dimension gate initialized at $c_0 = 0.1$. This has the same form as LayerScale (Touvron et al., 2021), which scales residual branches across depth, whereas this gate scales the predicted change over time.

The identity path makes slowly changing factors cheap to predict, while the slow channels add direct pressure toward slow change. Together, they are designed to favor slow factors without SG, echoing SG's local preference among competing modes. We call this model Slow SVReg.

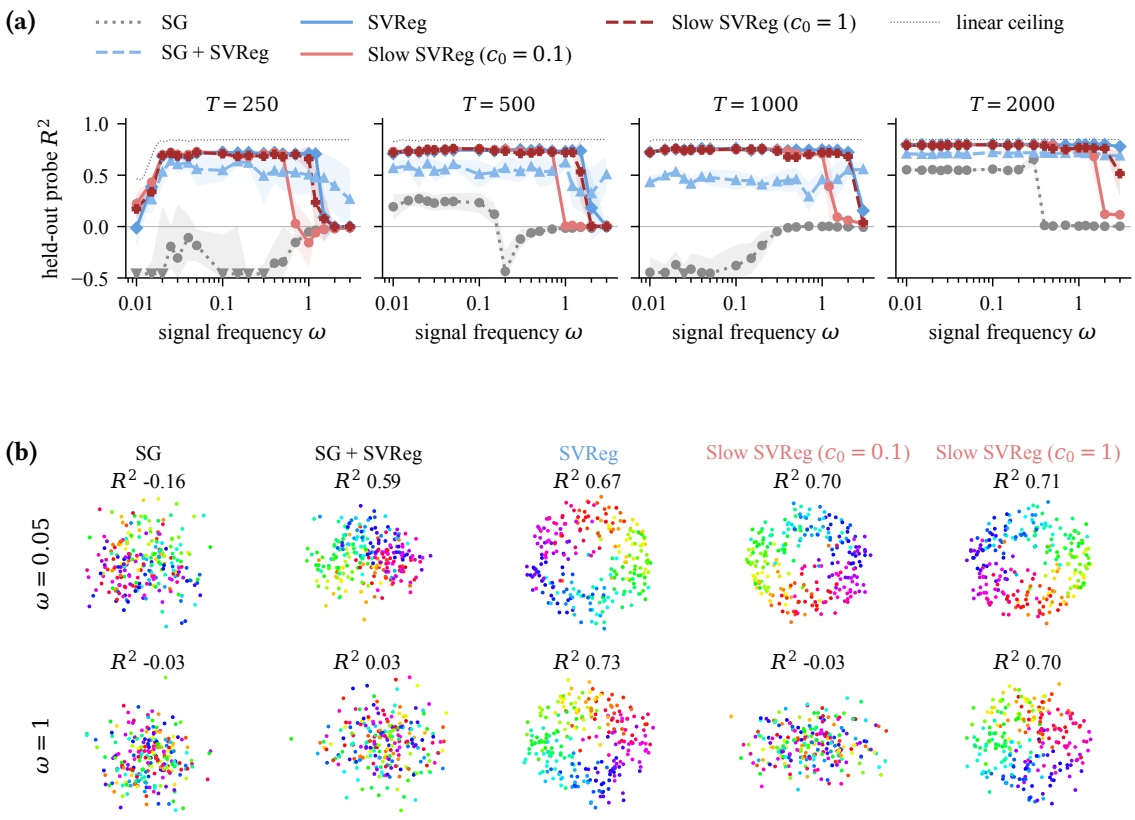


Figure 3: ***Linear dynamics: SVReg recovers slow and fast signals and keeps the circle structure.*** *(a) Held-out probe R^2 of the circle state against signal frequency at various trajectory length T (mean over 8 seeds). (b) Top two principal components of held-out latents (T=250, one seed), colored by circle phase. Protocol in Section B.*

Measuring content and geometry. We measure both properties on held-out data, separately from planning success. *Content* is the held-out R^2 of ridge readouts of state factors (agent position, joint angles, cube position and height, block pose), fit and scored on separate episodes, and the area under the ROC curve for binary factors such as gripper–cube contact (0.5 is chance). We measure two aspects of *geometry*, the ordering of latent distances and the smoothness of latent paths. The *distance rank correlation* ρ is the Spearman correlation between latent distance $\|z - z'\|$ and true task distance (shortest path in PointMaze, joint space in Reacher). In manipulation, *cost-to-goal* ρ correlates the latent L2 distance between two held-out frames (the planner's cost) with their true state distance (Section C). *Jaggedness* encodes a scripted smooth motion, such as a maze route or a full Reacher shoulder turn, and compares path lengths before and after a short moving average $\bar{z}_t$,

$$J = \frac{\sum_t \|z_{t+1} - z_t\|}{\sum_t \|\bar{z}_{t+1} - \bar{z}_t\|}, \tag{4}$$

so smooth paths give $J \approx 1$. The *participation ratio* (PR, *eff. dim* in figures and tables) counts the directions the latent effectively uses. With $\mu_1, \ldots, \mu_r$ the eigenvalues of the held-out latent covariance,

$$\mathrm{PR} = \frac{\left(\sum_i \mu_i\right)^2}{\sum_i \mu_i^2} \in [1, r]. \tag{5}$$

4 Noisy Linear Dynamics

We first test the geometry prediction of Section 2 in a linear world whose map is known. A two-dimensional state rotates at frequency ω and is observed through a random orthonormal map $U \in \mathbb{R}^{32\times 2}$ with Gaussian noise,

$$\boldsymbol{s}_{t+1} = R_\omega \boldsymbol{s}_t, \qquad \boldsymbol{o}_t = U\boldsymbol{s}_t + \sigma \boldsymbol{n}_t, \quad \boldsymbol{n}_t \sim \mathcal{N}(0, I),$$

where R_ω is the rotation by angle ω and $\sigma = 0.3$ (protocol in Section B). The circle is an easy task, since a PCA of the raw observations can recover the latent easily. The experiment is therefore a unit test that asks whether a JEPA objective can also find a latent linear map.

Interestingly, SG fails most at fast frequencies and short trajectories, as shown in Figure 3. By comparison, SVReg stays near the signal-recovery ceiling up to $\omega \approx 1$, and SG + SVReg lies between them. Slow SVReg matches SVReg on slow signals. With its default small gate (c_0=0.1) it loses the fastest signals (above $\omega = 0.5$ at T=250), and a gate of c_0=1 keeps them up to $\omega \approx 1$. In Figure 3, we show the different learned latent maps. Each encoder maps a loop, as an agent walking a closed route would. Without SG the map is a ring ordered by phase, while SG encodes scattered points without circular geometry.

The dependence on trajectory length follows the finite-data argument of Section 2.2. The spurious lag structure of the noise shrinks as $\sigma^2\sqrt{d/T}$, so a frozen-target encoder can select noise directions on short trajectories and needs longer ones to recover the circle. Without stop-gradient, SVReg penalizes unexplained target variance and so removes the noise directions at every length. The analysis does not predict the dependence on frequency, which we report as an empirical finding.

A second linear experiment tests content under limited capacity (Section B). An action-driven arm and an object that moves only when the arm touches it compete for a latent smaller than their joint state. SVReg keeps the arm with the best geometry but drops the object in about half the seeds (Figure 7). SG keeps the object at the cost of arm geometry, and SG + SVReg lies between them.

5 Visual Control

In this section, we test whether geometry and content decide planning success in visual goal-reaching with an action-conditioned JEPA. In navigation and reaching the agent moves only itself, and in manipulation it also moves an object.

5.1 Setup

Environments. We evaluate PointMaze U-maze and Medium (Fu et al., 2020), Reacher-Hard (Tassa et al., 2018), OGBench-Cube (Park et al., 2025) and PushT (Chi et al., 2025). Top-down PointMaze and Reacher train on one uniform-random exploration trajectory, first-person PointMaze on one trajectory of smooth random driving, Cube on 2,000 scripted-expert episodes and PushT on DINO-WM's offline dataset (Zhou et al., 2025).

Models and baselines. All models trained from scratch share a task-specific ResNet encoder and an action-conditioned transformer predictor (architectural details in Section C.2). VICReg (Bardes et al., 2022) combines element-wise variance and covariance penalties. As in other JEPA world models, it is paired with action-conditioned next-latent prediction, so on temporal data it has no term that keeps consecutive latents close. SIGReg, the sketched isotropic Gaussian regularizer (Balestriero and LeCun, 2025), matches Gaussianity along random slices. As in LeWM (Maes et al., 2026), we apply SIGReg after a BatchNorm MLP projector on the encoder output and call this SIGReg-BN. The projector is the only difference from plain

Table 1: *Planning success (%) in five visual control tasks.* *Mean ± s.d. over six training seeds.*

	Agent only: navigation and reaching					Object manipulation		
	U-maze (5k)		Medium (20k)		Reacher (5k)	Cube (80k)		PushT (470k)
steps × batch	10k × 32		20k × 32		50k × 32	50k × 64		50k × 32
Method	easy	hard	easy	hard		all	≥4 cm	
Stop-gradient								
SG	85.1±17.0	65.0±18.7	96.6±3.1	71.6±8.7	86.5±1.6	80.4±2.1	67.2±3.6	44.0±19.3
SG + straightening	86.3±5.5	88.0±3.1	96.9±1.6	72.8±5.6	86.6±2.9	85.4±1.0	75.6±1.6	62.1±7.7
SG + SVReg	96.3±1.6	95.8±1.1	96.0±1.5	62.7±6.2	96.3±1.6	77.2±1.3	61.7±2.3	76.9±4.3
No stop-gradient								
VICReg	**99.9±0.3**	94.3±2.3	96.4±0.8	77.2±2.3	93.1±1.7	69.6±5.8	49.8±9.3	55.0±3.6
SIGReg-BN	92.4±7.9	82.2±13.7	97.3±1.0	78.2±3.8	95.8±1.3	87.4±0.8	79.0±1.3	60.4±2.6
SVReg (ours)	99.7±0.4	96.4±1.3	**98.0±1.3**	81.3±2.3	98.7±0.9	76.9±8.8	62.9±13.1	75.0±8.6
Slow SVReg (ours)	99.7±0.4	**98.7±0.7**	97.9±0.8	**84.7±2.7**	**99.6±0.4**	**89.1±1.6**	**81.9±2.5**	**87.3±3.8**

SIGReg, and the other objectives act on the encoder output directly. LeWM computes the SIGReg statistics per frame, while LeJEPA pools them over the batch. To give SIGReg-BN its best chance, we ran both forms on every task and use the one that plans better (Section C). Per-frame statistics plan better on Cube, Reacher and the first-person mazes, and statistics pooled over the batch and time on the top-down mazes and PushT. Two hybrids test whether explicit regularization yields geometry with SG. *SG + SVReg* adds SVReg (Equation (1)), and *SG + straightening* adds the temporal-straightening curvature penalty (Wang et al., 2026b)

$$\mathcal{R}_{\text{curv}}(z) = \frac{1}{T-2}\sum_{t=0}^{T-3}\Big(1 - \cos\big(z_{t+1} - z_t,\ z_{t+2} - z_{t+1}\big)\Big) \tag{6}$$

over a training window of length T, which is 0 along a straight latent path.

Planning and selection. Cross-entropy method (CEM) model-predictive control minimizes latent distance from predicted states to the goal image and replans after a few steps, as in DINO-WM (Zhou et al., 2025). In the mazes, easy goals lie 25 steps from the start with no filter on their distance, and hard goals lie 50 steps away and at least 3 maze units apart. Cube is scored on all goals and on the goals that move the cube by at least 4 cm (≥4 cm, about 60% of goals). For each method, task and goal set we select one encoder learning rate and regularization weight, the setting with the highest mean success on a separate planning seed over all training seeds. We report that setting on three other planning seeds (Section C). Unless stated otherwise, tables report the mean and standard deviation over six training seeds at the selected setting. Bold marks the best value and underline the second best.

5.2 Main Results

Slow SVReg plans best on hard maze goals from both views, on Reacher, on both Cube protocols and on PushT (Tables 1 and 2). In the mazes, easy goals nearly saturate for the strongest methods, while hard goals lie 50 steps away along long routes and test whether latent distance tracks progress toward the goal.

Table 2: *Planning success (%) in first-person maze navigation.* *Mean ± s.d. over six training seeds. Setup in Section C.*

	U-maze (5k frames)		Medium (20k frames)	
Method	easy	hard	easy	hard
SG	97.2±1.1	42.0±6.8	98.4±0.9	21.2±6.4
SG + straightening	96.6±1.5	37.7±16.0	97.2±1.9	24.9±8.8
SG + SVReg	97.6±1.2	38.3±12.5	97.2±4.1	46.0±16.6
VICReg	94.2±1.2	40.3±4.4	95.0±1.0	29.6±7.3
SIGReg-BN	99.0±0.8	71.0±6.1	98.7±0.8	42.8±5.2
SVReg	98.2±0.8	90.4±1.6	97.0±0.7	75.3±5.5
Slow SVReg	**99.7**±0.6	**95.6**±1.4	**99.4**±0.5	**92.9**±3.2

SVReg alone shows where the two properties matter. It plans second only to Slow SVReg on hard goals in navigation and reaching, where only the agent moves, but it falls behind SIGReg-BN on Cube and SG + SVReg on PushT, with widely varying success. We trace its strength in navigation to geometry (Section 5.3) and its weakness with objects to lost content (Section 5.4).

In Table 2, we report planning in first-person view mazes, and the gaps widen. Here, the agent sees only the view ahead, so its position must be inferred from the views and actions. On hard goals SG drops sharply and the other baselines stay well below SVReg, while Slow SVReg nearly saturates on both mazes. Among the agent-only tasks, its gain over SVReg is largest here and small in the top-down mazes and Reacher, where each frame shows the agent.

SG + SVReg and SG + straightening add a geometry term to SG. They plan better than SG on hard U-maze goals and on PushT, but trail SVReg on hard Medium goals in both views and on first-person U-maze.

5.3 Geometry

Figure 4 plots the top two principal components of held-out latents in both mazes, colored by the agent's position. Without SG, SVReg and Slow SVReg show the maze layout in the PCA plot. Nearby positions are close in the representation space, and the corridors of Medium appear as separate branches. The layout emerges from first-person views as well, where the agent never sees its true position. SG keeps only a coarse ordering by position from above, and with first-person views its latents, like those of SG + straightening, mix positions from across the maze. The SVReg paths are smooth as well. A smooth shortest route stays smooth in most SVReg latents and zigzags under SG (Section D), and a full turn of the Reacher shoulder traces a smooth closed loop under SVReg and a jagged one under SG.

A latent map can be right globally and still be hard to plan with. The planner reaches a far goal by following latent distance over many short replanning steps, so it depends on the local shape of the latent and not only on its global layout. We therefore compare planning with two measures of geometry: the jaggedness of latent paths, a local measure, and the rank correlation between latent distance and maze distance over all frame pairs, a global measure. Jaggedness tracks planning closely. On U-maze, Medium and Reacher, the three methods with the most jagged paths are also the three weakest planners (Tables 7 and 8). The rank correlation does not track planning. SG + SVReg plans hard U-maze goals well although its latent

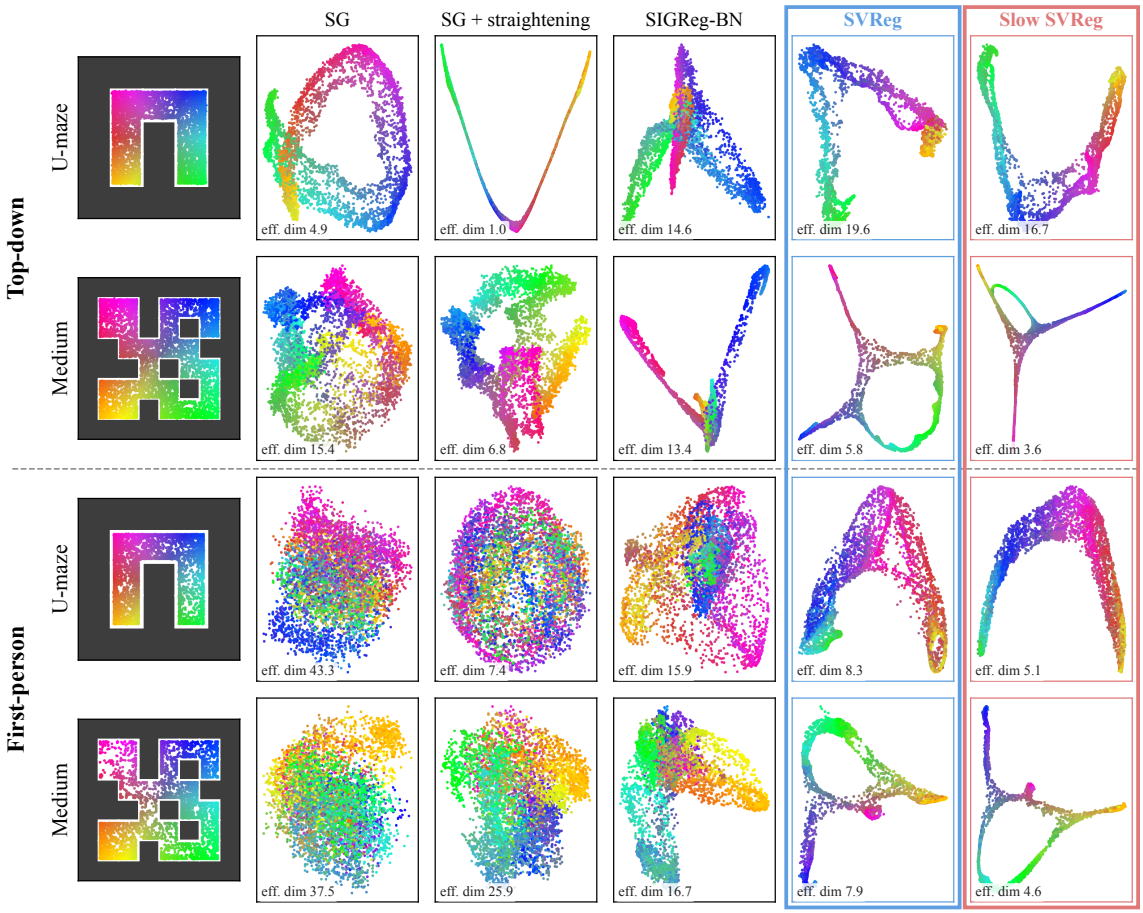


Figure 4: ***Without stop-gradient, the latent traces the maze from top-down and first-person views.*** *Held-out PCA latents colored by position, with SVReg boxed in blue and Slow SVReg in red. First-person panels are shown for the methods in Table 2. Setups in Section C.*

distance barely follows the maze. Planning success follows path smoothness more closely than global distance ranking, which fits a planner that compares latent distances over a few steps.

5.4 Content

Geometry alone does not explain the object tasks. On Cube, SIGReg-BN and SG + straightening plan second and third in Table 1 although their latent distances follow goal distance least well, while SVReg ranks goal distance well and still plans poorly. Here the planners differ in content, and in particular in the cube rather than the arm (Table 9). VICReg, SVReg and SG + SVReg, the weakest Cube planners, detect gripper–cube contact least reliably, and SVReg can lose the cube's height (Figure 5b).

PushT shows the same effect more sharply. A resting block changes only when the agent pushes it, so it is hard to predict from the agent's motion. VICReg, SIGReg-BN and SVReg all lose it, and their held-out block readouts fall below predicting the mean (Table 10). The two best PushT planners in Table 1, Slow SVReg and SG + SVReg, both keep the block.

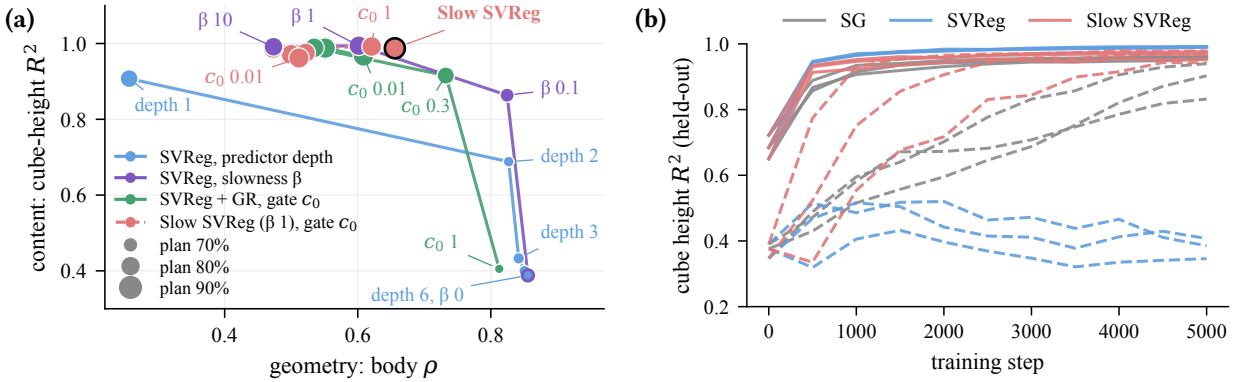


Figure 5: ***Cube content against geometry, and cube-height acquisition.*** *(a) Each curve varies predictor depth, slowness weight β, or the fixed gate c_0 of the gated residual predictor (GR), the last without and with slow channels. Gate curves use runs in which GR alone trains well. ρ: rank correlation between latent and effector distances. Circle area: planning success. Black edge: Slow SVReg. (b) Cube-height readout early in training, dashed for seeds with a weak initial readout (Section C).*

Keeping this content usually costs geometry. Along single knobs on Cube, such as predictor depth or the slowness weight, a higher cube-height readout comes with a lower body ρ (Figure 5a). Slow SVReg reaches the high-content end of this tradeoff and gives up little geometry. It reliably keeps cube height and contact, keeps the PushT block in most runs (Figure 1b), and its cost-to-goal ρ stays close to SVReg's on both tasks. Two-stage, an SVReg head trained on a frozen SG encoder, keeps content but plans below Slow SVReg on all tasks (Section D).

5.5 Content Is Decided Early

The random initial encoder decides which SVReg runs lose cube height. For each training seed, the three objectives start from the same encoder, and SVReg fails exactly from the initial encoders that read the least height. From these starts, SG keeps acquiring height while SVReg does not (Figure 5b). In shorter reruns, Slow SVReg acquires height from all starts, more slowly from the weakest ones (Figure 5b).

Once acquired, height survives the removal of SG. In Section F, switching from SG to SVReg at 20k steps keeps cube height and plans better than SG. SVReg's content failure therefore lies in acquisition.

5.6 Ablation

Table 3 adds the slow channels and the gated residual predictor (GR) to SVReg separately and together, and Table 4 adds GR to the other objectives on Cube. Each part fixes a different failure of SVReg, and together they give the largest gains on Cube and PushT. On Cube, the slow channels alone give most of the gain, while the gated residual alone fails in some runs, even over a larger set of training runs, where Slow SVReg never does. In these runs the latent stays longer in an early phase of fast, jittery change and acquires cube height late. The two failing runs we continued recover with twice the training steps, so their failure is a delay beyond the step budget. On PushT, both parts raise success, but the block becomes readable mainly with the gated residual (Section E).

We find that these gains are tied to the sliced floor trained without SG. Restricting SVReg's directions to the coordinate axes gives the element-wise floor, which loses cube height and plans at about 50% on

Table 3: ***Planning success (%) with each part of Slow SVReg.*** *Hard goals on the mazes. Plain SVReg at its Table 1 setting, the others at Slow SVReg's.*

Task	plain	+ GR	+ Slow	+ Slow + GR
U-maze	96.4±1.3	98.1±0.9	96.6±1.7	**98.7**±0.7
Medium	81.3±2.3	**86.0**±3.1	82.7±2.5	84.7±2.7
Reacher	98.7±0.9	**99.7**±0.3	96.9±1.5	99.6±0.4
Cube	76.9±8.8	78.6±9.7	84.6±3.3	**89.1**±1.6
PushT	75.0±8.6	85.9±3.9	78.9±6.6	**87.3**±3.8

Table 4: ***Planning success (%) on Cube with GR added.*** *Each objective at its Table 1 setting.*

Method	plain	+ GR
SG + SVReg	77.2±1.3	79.9±6.8
SG	80.4±2.1	51.4±7.7
SG + straightening	85.4±1.0	58.6±2.5
SIGReg-BN	87.4±0.8	86.4±0.5
VICReg	69.6±5.8	78.3±7.7

Cube (Section E). Under SG, the predictor's default copy is a trivial solution, and SG with the gated residual collapses to a latent without cube information (Table 4). Without SG, the gated residual also raises VICReg's Cube success.

5.7 Data Scaling

We test whether the gains hold beyond the dataset sizes and training budget adopted in Table 1. In Figure 13, Slow SVReg reaches high success from little data. On hard U-maze goals it nearly saturates with a fifth of the Table 1 data, where SG stays far below even with all of it. On Medium and Reacher, it plans well above SVReg and SG with only 1k to 2k frames, and on Cube it plans better than both from 5k frames up to the Table 1 size (protocol in Section G).

In Figure 11, Slow SVReg on PushT improves with longer training. With a short learning-rate cooldown, its success keeps rising up to 150k steps. SG + straightening, the next-best method, improves at a similar rate, and Slow SVReg keeps its lead.

6 City Walks

Inspired by the results in first-person mazes, we next ask whether the same recipe learns a map from first-person video and actions of real places. Wanderland (Liu et al., 2026) provides 3D Gaussian-splat reconstructions (Kerbl et al., 2023) of real city blocks and buildings, captured on foot. In each scene we render first-person walks with random turns and forward steps along the capture path, and train on the frames and actions only, without poses (Section H). Linear readouts fit on the training walks give the agent's position and heading on held-out walks. A map score measures how well the top two principal components of the latent lay out the walked area, and a held-out disc outside the training walks tests how far the map extends.

In Table 5, linear readouts of the latent recover position for the objectives trained without SG. Slow SVReg reads position most accurately on scene 13Skv and remains the most accurate in the held-out area, where every readout extrapolates. SG also reads position, and it reads heading better than the SVReg variants. SG + straightening reads position in some runs and loses it in others, and among the trained models VICReg has the largest position errors. An untrained encoder of the same architecture reads position barely better than predicting the mean position and keeps no map.

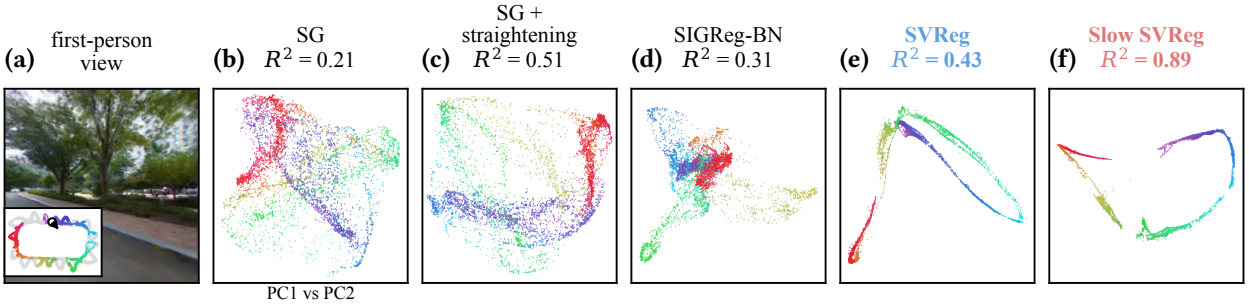


Figure 6: ***Held-out walks on Wanderland scene 13Skv in the top two principal components of each latent.*** *Color: true position. Titles: map score. (a) A rendered first-person frame and the walked area (inset). (b–f) SG, SG + straightening, SIGReg-BN, SVReg and Slow SVReg.*

Table 5: ***Latent maps from city walks.*** *Scene 13Skv and the mean over ten further scenes (Table 12). Seeds and selection in Section H.*

	scene 13Skv				ten further scenes (mean)	
Method	position (m) ↓	held-out area (m) ↓	heading (°) ↓	map R^2 ↑	position (m) ↓	map R^2 ↑
Untrained encoder (control)	25.9	41.2	54.1	0.02	13.7	0.04
SG	3.4	10.0	**3.4**	0.26	5.4	0.10
SG + straightening	9.1	14.5	11.0	0.24	5.2	0.21
VICReg	12.5	21.3	21.1	0.12	5.3	0.13
SIGReg-BN	2.1	11.9	7.1	0.30	**2.1**	0.21
SVReg	5.4	12.2	8.0	0.54	4.1	0.60
Slow SVReg (ours)	**1.6**	**6.3**	6.3	**0.88**	3.1	**0.63**

The latents trained without SG also recognize places across viewing directions. For most validation frames, the nearest training frame in the latent among those facing more than 90° away was taken within 5 m, whereas SG mostly matches views with similar headings (Section H).

What Slow SVReg and SVReg add is geometry. In Figure 6, the walked layout appears in the leading principal components of their latents, with no fitted readout. Slow SVReg forms the clearest map, followed by SVReg, both well above the other single-stage objectives (Table 5).

The map forms with experience. In Figure 14, position error drops by an order of magnitude as the training walks grow from two to one hundred and revisit the same streets. Earlier predictive and slow-feature models found position codes in simulated or small real environments, mostly through trained decoders (Section 7). Here the street layout of a reconstructed city block appears in the plain principal components of an action-conditioned JEPA trained without poses or stop-gradient.

We also train separate models on ten further scenes, four indoor and six outdoor, at the scene-13Skv setting, with two runs per scene. In Table 12, an SVReg variant has the highest map score in all ten scenes, most often Slow SVReg. Using a linear readout, SIGReg-BN localizes best across scenes, but its leading components form no map.

7 Related Work

Self-supervised learning and anti-collapse. To prevent collapse, contrastive methods use negative pairs (Chen et al., 2020; He et al., 2020), and non-contrastive methods with stop-gradient, with or without an exponential moving average (EMA) target (Grill et al., 2020; Chen and He, 2021), or centering (Caron et al., 2021). These objectives are equivalent under limited assumptions (Garrido et al., 2023a). Barlow Twins decorrelates (Zbontar et al., 2021), VICReg floors element-wise standard deviations (Bardes et al., 2022), and SIGReg and VISReg match a Gaussian shape along random slices (Balestriero and LeCun, 2025; Wu et al., 2026). BYOL also trains with weight-standardized group normalization instead of batch normalization (Richemond et al., 2020). Theory shows that the non-contrastive objective collapses without stop-gradient (Tian et al., 2021) and characterizes the features it learns (Tang et al., 2023; Littwin et al., 2024). Our floor is one-sided like VICReg's and sliced like SIGReg's, and we compare a stop-gradient target with a regularized target that receives gradients by what content and geometry each keeps for a planner.

JEPA and world models. JEPA learns through temporal prediction (LeCun, 2022) and scales to images and video (Assran et al., 2023; Bardes et al., 2024). Latent dynamics support planning in World Models, PlaNet, Dreamer, TD-MPC and DINO-WM (Ha and Schmidhuber, 2018; Hafner et al., 2019, 2020, 2021; Hansen et al., 2022, 2024; Zhou et al., 2025). PLDM trains without SG using VICReg-style losses, inverse dynamics and whole-latent slowness (Sobal et al., 2025), and LeWM uses SIGReg from pixels (Maes et al., 2026). Terver et al. (2025) study what drives planning success in JEPA world models, and AdaJEPA adapts a pretrained JEPA world model at test time under SG (Wang et al., 2026a). Content loss also appears in JEPAs that focus on slow features and fail with a fixed distractor (Sobal et al., 2022), and in Pong, where LeWM suppresses the ball that a stop-gradient variant recovers (Karmann et al., 2026). We analyze how the target gradient affects content and noise, and favor slowly changing content without SG.

Temporal representation learning and straightening. Slow Feature Analysis extracts slow features under unit-variance and decorrelation constraints (Wiskott and Sejnowski, 2002), CPC contrasts future observations with unrelated ones (van den Oord et al., 2018), and deep networks have used the temporal coherence of video frames to learn image features (Mobahi et al., 2009; Goroshin et al., 2015; Jayaraman and Grauman, 2016). Koopman methods learn linear latent dynamics, scored by VAMP (Wu and Noé, 2020) or trained with autoencoders (Lusch et al., 2018), and JEPA connects to Koopman invariant subspaces (Ruiz-Morales et al., 2025). Perception straightens natural-video trajectories (Hénaff et al., 2019), and explicit curvature regularization supports latent planning (Wang et al., 2026b). Our no-stop-gradient objective is VAMP-like in the linear setting (Theorem 2.1), but SVReg learns smooth latent paths with nonlinear encoders and predictors, without a curvature penalty.

Spatial representations from prediction. Predictive and slow-feature models encode position, but prior work reads it mostly through trained decoders or single units. Examples are navigation agents with auxiliary tasks (Mirowski et al., 2017, 2018), world models with belief states or next-frame prediction (Gregor et al., 2019; Gornet and Thomson, 2024), and slow feature analysis of camera views in small areas (Franzius et al., 2007; Metka et al., 2018). Without a decoder, coarser structure appears in the top principal components of a network that predicts 20-dimensional sensor readings (Recanatesi et al., 2021) and in spatially contiguous k-means clusters of a regularized world model (Xia et al., 2026). In our latents, the top two principal components lay out first-person mazes and the walked streets of reconstructed city blocks, learned from frames and actions without poses.

8 Conclusion

A latent world model for planning needs to learn *content* that encodes task-relevant states, and *geometry* that smoothly tracks progress toward a goal. SG, whose target is one-sided, learns content but poor geometry, and common regularizers, which constrain the latent feature distribution beyond a variance floor, also struggle with geometry. In contrast, SVReg learns smooth latent maps, but it can lose task-relevant content. To address this tradeoff, we introduce Slow SVReg, which favors slowly changing factors. It keeps this content and improves planning. Slow SVReg plans best in the hard settings of visual control tasks, and moreover, in first-person mazes and city walks, it also learns the map layout in its leading principal components.

Acknowledgments

This research was supported by NSF Award 1922658, 2545541, AFOSR grant FA95502310139, Visko Platform, Pionex, a Google TPU Award, a Google ML and Systems Junior Faculty Award, and the Institute of Information & Communications Technology Planning & Evaluation (IITP) with a grant funded by the Ministry of Science and ICT (MSIT) of the Republic of Korea in connection with the Global AI Frontier Lab International Collaborative Research Agreement (No. RS-2024-00469482 & RS-2024-00509279). The compute is supported by the NYU High Performance Computing resources, services, and staff expertise. We gratefully acknowledge use of the research computing resources of the Empire AI Consortium, Inc, with support from Empire State Development of the State of New York, the Simons Foundation, and the Secunda Family Foundation.

References

Assran, M., Duval, Q., Misra, I., Bojanowski, P., Vincent, P., Rabbat, M., LeCun, Y., and Ballas, N. (2023). Self-supervised learning from images with a joint-embedding predictive architecture. In *CVPR*.

Balestriero, R. and LeCun, Y. (2025). LeJEPA: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*.

Bardes, A., Garrido, Q., Ponce, J., Chen, X., Rabbat, M., LeCun, Y., Assran, M., and Ballas, N. (2024). Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*.

Bardes, A., Ponce, J., and LeCun, Y. (2022). VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *ICLR*.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. (2021). Emerging properties in self-supervised vision transformers. In *ICCV*.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A simple framework for contrastive learning of visual representations. In *ICML*.

Chen, X. and He, K. (2021). Exploring simple siamese representation learning. In *CVPR*.

Chi, C., Xu, Z., Feng, S., Cousineau, E., Du, Y., Burchfiel, B., Tedrake, R., and Song, S. (2025). Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704.

Denton, E. L. and Birodkar, V. (2017). Unsupervised learning of disentangled representations from video. In *Advances in Neural Information Processing Systems (NIPS)*.

Franzius, M., Sprekeler, H., and Wiskott, L. (2007). Slowness and sparseness lead to place, head-direction, and spatial-view cells. *PLoS Computational Biology*, 3(8):e166.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. (2020). D4rl: Datasets for deep data-driven reinforcement learning.

Garrido, Q., Chen, Y., Bardes, A., Najman, L., and LeCun, Y. (2023a). On the duality between contrastive and non-contrastive self-supervised learning. In *ICLR*.

Garrido, Q., Najman, L., and LeCun, Y. (2023b). Self-supervised learning of split invariant equivariant representations. In *International Conference on Machine Learning (ICML)*.

Gornet, J. and Thomson, M. (2024). Automated construction of cognitive maps with visual predictive coding. *Nature Machine Intelligence*, 6(7):820–833.

Goroshin, R., Bruna, J., Tompson, J., Eigen, D., and LeCun, Y. (2015). Unsupervised learning of spatiotemporally coherent metrics. In *ICCV*.

Gregor, K., Jimenez Rezende, D., Besse, F., Wu, Y., Merzic, H., and van den Oord, A. (2019). Shaping belief states with generative environment models for RL. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Avila Pires, B., Guo, Z. D., Gheshlaghi Azar, M., Piot, B., Kavukcuoglu, K., Munos, R., and Valko, M. (2020). Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*.

Ha, D. and Schmidhuber, J. (2018). World models. *arXiv preprint arXiv:1803.10122*.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. (2020). Dream to control: Learning behaviors by latent imagination. In *ICLR*.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. (2019). Learning latent dynamics for planning from pixels. In *ICML*.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. (2021). Mastering atari with discrete world models. In *ICLR*.

Hansen, N., Su, H., and Wang, X. (2024). TD-MPC2: Scalable, robust world models for continuous control. In *ICLR*.

Hansen, N., Wang, X., and Su, H. (2022). Temporal difference learning for model predictive control. In *ICML*.

Hauri, M., Buttaroni, P., Mikulasch, F. A., and Zenke, F. (2026). Learning commute-time-preserving world models for planning. *arXiv preprint arXiv:2610.01373*.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. In *CVPR*.

Hénaff, O. J., Goris, R. L. T., and Simoncelli, E. P. (2019). Perceptual straightening of natural videos. *Nature Neuroscience*, 22:984–991.

Jayaraman, D. and Grauman, K. (2016). Slow and steady feature analysis: Higher order temporal coherence in video. In *CVPR*.

Karmann, M., Li, S., Internò, C., Andreis, B., Klindt, D., Balestriero, R., Gu, J., Torr, P., Zhang, Q., Jiang, P.-T., Zhang, H., Li, B., and Urfalioglu, O. (2026). MotionJEPA: Preventing temporal feature collapse by capturing visual changes in latent space. *arXiv preprint arXiv:2609.23881*.

Kerbl, B., Kopanas, G., Leimkühler, T., and Drettakis, G. (2023). 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42(4).

Kuang, Y., Dagade, Y., Le Lidec, Q., Maes, L., Balestriero, R., and LeCun, Y. (2026). LpWM: A case for sparse representations in world models. *arXiv preprint arXiv:2608.22764*.

LeCun, Y. (2022). A path towards autonomous machine intelligence. OpenReview preprint, https://openreview.net/forum?id=BZ5a1r-kVsf.

Littwin, E., Saremi, O., Advani, M., Thilak, V., Nakkiran, P., Huang, C., and Susskind, J. M. (2024). How JEPA avoids noisy features: The implicit bias of deep linear self distillation networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37.

Liu, X., Li, J., Deng, Y., Chen, R., Zhang, Y., Ma, Y., Guo, L., Li, Y., Zhang, J., and Feng, C. (2026). Wanderland: Geometrically grounded simulation for open-world embodied AI. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1041–1052.

Lusch, B., Kutz, J. N., and Brunton, S. L. (2018). Deep learning for universal linear embeddings of nonlinear dynamics. *Nature Communications*, 9:4950.

Maes, L., Le Lidec, Q., Scieur, D., LeCun, Y., and Balestriero, R. (2026). LeWorldModel: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*.

Mardt, A., Pasquali, L., Wu, H., and Noé, F. (2018). VAMPnets for deep learning of molecular kinetics. *Nature Communications*, 9:5. arXiv:1710.06012.

Metka, B., Franzius, M., and Bauer-Wersing, U. (2018). Bio-inspired visual self-localization in real world scenarios using slow feature analysis. *PLOS ONE*, 13(9):e0203994.

Mirowski, P., Grimes, M., Malinowski, M., Hermann, K. M., Anderson, K., Teplyashin, D., Simonyan, K., Kavukcuoglu, K., Zisserman, A., and Hadsell, R. (2018). Learning to navigate in cities without a map. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31.

Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A. J., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., Kumaran, D., and Hadsell, R. (2017). Learning to navigate in complex environments. In *ICLR*.

Mobahi, H., Collobert, R., and Weston, J. (2009). Deep learning from temporal coherence in video. In *ICML*.

Ni, T., Eysenbach, B., Seyedsalehi, E., Ma, M., Gehring, C., Mahajan, A., and Bacon, P.-L. (2024). Bridging state and history representations: Understanding self-predictive RL. In *International Conference on Learning Representations (ICLR)*.

Park, S., Frans, K., Eysenbach, B., and Levine, S. (2025). OGBench: Benchmarking offline goal-conditioned RL. In *International Conference on Learning Representations (ICLR)*.

Ponce, J., Terver, B., Hebert, M., and Arbel, M. (2026). Dual perspectives on non-contrastive self-supervised learning. In *International Conference on Learning Representations (ICLR)*.

Recanatesi, S., Farrell, M., Lajoie, G., Deneve, S., Rigotti, M., and Shea-Brown, E. (2021). Predictive learning as a network mechanism for extracting low-dimensional latent space representations. *Nature Communications*, 12(1):1417.

Richemond, P. H., Grill, J.-B., Altché, F., Tallec, C., Strub, F., Brock, A., Smith, S., De, S., Pascanu, R., Piot, B., and Valko, M. (2020). BYOL works even without batch statistics. *arXiv preprint arXiv:2010.10241*.

Ruiz-Morales, P., Vanoost, D., Pissoort, D., and Verbeke, M. (2025). Koopman invariants as drivers of emergent time-series clustering in joint-embedding predictive architectures. *arXiv preprint arXiv:2511.09783*.

Sobal, V., S V, J., Jalagam, S., Carion, N., Cho, K., and LeCun, Y. (2022). Joint embedding predictive architectures focus on slow features. *arXiv preprint arXiv:2211.10831*. NeurIPS 2022 Workshop on Self-Supervised Learning: Theory and Practice.

Sobal, V., Zhang, W., Cho, K., Balestriero, R., Rudner, T. G. J., and LeCun, Y. (2025). Learning from reward-free offline data: A case for planning with latent dynamics models. *arXiv preprint arXiv:2502.14819*.

Tang, Y., Guo, Z. D., Richemond, P. H., Pires, B. Á., Chandak, Y., Munos, R., Rowland, M., Azar, M. G., Lan, C. L., Lyle, C., György, A., Thakoor, S., Dabney, W., Piot, B., Calandriello, D., and Valko, M. (2023). Understanding self-predictive learning for reinforcement learning. In *International Conference on Machine Learning (ICML)*, volume 202 of *PMLR*, pages 33632–33656.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., de Las Casas, D., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., Lillicrap, T., and Riedmiller, M. (2018). DeepMind Control Suite. *arXiv e-prints*, page arXiv:1801.00690.

Terver, B., Yang, T.-Y., Ponce, J., Bardes, A., and LeCun, Y. (2025). What drives success in physical planning with joint-embedding predictive world models? *arXiv preprint arXiv:2512.24497*.

Tian, Y., Chen, X., and Ganguli, S. (2021). Understanding self-supervised learning dynamics without contrastive pairs. In *ICML*.

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., and Jégou, H. (2021). Going deeper with image transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 32–42.

van den Oord, A., Li, Y., and Vinyals, O. (2018). Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Wang, Y., Bounou, O., LeCun, Y., and Ren, M. (2026a). Adajepa: An adaptive latent world model. *arXiv preprint arXiv:2606.32026*.

Wang, Y., Bounou, O., Zhou, G., Balestriero, R., Rudner, T. G. J., LeCun, Y., and Ren, M. (2026b). Temporal straightening for latent planning. *arXiv preprint arXiv:2603.12231*.

Wiskott, L. and Sejnowski, T. J. (2002). Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770.

Wu, H., Balestriero, R., and Levine, M. (2026). VISReg: Variance-invariance-sketching regularization for JEPA training. *arXiv preprint arXiv:2606.02572*.

Wu, H. and Noé, F. (2020). Variational approach for learning Markov processes from time series data. *Journal of Nonlinear Science*, 30:23–66. arXiv:1707.04659.

Xia, Z., Lu, Y., Li, X., Xu, Y., and Chen, Y. (2026). Cloning deterministic worlds: The critical role of latent geometry in long-horizon world models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 32602–32612.

Zbontar, J., Jing, L., Misra, I., LeCun, Y., and Deny, S. (2021). Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*.

Zhou, G., Pan, H., LeCun, Y., and Pinto, L. (2025). DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *ICML*.

A Analysis: Proofs and Discussion

This appendix supports the three claims of Section 2. The proof of Theorem 2.1 and its extension to actions (Section A.3) show that without stop-gradient the loss penalizes unpredictable variance (Section 2.1). Section A.4 shows that with finite data a stop-gradient encoder can read noise while its latent statistics look healthy (Section 2.2). Section A.5 shows how a hard-to-predict factor loses a latent coordinate under limited capacity (Section 2.3). Section A.6 relates the analysis to prior work.

A.1 Setting and Notation

For Theorem 2.1 and its action extension (Section A.3), we use a linear encoder $\Phi(o) = Wo$ with $W \in \mathbb{R}^{r\times d}$, a linear predictor $A \in \mathbb{R}^{r\times r}$, mean-centered observations and the sample covariances $\hat{\Sigma}_{00} = \frac{1}{T-1}\sum_t \boldsymbol{o}_t\boldsymbol{o}_t^\top$, $\hat{\Sigma}_{11} = \frac{1}{T-1}\sum_t \boldsymbol{o}_{t+1}\boldsymbol{o}_{t+1}^\top$, $\hat{\Sigma}_{01} = \frac{1}{T-1}\sum_t \boldsymbol{o}_t\boldsymbol{o}_{t+1}^\top$. Index 0 refers to the current frame and index 1 to the next frame. Write $\Sigma_z = W\hat{\Sigma}_{00}W^\top$ and $C_z = W\hat{\Sigma}_{01}W^\top$, and assume Σ_z is invertible, so that no latent direction is constant on the training frames. We treat each block of $T-1$ frames as centered, so these second-moment matrices are covariances.

A.2 Proof of Theorem 2.1

Part 1 (without stop-gradient). The empirical loss expands as

$$\mathcal{L}(W,A) \;=\; \frac{1}{T-1}\sum_t \|AW\boldsymbol{o}_t - W\boldsymbol{o}_{t+1}\|^2 \;=\; \operatorname{tr}(A\Sigma_z A^\top) - 2\operatorname{tr}(AC_z) + \operatorname{tr}\big(W\hat{\Sigma}_{11}W^\top\big),$$

which is quadratic in A. Setting $\nabla_A\mathcal{L} = 2A\Sigma_z - 2C_z^\top = 0$ gives $A^\star = C_z^\top\Sigma_z^{-1}$. Since Σ_z is positive definite, the loss is strictly convex in A and $A^\star$ is its unique minimizer. With $A^{\star\top} = \Sigma_z^{-1}C_z$, the first two terms become $\operatorname{tr}(C_z^\top\Sigma_z^{-1}C_z) - 2\operatorname{tr}(C_z^\top\Sigma_z^{-1}C_z)$, so

$$\mathcal{L}^\star_{\text{no-sg}}(W) \;=\; \operatorname{tr}\big(W\hat{\Sigma}_{11}W^\top\big) \;-\; \operatorname{tr}(C_z^\top\Sigma_z^{-1}C_z).$$

This is the trace of the residual covariance of $W\boldsymbol{o}_{t+1}$ after linear regression on $\boldsymbol{z}_t$, the unexplained target variance.

Part 2 (with stop-gradient, frozen W_{tgt}). The target is $z'_{t+1} = W_{\text{tgt}}\,\boldsymbol{o}_{t+1}$, treated as constant under ∇_W. Let $\hat{\Sigma}_{zz'} = W\hat{\Sigma}_{01}W_{\text{tgt}}^\top$ and $\Sigma_{z'z'} = W_{\text{tgt}}\hat{\Sigma}_{11}W_{\text{tgt}}^\top$. The loss is

$$\mathcal{L}(W,A) \;=\; \operatorname{tr}(A\Sigma_z A^\top) \;-\; 2\operatorname{tr}(A\,\hat{\Sigma}_{zz'}) \;+\; \operatorname{tr}(\Sigma_{z'z'}).$$

Setting $\nabla_A\mathcal{L} = 0$ gives $A^\star = \hat{\Sigma}_{zz'}^\top\Sigma_z^{-1}$. Substituting as in Part 1 gives

$$\mathcal{L}^\star_{\text{sg}}(W;\,W_{\text{tgt}}) \;=\; \operatorname{tr}(\Sigma_{z'z'}) \;-\; \operatorname{tr}\big(\hat{\Sigma}_{zz'}^\top\Sigma_z^{-1}\hat{\Sigma}_{zz'}\big).$$

The first term does not depend on W, so minimizing the loss over W maximizes the second term, the variance of the frozen target predicted from $\boldsymbol{z}_t$. In both parts $\nabla_A\mathcal{L} = 0$ at $A^\star$, so by the envelope theorem the gradient of $\mathcal{L}^\star$ in W equals the gradient of $\mathcal{L}$ at $A^\star$, the encoder update for a predictor kept at its least-squares fit. ∎

A.3 Action-Conditioned Predictor

With the predictor $\hat{\boldsymbol{z}}_{t+1} = A\boldsymbol{z}_t + B\boldsymbol{a}_t$, both parts of Theorem 2.1 keep their form once the action is linearly regressed out of the observations. The reason is that the optimal B predicts the part of the next latent that

the action explains linearly. Without stop-gradient, the loss then penalizes only the next-step variance left unexplained by the current latent and action together. The action predicts most of the agent's own motion, so this motion adds little to the penalty, while contact-driven object motion adds to it. This is why Section 2.3 defines the residual δ given both the state and the action.

Residual statistics. Let the actions $\boldsymbol{a}_t \in \mathbb{R}^m$ be mean-centered together with the observations, with invertible sample covariance $\hat{\Sigma}_{aa} = \frac{1}{T-1}\sum_t \boldsymbol{a}_t\boldsymbol{a}_t^\top$ and cross-covariances $\hat{\Sigma}_{0a} = \frac{1}{T-1}\sum_t \boldsymbol{o}_t\boldsymbol{a}_t^\top$ and $\hat{\Sigma}_{1a} = \frac{1}{T-1}\sum_t \boldsymbol{o}_{t+1}\boldsymbol{a}_t^\top$ over the same $T-1$ pairs. For each pair t, the least-squares residuals of the current and next frame after regressing out the action $\boldsymbol{a}_t$ are

$$\tilde{\boldsymbol{o}}_{0,t} = \boldsymbol{o}_t - \hat{\Sigma}_{0a}\hat{\Sigma}_{aa}^{-1}\boldsymbol{a}_t, \qquad \tilde{\boldsymbol{o}}_{1,t} = \boldsymbol{o}_{t+1} - \hat{\Sigma}_{1a}\hat{\Sigma}_{aa}^{-1}\boldsymbol{a}_t,$$

and their sample covariances are, for $i, j \in \{0, 1\}$,

$$\hat{\Sigma}_{ij|a} = \hat{\Sigma}_{ij} - \hat{\Sigma}_{ia}\hat{\Sigma}_{aa}^{-1}\hat{\Sigma}_{aj},$$

with $\hat{\Sigma}_{ai} = \hat{\Sigma}_{ia}^\top$ and $\hat{\Sigma}_{10} = \hat{\Sigma}_{01}^\top$.

Without stop-gradient. For fixed (W, A), the loss $\mathcal{L}(W, A, B) = \frac{1}{T-1}\sum_t \|AW\boldsymbol{o}_t + B\boldsymbol{a}_t - W\boldsymbol{o}_{t+1}\|^2$ is a convex quadratic in B, minimized by $B^\star(A) = (W\hat{\Sigma}_{1a} - AW\hat{\Sigma}_{0a})\hat{\Sigma}_{aa}^{-1}$. With this choice the prediction error is

$$AW\boldsymbol{o}_t + B^\star(A)\,\boldsymbol{a}_t - W\boldsymbol{o}_{t+1} = AW\tilde{\boldsymbol{o}}_{0,t} - W\tilde{\boldsymbol{o}}_{1,t},$$

so $\min_B \mathcal{L}(W, A, B)$ is the action-free loss of Part 1 evaluated on the residuals. Minimizing it over A as in Part 1, with $\tilde{\Sigma}_z = W\hat{\Sigma}_{00|a}W^\top$ and $\tilde{C}_z = W\hat{\Sigma}_{01|a}W^\top$, gives $A^\star = \tilde{C}_z^\top\tilde{\Sigma}_z^{-1}$, $B^\star = B^\star(A^\star)$ and

$$\mathcal{L}^\star_{\text{no-sg}}(W) = \operatorname{tr}\big(W\hat{\Sigma}_{11|a}W^\top\big) - \operatorname{tr}\big(\tilde{C}_z^\top\tilde{\Sigma}_z^{-1}\tilde{C}_z\big).$$

Minimizing over B and then over A gives the joint minimum over (A, B). We assume $\tilde{\Sigma}_z$ is invertible, which holds exactly when the predictor input $(\boldsymbol{z}_t, \boldsymbol{a}_t)$ has an invertible sample covariance. This excludes latent directions that are exact linear functions of the action on the sample.

With stop-gradient. With the frozen target $W_{\text{tgt}}\boldsymbol{o}_{t+1}$, this elimination gives

$$B^\star(A) = (W_{\text{tgt}}\hat{\Sigma}_{1a} - AW\hat{\Sigma}_{0a})\hat{\Sigma}_{aa}^{-1}$$

and the error $AW\tilde{\boldsymbol{o}}_{0,t} - W_{\text{tgt}}\tilde{\boldsymbol{o}}_{1,t}$. Part 2 therefore holds with Σ_z replaced by $\tilde{\Sigma}_z$, $\hat{\Sigma}_{zz'} = W\hat{\Sigma}_{01|a}W_{\text{tgt}}^\top$ and the constant first term $\operatorname{tr}(W_{\text{tgt}}\hat{\Sigma}_{11|a}W_{\text{tgt}}^\top)$.

A.4 Signal Collapse (Proposition A.1)

Observation model. Observations combine a zero-mean k-dimensional signal with independent noise, $\boldsymbol{o}_t = U\boldsymbol{s}_t + \sigma\boldsymbol{n}_t$, where $U \in \mathbb{R}^{d\times k}$ has orthonormal columns, $\sigma > 0$, and $\boldsymbol{n}_t \sim \mathcal{N}(0, I_d)$ is independent across time and of the signal (Section B).

Spurious sample structure in finite data. For T samples of i.i.d. noise $n_t \in \mathbb{R}^d$, random-matrix theory gives the empirical lag-1 cross-covariance $\frac{1}{T-1}\sum_t n_t n_{t+1}^\top$ a nonzero spectral norm despite its zero population value. Its scale is of order $\sqrt{d/T}$ for $d \ll T$ and grows further for $d \gtrsim T$, contributing spurious structure of order $\sigma^2\sqrt{d/T}$ to the observation lag-covariance.

The fixed-target objective in Part (2) can select this spurious structure when it exceeds the signal's lag structure. An encoder can keep healthy latent variance while reading noise instead of state. We call this *signal collapse*, and latent statistics can hide it.

Proposition A.1 (Latent covariance does not identify signal recovery). *Let the signal have zero mean and unit covariance, and the noise be isotropic with positive variance. Let the faithful encoder $\Phi^\star$ project onto the signal directions U and the collapsed encoder $\tilde{\Phi}$ onto k directions orthogonal to U, both followed by the same output map with orthonormal columns. Then the two population latent covariances differ only by a global scale, which a variance regularizer can hide. A linear probe of the true state separates them, with population R^2 equal to $1/(1+\sigma^2)$ at $\Phi^\star$ and 0 at $\tilde{\Phi}$.*

Proof. Let $\Phi^\star(o) = M_s U^\top o$ with $M_s \in \mathbb{R}^{r\times k}$ satisfying $M_s^\top M_s = I_k$, and $\tilde{\Phi}(o) = M_s V^\top o$ with $V \in \mathbb{R}^{d\times k}$ orthonormal and $V^\top U = 0$ (assuming $d \geq 2k$).

Part 1: Covariance shape. With $\Sigma_o = U\,\mathrm{Cov}(s)\,U^\top + \sigma^2 I_d$ and $\mathrm{Cov}(s) = I_k$:

$$\mathrm{Cov}(\Phi^\star(o)) = M_s U^\top \Sigma_o U M_s^\top = (1+\sigma^2)\, M_s M_s^\top,$$
$$\mathrm{Cov}(\tilde{\Phi}(o)) = M_s V^\top \Sigma_o V M_s^\top = \sigma^2\, M_s M_s^\top.$$

Both are positive multiples of the same matrix, so the two encoders share their eigenvalue profile up to a global scale, their scale-invariant variance ratios, and their Pearson correlations. Rescaling the collapsed encoder by $\sqrt{1+\sigma^2}/\sigma$ makes its population covariance identical to the faithful encoder's, so covariance alone cannot identify signal recovery when the encoder gain is unrestricted.

Part 2: Ground-truth probe. Define the population coefficient of determination $R^2(z,s) = 1 - \mathbb{E}\|s - \hat{s}(z)\|^2/\mathbb{E}\|s\|^2$, where $\hat{s}(z)$ is the optimal linear predictor of s from z. For $\tilde{\Phi}$, $z = \sigma M_s V^\top n$ contains no s component, so $R^2(\tilde{\Phi}(o), s) = 0$. For $\Phi^\star$, $z = M_s s + \sigma M_s U^\top n$ and, since $M_s^\top M_s = I_k$, applying $M_s^\top$ gives the equivalent observation model $M_s^\top z = s + \sigma U^\top n$, each signal component observed with signal variance 1 and noise variance σ^2, so $R^2(\Phi^\star(o), s) = 1/(1+\sigma^2)$ per component. The gap $1/(1+\sigma^2)$ is positive at any finite noise level. On the training trajectory, finite-sample R^2 can exceed this ceiling when $d \geq T$ through memorization, which is why the population gap is measured on held-out trajectories. ∎

A.5 Content Competition (Proposition A.2)

Proposition A.2 explains why training without stop-gradient can drop a hard-to-predict factor, such as the cube's height in Section 2.3. It uses the scalar mode of Section 2.3, $s_{t+1} = \varphi s_t + \kappa a_t + \epsilon_t$ with $|\varphi| < 1$, innovation (process noise) variance $q = \mathrm{Var}(\epsilon_t)$, $\kappa^2 + q > 0$, and stationary variance $v = \mathrm{Var}(s_t)$. Actions and innovations have zero mean, so the stationary mode has zero mean. Its *one-step residual* $\delta = q/v$ is the share of the mode's variance that is not predicted by the latent and the action. For a unit-variance action, the mode's variance splits into shares φ^2 from persistence, $\alpha = \kappa^2/v$ from the action and δ from the innovation, which sum to one, so $1-\delta$ is the R^2 of the best linear one-step prediction.

Proposition A.2 (Competition for one latent coordinate). *Let two independent stationary modes i and j, each driven by its own action, have persistence φ_i, φ_j, stationary variances v_i, v_j and one-step residuals δ_i, δ_j. They share a one-dimensional latent, trained by gradient flow with encoder learning rate η, a predictor held at its least-squares fit, and SVReg. In one dimension, SVReg is the floor $\lambda(1-c)_+^2$ with $\lambda > 0$, where c is the standard deviation of the latent. Start from the endpoint where the latent holds only mode i at its equilibrium scale, $c = \lambda/(\lambda+\delta_i)$ without stop-gradient and any $c \geq 1$ with it. To first order, a small encoder component along mode j grows at rate γ, with*

$$\gamma_{\text{no-sg}} = 2\eta v_j\big[\delta_i - \delta_j - (\varphi_i - \varphi_j)^2\big], \qquad \gamma_{\text{sg}} = 2\eta v_j\,\varphi_i(\varphi_j - \varphi_i).$$

Comparing the rates. Substituting $\delta = 1 - \varphi^2 - \alpha$ gives

$$\gamma_{\text{no-sg}} = 2\gamma_{\text{sg}} + 2\eta v_j(\alpha_j - \alpha_i).$$

Both rates contain the persistence term $\varphi_i(\varphi_j - \varphi_i)$, which for $\varphi_i > 0$ favors a more persistent newcomer. Only training without stop-gradient adds the action term, so only there can a less persistent, action-driven mode displace a passive, persistent one. Without stop-gradient, the brackets for j entering i and for i entering j sum to $-2(\varphi_i - \varphi_j)^2$, so both endpoints can be stable and the first mode to fill the latent can keep it. With stop-gradient and distinct positive persistences, only the more persistent endpoint is stable.

Proof. Actions (unit variance) and innovations have zero mean, are fresh at each step, and are independent of each other and of the state, so $v = (\kappa^2 + q)/(1 - \varphi^2)$. For one mode with encoder $z_t = ws_t$ ($w \neq 0$), scale $c = |w|\sqrt{v}$ and predictor $\hat{z}_{t+1} = \hat{\varphi} z_t + \hat{\kappa} a_t$, the one-step prediction error e splits into uncorrelated state, action and innovation terms, so

$$\mathbb{E}e^2 = c^2\big[(\hat{\varphi} - \varphi)^2 + \delta\big] + (\hat{\kappa} - w\kappa)^2.$$

Write $h(c) = \lambda(1/c - 1)_+$ for the floor's push, so that $\partial_w \lambda(1 - c)_+^2 = -2h(c)\, wv$.

Endpoint. Without stop-gradient, the loss at the fitted predictor ($\hat{\varphi} = \varphi$, $\hat{\kappa} = w\kappa$) is $\delta c^2 + \lambda(1 - c)_+^2$, where δc^2 is $\mathcal{L}^\star_{\text{no-sg}}$ for this mode (Section A.3). The predictor's gradient vanishes at its fit, so by the envelope theorem the encoder gradient is the derivative of this reduced loss. For $\delta > 0$ it is strictly convex with its minimum where $h(c) = \delta$, at $c = \lambda/(\lambda + \delta)$. So the latent settles slightly below the floor, and the floor keeps pushing for more variance with strength δ. With stop-gradient, w in the target ws_{t+1} is held fixed, the encoder gradient of $\mathbb{E}e^2$ is $2\hat{\varphi}(\hat{\varphi} - \varphi)\, wv$, which vanishes at the fitted predictor, and any $c \geq 1$, where $h = 0$, is a fixed point. So the latent can sit at or above the floor, where the floor stops pushing.

Growth rate. Perturb the encoder to $z = ws_i + \zeta s_j$, with $c_i = |w|\sqrt{v_i}$. For independent modes, the fitted $\hat{\varphi}$ and the latent standard deviation change only at $O(\zeta^2)$, and the loss splits by mode. The mode-j part is $\zeta^2 v_j[(\hat{\varphi} - \varphi_j)^2 + \delta_j]$, and the floor contributes $-2h\zeta v_j$ to the gradient in ζ. With stop-gradient, only the prediction is differentiated, and the prediction-loss gradient is $2\hat{\varphi}\,\mathbb{E}[e\, s_j] = 2\hat{\varphi}(\hat{\varphi} - \varphi_j)\,\zeta v_j$. At $\hat{\varphi} = \varphi_i$,

$$\text{no SG: } \dot{\zeta} = 2\eta v_j\, \zeta\big[h(c_i) - \delta_j - (\varphi_i - \varphi_j)^2\big], \qquad \text{SG: } \dot{\zeta} = 2\eta v_j\, \zeta\big[h(c_i) + \varphi_i(\varphi_j - \varphi_i)\big],$$

up to $O(\zeta^2)$. With $h(c_i) = \delta_i$ without stop-gradient and $h(c_i) = 0$ with it, these are the rates of the proposition. In words, without stop-gradient a new mode enters when the floor's push δ_i outweighs its cost, its own unpredictable variance δ_j plus $(\varphi_i - \varphi_j)^2$ for sharing one predictor coefficient. With stop-gradient, it enters when it is more persistent than the incumbent, for $\varphi_i > 0$. ∎

Shrinkage alone keeps the mode. For any $\lambda > 0$, the endpoint scale without stop-gradient is $\lambda/(\lambda+\delta) \geq \lambda/(\lambda + 1)$, since $\delta \leq 1$, which is 0.5 at $\lambda = 1$ and 0.99 at $\lambda = 100$. The linear readout z/w recovers s at any nonzero scale, so losing a mode needs another mechanism, such as competition for a limited latent.

A.6 Relation to Prior Work

CCA and VAMP. Part (1) of Theorem 2.1 restates known VAMP and Koopman results (Mardt et al., 2018; Wu and Noé, 2020; Ruiz-Morales et al., 2025). With whitened target latents ($W\hat{\Sigma}_{11}W^\top = I_r$), its loss is $r - \sum_i \rho_i^2$, where ρ_i are the canonical correlations between $W\boldsymbol{o}_t$ and $W\boldsymbol{o}_{t+1}$. Minimizing the loss then maximizes the VAMP-2 score (Wu and Noé, 2020), with one W shared across both time steps.

Per-mode analyses. Earlier analyses follow each mode separately. They derive contraction without stop-gradient for static views (Tian et al., 2021) and subspace dynamics under stop-gradient with an optimal predictor (Tang et al., 2023; Ni et al., 2024). Under stop-gradient, joint updates of encoder and predictor generally do not follow the gradient of a scalar objective (Ponce et al., 2026), so Part (2) of Theorem 2.1 describes a single update against a frozen target. Proposition A.2 adds the action-conditioned residual δ and the equilibrium scale $\lambda/(\lambda + \delta)$, where the floor balances the prediction loss.

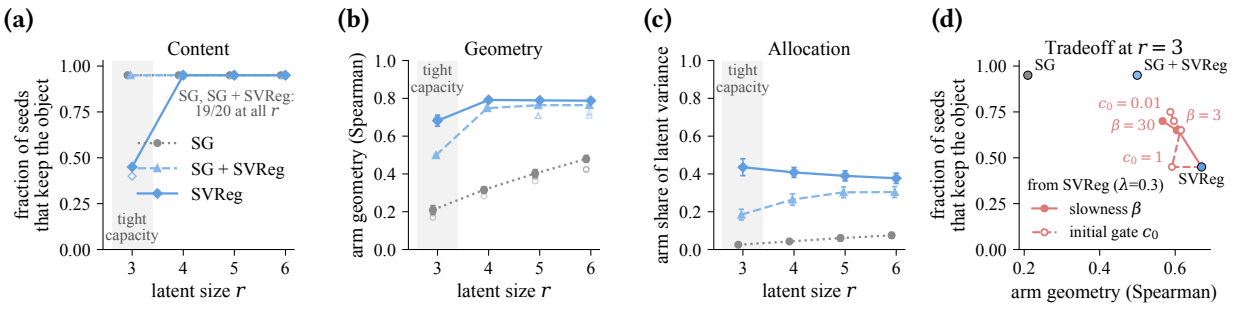


Figure 7: ***Object content and arm geometry under limited latent capacity.*** *(a–c) Objects kept, arm geometry and arm share of latent variance against latent size. (d) Latent size 3 with varied slowness weight and initial gate.*

B Linear Dynamics Experiment Details

High-dimensional linear dynamics. A low-dimensional state $\boldsymbol{s}_t \in \mathbb{R}^k$ follows $\boldsymbol{s}_{t+1} = R\,\boldsymbol{s}_t + K\boldsymbol{a}_t + \boldsymbol{\epsilon}_t$, where $R \in \mathbb{R}^{k\times k}$ is the transition matrix, $K \in \mathbb{R}^{k\times m}$ maps actions to states, and $\boldsymbol{\epsilon}_t$ is a zero-mean innovation. The geometry experiment of Section 4 uses deterministic dynamics without actions. The agent observes the state through a random orthonormal map $U \in \mathbb{R}^{d\times k}$ with isotropic Gaussian noise:

$$\boldsymbol{o}_t = U\boldsymbol{s}_t + \sigma n_t, \qquad n_t \sim \mathcal{N}(0, I_d), \quad t = 0, 1, \ldots, T-1.$$

The noise level σ and sample size T govern finite-sample uncertainty.

The linear world tests the geometry and content predictions of Section 2. Both experiments sample training windows i.i.d. from a fixed trajectory, as in the visual experiments.

Geometry: recovering the dynamics. A state following the model above traces a circle at frequency ω, observed through a random linear map with noise σ=0.3. Recovering this circle on new episodes allows latent distance to follow the state. We sweep ω from slow to fast and the trajectory length T from 250 to 2,000 frames. We compare SG, SVReg and SG + SVReg over eight training runs, with each weight λ selected on the other runs. The held-out probe measures signal recovery on a fresh episode, with an oracle ceiling set by observation noise.

Content: an object under limited latent size. An action-driven arm and a contact-driven object, each with a 2-D position, compete for latent capacity. The object jumps to a random position when the arm enters a contact radius. Observations are a fixed random linear map of the state plus noise, as above, and a linear encoder with an action-conditioned linear predictor learns a latent of size r. At r=3, the latent is smaller than the 4-D state. Over 20 runs, we count how often the object is kept (held-out $R^2 \geq 0.7$) and measure *arm geometry*, the rank correlation of latent and arm distances on held-out frames. Open markers show variants with 3 and 6 distractors.

At r=3, the latent cannot represent both factors fully, so each method trades one against the other. In Figure 7, SVReg keeps the arm and drops the object in about half the runs, while SG and SG + SVReg keep the object at the cost of arm geometry. With more capacity, all three lose the object in at most one run, and SG still spends little latent variance on the arm. Across latent sizes, SVReg has the best arm geometry and SG the worst. A larger slowness weight or a smaller initial gate moves SVReg along this frontier toward keeping the object (Figure 7d). The slow channel here is one of three latent dimensions, a much larger share than in the visual models.

C Experimental Setup Details

C.1 Data and Baselines

Data. PointMaze U-maze (5k frames), PointMaze Medium (20k frames) and Reacher (5k frames) are each trained on one exploration trajectory with uniform-random actions. Cube is trained on 2,000 episodes of the OGBench scripted expert (80k frames). PushT uses the offline dataset of DINO-WM (Zhou et al., 2025). In first-person PointMaze, the agent drives a unicycle through the maze and sees only the corridor in front of it. Its data come from one exploration trajectory of smooth random driving, with 5k frames on the U-maze and 20k on Medium, and the encoder and planner follow the top-down setup.

Baselines. VICReg applies its variance and covariance terms to all frames of the batch. The straightening penalty skips near-stationary steps and averages over the remaining ones. Two-stage selects its SG trunk and head learning rates by the selection rule of the other methods.

C.2 World Model Architecture and Training

Encoders. Table 6 lists the encoder of each task. The methods on a task share it, so within a column of Table 1 they differ only in their objective, and Slow SVReg also in its gated residual predictor. PushT initializes GeM pooling at p=1 (average pooling).

Table 6: ***World models and training per task.*** *One frame is five environment steps.*

	U-maze	Medium	Reacher	Cube	PushT
Data and model					
Data	one uniform-random exploration trajectory			OGBench scripted expert	DINO-WM offline data
Training frames	5k	20k	5k	80k	470k
Image size	224	224	128	224	224
Encoder	ResNet (5 blocks), BatchNorm			pre-activation ResNet (5 blocks), GroupNorm	
Readout	flatten	flatten	global average	GeM	GeM
Latent dim. D	128	128	128	384	128
Predictor	block-causal ViT (depth 6), 3-frame history, 10-d action embedding concatenated to the latent				
Training					
Steps × batch	10k × 32	20k × 32	50k × 32	50k × 64	50k × 32
Optimizer	Adam (encoder), AdamW with lr $5{\times}10^{-4}$ (predictor), no schedule				
Slow SVReg	slow channels: last 1/8 of the latent, weight $1/\lambda$; gated residual: gate initialized at $c_0 = 0.1$				
Slow SVReg (λ, lr)	100, 10^{-4}	10, 10^{-3}	1, 10^{-5}	30, 10^{-4}	100, 10^{-4}
Two-stage	frozen SG encoder, Linear–GELU–Linear head trained with SVReg				
Planning (CEM)					
Samples / elites / iters	200 / 30 / 30	200 / 30 / 30	200 / 30 / 30	300 / 30 / 10	200 / 30 / 30
Horizon (frames)	10 (easy 5)	10 (easy 5)	10	5	5
Replan every / step budget	1 frame / 100	1 frame / 100	1 frame / 100	5 frames / 50	1 frame / 100

Normalization. In pilots, SG with group normalization failed on the mazes, and batch normalization lowered SVReg on PushT. The mazes and Reacher therefore use batch normalization, and Cube and PushT use group normalization.

C.3 Planning

Planner. The planner is CEM model-predictive control. On the mazes and Reacher, its cost is the squared L2 latent distance to the goal summed over all predicted steps, with weight 2^t on step t. PushT uses the final predicted step for the early CEM iterations and this weighted cost afterwards, and Cube uses the final predicted state only. Sampled actions are clamped to $[-1, 1]$ on the mazes and Reacher. On PushT, unbounded CEM drives the agent out of the arena, so actions are clamped to the data's action range, ± 4 in normalized units or about the 0.1–99.9th percentiles. The Cube planner is unbounded, and OGBench clips Cube actions to $[-1, 1]$. Bounding it lowers SG and SVReg by a few points and keeps their order.

Goals. In PointMaze, easy goals are 25 steps from the start with no filter on how far they lie, and hard goals are 50 steps away and at least 3 maze units apart. Cube goals are 25 environment steps ahead with a 50-step budget, as in LeWM (Maes et al., 2026). The Cube *all* column scores all goals, and the $\geq$4 cm column scores only goals whose cube moves more than 4 cm (about 60% of goals). Our start and goal frames come from held-out episodes, whereas LeWM draws them from its training episodes.

Scoring. *Planning success* is the percentage of episodes that reach the goal within the step budget (0 to 100, higher is better). Each trained model is scored by its mean success over three planning seeds, with 50 episodes each or 100 on Reacher. The plan column of the geometry tables in Section D reports success at the Table 1 setting, on hard goals in PointMaze.

C.4 Hyperparameter Selection

Grids. Encoder learning rates range over $\{10^{-6}, 10^{-5}, 10^{-4}, 5{\times}10^{-4}, 10^{-3}\}$. Regularization weights range from 0.3 to 1000 for SVReg and Slow SVReg, 1 to 300 for SG + SVReg, 0.01 to 100 for VICReg and 10^{-4} to 1000 for SG + straightening. SIGReg-BN uses 0.3 to 30 with pooled statistics and 0.05 to 192 with per-frame statistics. For two-stage, the SG trunk is trained with learning rate 10^{-6} or 10^{-5}, and the SVReg head with learning rate 10^{-4}, $5{\times}10^{-4}$ or 10^{-3}. The head's SVReg weight is 1 on Cube and 10 on the other tasks.

Selection rule. Settings are selected on a separate planning seed. Grids are trained on three training seeds, and the leading settings on three more. For each method and task we take the setting with the highest mean success over all six, breaking ties by the lower learning rate and then the smaller weight. Each column is selected on its own goals, so the easy, hard, all and $\geq$4 cm columns can use different settings.

SIGReg-BN. SIGReg-BN screens its grid on one training seed and trains its two best settings per task on the rest. The per-frame grid multiplies the pooled setting's weight by 1/64, 1/16, 1/4, 1 and 4 at three learning rates around the pooled setting's, and extends an axis while the best setting lies on its edge. With pooled statistics, PushT uses encoder learning rate 10^{-5} only. The statistic form omitted from Table 1 plans worse on all tasks.

First-person PointMaze. Each method is selected from its learning-rate grid as in Table 1. SIGReg-BN (both statistic forms), VICReg and SG + SVReg also screen the weight grids of their top-down maze runs and extend their best settings before selection.

C.5 Metrics

Readouts (content). Readout R^2, AUC, distance rank correlation, jaggedness and the participation ratio (eff. dim) are defined in Section 3. Readouts are ridge regressions (α=1) fit on one set of held-out episodes and scored on another, with 18 and 7 episodes on Cube and 15 and 6 on PushT.

- **Cube.** *Contact AUC* detects gripper–cube contact with a linear readout. *Free cube* R^2 reads the cube position on frames without contact and with the gripper at least 5 cm away, and *grasped arm* R^2 reads the gripper position on contact frames. The cube-height R^2 in Figure 5b also comes from a linear readout.
- **PushT.** *Agent* R^2 reads the agent position. *Free block* R^2 reads the block position and angle (x, y, $\cos\theta$, $\sin\theta$) on frames where the block does not move and the agent is at least 60 px away, and *pushed block* R^2 reads them on frames where the block moves.

Distance correlations (geometry). Higher is better for all measures below, and rank correlations range from -1 to 1.

- **Reacher.** *Dist.* ρ is the Spearman correlation of latent and joint-space distance. *10-NN* is the overlap of the 10 nearest neighbors of a frame in latent and joint space (0 to 1).
- **PointMaze.** *Geodesic* ρ is the Spearman correlation between the latent L2 distance of two held-out frames and the shortest-path distance between their agent positions inside the maze. It uses 150 anchors $\times$ 1,500 frames from the second half of the held-out trajectory. *Local* ρ restricts it to pairs at most one maze cell apart.
- **Cube and PushT.** *Cost-to-goal* ρ is the Spearman correlation, over 20,000 random pairs of held-out frames, between the latent L2 distance (the planner's cost) and the true state distance. The true state is the gripper and cube positions in meters on Cube (all 25 held-out episodes). On PushT, it is the agent and block positions in pixels plus the block-angle difference scaled by $20/(\pi/9)$ px/rad, over all 21 held-out episodes. This scaling gives the angle tolerance of the success check, $\pi/9$, the weight of its position tolerance, 20 px.

Jaggedness. Jaggedness uses a 9-step moving average over the full latent. In PointMaze it is the mean over five smooth shortest routes, a fixed route and four routes between random cell pairs at least half as long. The routes are rendered in the simulator at zero velocity, with first-person views facing the direction of travel. The Reacher sweep turns the shoulder through 360° at elbow 60°. Cube has no scripted motion, so its jaggedness is the mean over each held-out episode's frames before the first gripper–cube contact, over 25 segments with at least 20 frames and 2 cm of arm travel each. The true state path on these segments has jaggedness 1.09.

C.6 Figures

Cube content and geometry (Figure 5). Each curve in panel (a) varies one knob, either SVReg predictor depth (1 to 6 blocks), SVReg slowness weight β, or the gate of the gated residual predictor, held fixed at c_0, without and with slow channels ($\beta = 1$, $\lambda = 10$). *Body* ρ is the Spearman correlation between the latent L2 distance and the true effector distance over held-out frame pairs. The black-edged circle marks Slow SVReg at its Table 1 setting, and panel (b) uses the Table 1 settings. The gate curves use runs in which the gated residual alone trains well.

D Latent Geometry and Content

PCA panels show the top two principal components of held-out frames from the Table 1 checkpoints, and the tables use the full latent.

PointMaze. Table 7 gives the path and distance measures of the maze latents in Figure 4 and Figure 8.

Table 7: *PointMaze latent geometry.* *Settings for hard goals.*

	U-maze				Medium			
method	plan (%)	geodesic $\rho \uparrow$	local $\rho \uparrow$	jag. $\downarrow$	plan (%)	geodesic $\rho \uparrow$	local $\rho \uparrow$	jag. $\downarrow$
SG	65.0±18.7	0.68±0.06	0.86±0.03	5.03±0.86	71.6±8.7	0.63±0.08	0.71±0.12	4.48±0.53
SG + straightening	88.0±3.1	**0.95±0.01**	0.67±0.02	**1.02±0.00**	72.8±5.6	0.57±0.08	0.82±0.06	4.38±0.52
SG + SVReg	95.8±1.1	0.15±0.01	0.78±0.04	2.22±0.46	62.7±6.2	0.35±0.09	0.82±0.06	3.96±0.49
Two-stage (SG trunk + SVReg head)	76.0±20.6	**0.95±0.03**	0.85±0.03	4.32±1.08	83.0±2.0	0.94±0.01	0.71±0.04	3.03±0.48
VICReg	94.3±2.3	0.63±0.01	0.78±0.03	1.90±0.64	77.2±2.3	0.92±0.00	0.55±0.03	2.29±0.43
SIGReg-BN	82.2±13.7	0.38±0.07	0.77±0.02	5.14±0.43	78.2±3.8	0.64±0.02	**0.89±0.01**	1.65±0.31
SVReg	96.4±1.3	0.85±0.03	0.86±0.02	1.51±0.23	81.3±2.3	**0.95±0.01**	0.80±0.01	1.07±0.04
Slow SVReg (ours)	98.7±0.7	0.89±0.04	**0.89±0.01**	1.29±0.13	84.7±2.7	0.94±0.00	0.70±0.00	**1.02±0.01**

Reacher. A Reacher frame shows two joint angles. The shoulder turns fully and the elbow covers ±160°, so a faithful latent is a smooth surface with one loop for the shoulder. The elbow may close a second loop, because the two fully folded poses look alike. We pose the arm in the simulator, turn the shoulder through 360° at a fixed elbow and project the frames onto each method's PCA. In Table 8, SVReg and SG + SVReg trace one smooth closed loop, while SG and two-stage trace jagged paths. As in the mazes, local smoothness tracks planning more closely than the global ranking of distances.

Table 8: *Reacher latent geometry.*

method	plan (%)	sweep jag. $\downarrow$	dist. $\rho \uparrow$	10-NN $\uparrow$
SG	86.5±1.6	3.48±0.40	0.29±0.02	0.66±0.01
SG + straightening	86.6±2.9	3.81±0.45	0.18±0.03	0.63±0.02
SG + SVReg	96.3±1.6	**1.06±0.02**	0.46±0.08	0.70±0.01
Two-stage (SG trunk + SVReg head)	92.7±1.1	2.56±0.22	0.47±0.03	0.60±0.03
VICReg	93.1±1.7	1.37±0.07	0.28±0.01	0.68±0.02
SIGReg-BN	95.8±1.3	1.83±0.14	**0.62±0.03**	0.65±0.01
SVReg	98.7±0.9	1.10±0.03	0.45±0.06	**0.72±0.01**
Slow SVReg (ours)	99.6±0.4	1.12±0.03	0.44±0.02	**0.72±0.01**

Cube. The Cube state space has two sheets that meet at grasp and release. Without contact, arm and cube move independently, and the latent must keep the cube's position although nothing moves it. During a grasp, the cube moves with the gripper, so the latent should be a smooth function of the arm. Table 9 gives ridge readouts on held-out episodes for the free cube, contact and geometry, and Figure 9 shows the latent PCA of all methods in Reacher and Cube. SIGReg-BN and VICReg keep cube height only in low-variance

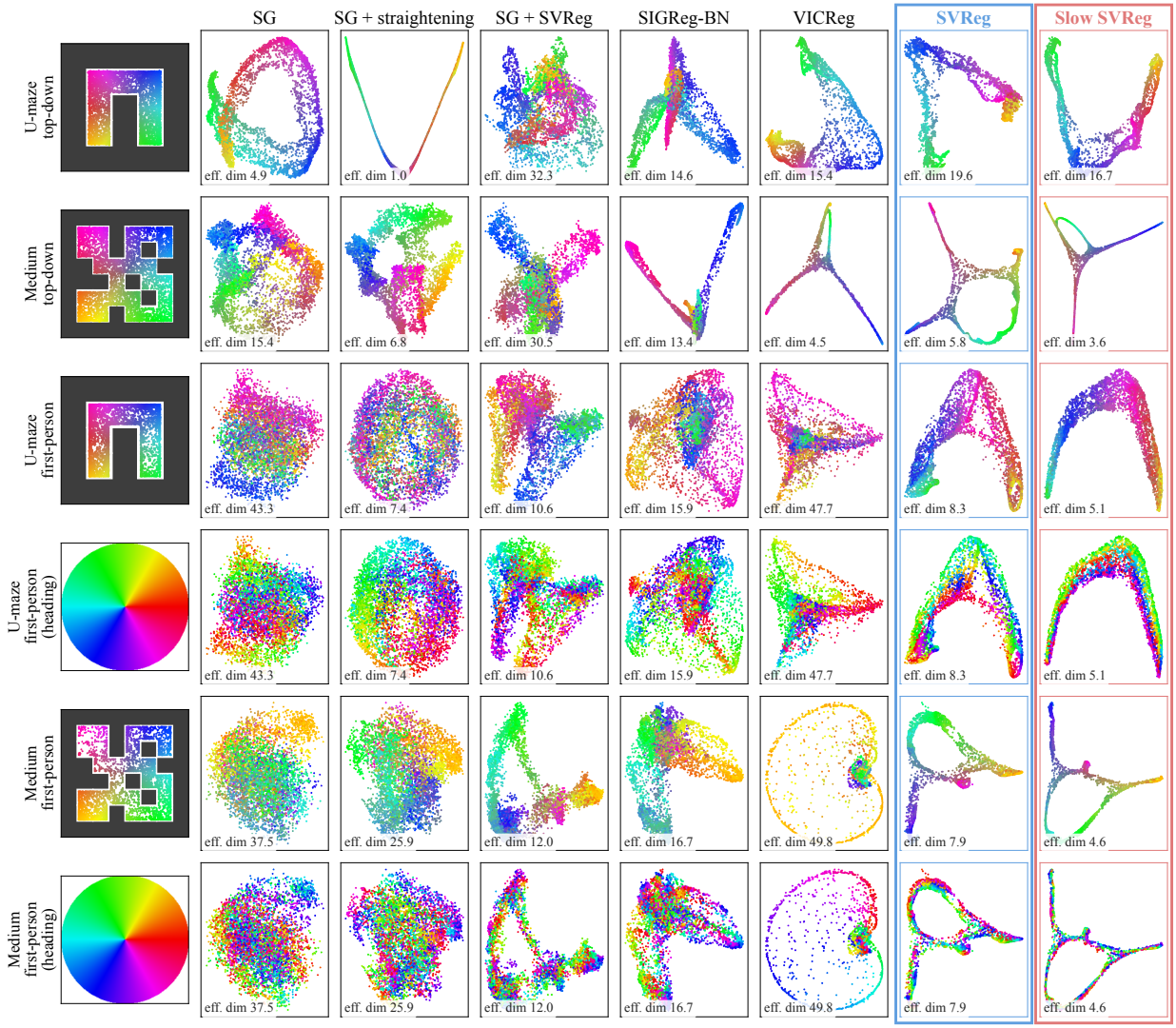


Figure 8: ***Latent PCA for top-down and first-person PointMaze.*** *SVReg and Slow SVReg preserve the map layout, and heading is also visible.*

components. VICReg and SVReg are the only methods that fail to separate contact in some runs. SVReg merges the two sheets in three of six runs, which are also the runs that lose cube height.

Table 9: *Cube latent geometry and content.*

method	plan (%)	eff. dim	contact AUC ↑	free cube R^2 ↑	cost-to-goal ρ ↑	jag. ↓
SG	80.4±2.1	17.7±0.6	0.99±0.00	0.43±0.11	0.38±0.03	1.69±0.03
SG + straightening	85.4±1.0	2.9±1.5	0.99±0.01	0.51±0.08	0.31±0.09	1.19±0.05
SG + SVReg	77.2±1.3	35.4±1.4	0.97±0.02	0.45±0.08	0.35±0.02	2.19±0.21
Two-stage (SG trunk + SVReg head)	86.8±2.4	6.5±0.2	0.99±0.00	**0.61±0.03**	**0.86±0.02**	1.16±0.01
VICReg	69.6±5.8	19.8±0.9	0.86±0.08	0.58±0.06	0.39±0.03	1.63±0.13
SIGReg-BN	87.4±0.8	8.8±0.6	**1.00±0.00**	0.59±0.02	0.29±0.04	**1.11±0.01**
SVReg	76.9±8.8	10.5±1.1	0.80±0.20	0.50±0.05	0.76±0.05	1.19±0.11
Slow SVReg (ours)	89.1±1.6	12.0±0.8	0.99±0.01	0.57±0.09	0.73±0.05	1.12±0.01

PushT. The PushT state is the agent position and the block pose. In Table 10, SVReg's block readout does not transfer to new episodes in five of six runs, whether the block rests or is pushed, and its leading principal components carry the agent instead. Figure 10 traces content and geometry along single knobs on the full PushT dataset.

Table 10: *PushT latent geometry and content.* *SG (two-stage trunk): the frozen SG encoder of two-stage.*

method	plan (%)	eff. dim	agent R^2 ↑	free block R^2 ↑	pushed block R^2 ↑	cost-to-goal ρ ↑
SG	44.0±19.3	13.4±1.1	0.83±0.04	0.20±0.18	0.13±0.20	0.55±0.02
SG + straightening	62.1±7.7	4.8±0.5	0.92±0.01	0.55±0.13	0.61±0.09	0.68±0.02
SG + SVReg	76.9±4.3	14.8±1.2	0.89±0.03	0.72±0.07	0.80±0.07	0.62±0.04
SG (two-stage trunk, lr 1e-5)	49.0±6.1	8.8±0.8	0.91±0.02	0.72±0.05	0.76±0.08	0.80±0.03
Two-stage (SG trunk + SVReg head)	86.7±3.0	7.1±0.3	0.93±0.01	**0.89±0.04**	**0.91±0.03**	**0.84±0.03**
VICReg	55.0±3.6	36.9±4.7	0.90±0.05	−0.57±0.40	−0.28±0.38	−0.01±0.10
SIGReg-BN	60.4±2.6	24.7±0.5	**0.98±0.00**	−0.92±0.23	−0.32±0.22	0.37±0.05
SVReg	75.0±8.6	13.0±0.3	0.97±0.04	−0.27±0.64	0.08±0.41	0.67±0.05
Slow SVReg (ours)	87.3±3.8	10.8±0.5	0.90±0.06	0.79±0.23	0.76±0.26	0.71±0.02

Two-stage. Two-stage freezes an SG encoder and trains a fresh head, predictor and action encoder on its output with SVReg and without SG (Table 6). Table 11 compares it with SG and Slow SVReg on all tasks.

E Content and Ablation Details

Content. At the Table 1 settings, Slow SVReg reads cube height ($R^2 \geq 0.98$) and separates contact in every run, and keeps the PushT block in five of six.

Ablation. The gated residual alone fails, with Cube success below 80% on all goals, in three of the six runs of Table 3 ($\lambda = 30$). Over 16 training seeds at $\lambda = 10$, the gated residual alone fails in four and Slow SVReg in none. On PushT, the resting-block R^2 is 0.12, 0.59 and 0.79 with slow channels, the gated residual and both.

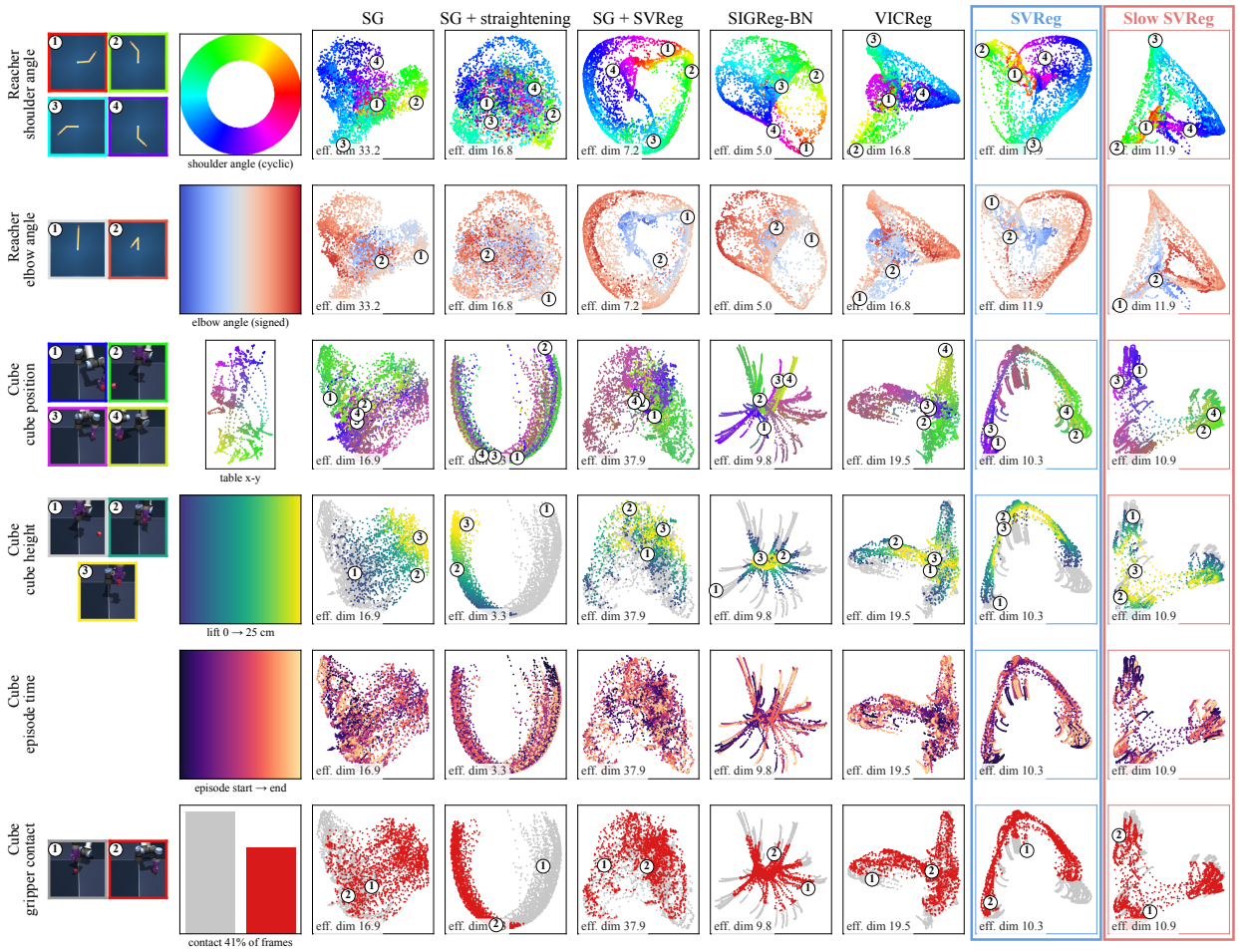


Figure 9: ***Latent PCA in Reacher and Cube.*** *Rows color held-out frames by joint angles (Reacher) and by cube position, lift, episode time and contact (Cube). Numbers mark the reference frames on the left. SVReg spreads the cube's table position over the leading components, and Slow SVReg also separates the lifted cube.*

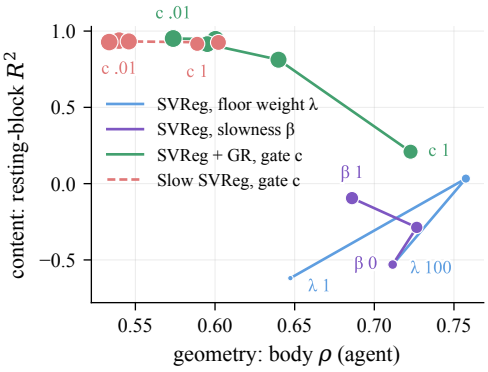


Figure 10: ***PushT content and geometry along single knobs.*** *ρ: rank correlation between latent and agent distances. Circle area: planning success.*

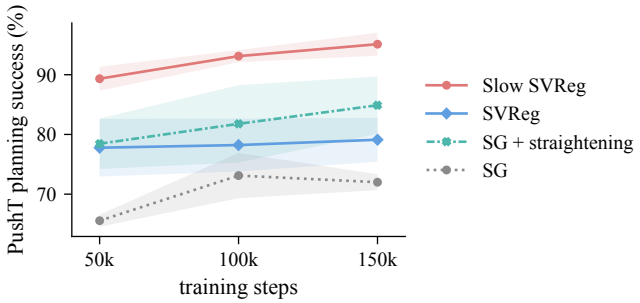


Figure 11: ***Slow SVReg improves with training steps on PushT.*** *Mean with ±1 s.d. shaded. Each method trains on the full data at its Table 1 setting and a constant learning rate for 150k steps. Each point branches off and decays the learning rate linearly to zero over its last 10% of steps.*

Table 11: ***Planning success (%) of two-stage.***

	Agent only: navigation and reaching					Object manipulation		
	U-maze (5k)		Medium (20k)		Reacher (5k)	Cube (80k)		PushT (470k)
Method	easy	hard	easy	hard		all	≥4 cm	
SG	85.1±17.0	65.0±18.7	96.6±3.1	71.6±8.7	86.5±1.6	80.4±2.1	67.2±3.6	44.0±19.3
Two-stage (SG trunk + SVReg head)	80.6±21.7	76.0±20.6	91.9±0.8	83.0±2.0	92.7±1.1	86.8±2.4	78.0±3.6	86.7±3.0
Slow SVReg (ours)	99.7±0.4	98.7±0.7	97.9±0.8	84.7±2.7	99.6±0.4	89.1±1.6	81.9±2.5	87.3±3.8

Early phase. We rerun four seeds of each variant at $\lambda = 10$ with intermediate checkpoints and measure jitter as the variance-weighted share of latent power at periods under five model steps on held-out trajectories. At step 5000, the gated residual alone is jittery in three seeds and reads height less well (R^2 0.54–0.79), while Slow SVReg is already smooth (R^2 0.99). The two failing seeds recover by 100k steps (89% and 87%). The slow channels hold almost none of the height readout's weight, and slowness on the whole latent without a slow block also removes all failures over the 16 seeds.

Controls. The element-wise floor plans at about 50% on Cube and reads cube height poorly ($R^2 < 0.5$) at every $\lambda \in \{1, 10, 100\}$. SG with the gated residual reads no cube height in any run.

F Switching Objectives

Switching from SG to SVReg tests whether acquired content persists. In Figure 12, switching at 20k steps keeps cube height in all runs, while switching at 5k loses it in one.

G Data Scaling

Each scaling run repeats a Table 1 run with less training data and keeps its setting, validation data, number of training steps and planning protocol. Each dataset is a prefix of the original collection, a single explo-

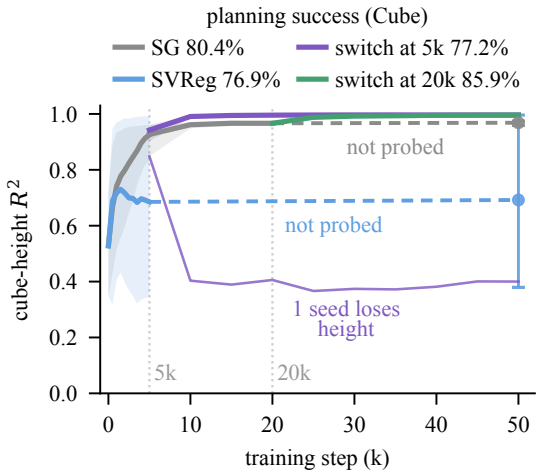


Figure 12: ***Cube-height readout after switching from SG to SVReg.*** *Mean and range over runs. Legend: planning success. One 5k-switch run is drawn apart. Dashed segments were not probed.*

ration trajectory in the mazes and Reacher and a set of expert episodes on Cube. The mazes are scored on hard goals and Cube on goals that move the cube by at least 4 cm, as in Table 1. Each point uses three training runs, and the full-data points use six.



Figure 13: ***Planning success against dataset size.*** *Median over training seeds with the range shaded. Each panel ends at the Table 1 dataset size.*

H City Walks

Data. The random walker stays within 1.2 m of the capture path, where renders stay sharp, and moves into nearby geometry are refused using the rendered depth. The main scene, 13Skv, has 100 training walks of 500 steps, 10 validation walks and a set of held-out walks. A disc of radius one seventh of the walkable extent is held out of all training walks and is scored as a separate set.

Models. The encoder, predictor and objectives follow the maze setup. SIGReg-BN uses per-frame statistics in City Walks, as in LeWM. The untrained control has random weights and BatchNorm statistics recalibrated on the training walks (three initializations). Predicting the mean position gives a 28.0 m error.

Selection. We select each method's learning rate and regularization weight by the lowest validation position error on one training seed. Table 5 averages the selected setting over three further runs, and Figure 6 shows the best of them. The ten further scenes use two runs each.

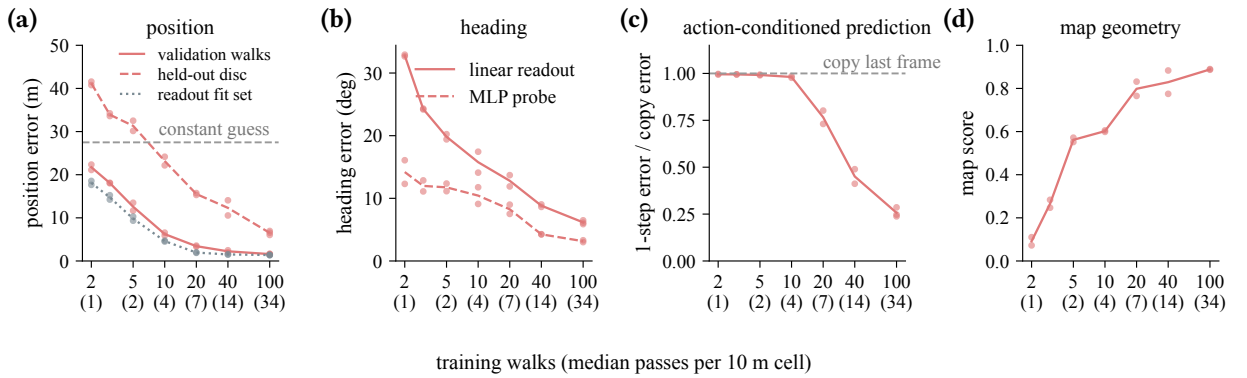


Figure 14: ***City-map measures against the number of training walks.*** *Slow SVReg on scene 13Skv. Dots: runs. (a) Position error. (b) Heading error. (c) One-step prediction error relative to copying the last frame. (d) Map score.*

Table 12: ***Latent maps on ten further Wanderland scenes.*** *Separate models per scene at the scene-13Skv setting, mean of two runs. Bold: best per scene.*

	position (m) ↓						map score ↑					
Scene	SG	SG + str.	VICReg	SIGReg-BN	SVReg	Slow SVReg	SG	SG + str.	VICReg	SIGReg-BN	SVReg	Slow SVReg
Outdoor												
11H3	2.6	5.5	6.3	1.4	2.7	**1.1**	0.14	0.10	0.06	0.10	0.74	**0.84**
1HUMP	5.7	3.1	7.8	**0.7**	2.1	1.0	0.33	0.32	0.09	0.37	0.72	**0.79**
18qtBG3q	3.4	4.5	5.9	**1.0**	3.2	1.3	0.02	0.19	0.11	0.30	0.63	**0.72**
1IjqGTz0	8.1	8.2	8.5	**7.9**	10.0	10.0	0.18	0.41	0.09	0.20	0.56	**0.59**
1L58FmPh	1.9	1.3	3.2	**0.9**	2.0	**0.9**	0.08	0.39	0.15	0.26	**0.58**	0.53
1All8jGb	5.8	9.1	4.5	1.1	3.7	**1.0**	0.01	0.00	0.09	0.23	0.67	**0.72**
Indoor												
1_nV	8.6	5.3	5.7	**2.7**	5.7	3.5	0.03	0.14	0.15	0.05	0.68	**0.77**
12Qy	8.7	6.9	5.6	**3.5**	7.1	6.6	0.04	0.21	0.16	0.20	0.21	**0.32**
1yPum	5.0	4.6	3.8	**1.3**	2.6	3.7	0.05	0.06	0.05	0.04	**0.40**	0.23
1qbLFA84	3.8	3.5	2.1	**1.0**	1.9	2.3	0.15	0.24	0.36	0.39	**0.79**	**0.79**
Mean	5.4	5.2	5.3	**2.1**	4.1	3.1	0.10	0.21	0.13	0.21	0.60	**0.63**
Median	5.4	4.9	5.7	**1.2**	2.9	1.8	0.06	0.20	0.10	0.21	0.65	**0.72**

Other variants. With pooled statistics, SIGReg-BN localizes less accurately and forms no map on any of the eleven scenes. On 13Skv, two-stage (Section D, one setting) maps nearly as clearly as Slow SVReg but reads position less accurately, and SG + straightening reads position in two of three runs.

Metrics. Position and heading come from ridge readouts fit on 6,000 training-walk frames and scored on the validation walks and the held-out disc. The map score is the Procrustes R^2 between the top two principal components of the validation latents and the true positions: 1 for a rotated and scaled copy of the walked area, 0 for no layout. For cross-heading recall (Section 6), chance is 4% and the untrained encoder reaches 5%. On scene 13Skv, Slow SVReg, SIGReg-BN and SVReg reach 75–89%, VICReg and two-stage 26–36%, and SG + straightening and SG about 10%. The ten further scenes give a similar ranking.