LeWorldModel Embeddings are Misspecified for Control

William Peng, Oscar Rodriguez
Stanford University
wpeng@stanford.edu, oscarrm@stanford.edu

Abstract

Joint-embedding predictive architectures (JEPAs) promise to learn a world model once from pixels and then solve tasks by planning, searching for actions whose predicted terminal embedding matches a goal image in L2, with no per-task reward. We ask whether the representations learned by LeWorldModel, trained with next-embedding prediction and the isotropic-Gaussian SIGReg penalty, support such planning beyond the short horizons usually reported. Across two OGBench environments they do not: planning degrades monotonically with the start–goal offset, collapsing to near-failure at long horizons. A mechanistic analysis traces the cause to the representation rather than the dynamics or planning budget, revealing two failure modes. In PointMaze, position is linearly decodable from the embedding ($R^2 = 0.95$) yet lies in a low-variance subspace the raw L2 cost ignores; swapping that cost for a linear position readout restores planning at every horizon. In Scene, the high-DOF state is absent from the planner's embedding yet recoverable from upstream patch tokens, an informational collapse in which a Gaussian-by-construction embedding discards the factors control depends on. We conclude that the task-agnostic predictive objective is fundamentally misspecified for control, and point toward variance- and task-aware capacity allocation as a remedy.

1. Introduction

A central goal for embodied intelligence is to accurately model the world and its dynamics. As such, a great amount of recent effort has been spent in developing world models that learn high-quality representations directly from pixels. In particular, the joint-embedding predictive architecture (JEPA) line of work forgoes pixel-level reconstruction and instead predicts the latent consequences of actions: an encoder maps each observation to an embedding, and a predictor is trained to forecast the embedding of the next observation [13]. By placing the loss in representation space, JEPAs avoid spending capacity on perceptually salient but irrelevant detail, and are intended to distill compact, abstract features of the world. Recent work has shown that these architectures can be trained stably end-to-end from raw pixels by using a next-embedding prediction term and a Gaussianity regularizer that prevents trivial collapse [4]. A key use of these world models is to plan by simple optimization in latent space: given a goal image, one searches for an action sequence whose predicted terminal embedding most closely matches the goal embedding.

This planning-as-inference view makes latent world models attractive for control tasks. The standard instantiation is model predictive control (MPC) via the Cross Entropy Method (CEM) [20]. At each step, CEM samples candidate action sequences, rolls each forward in latent space, and refits its sampling distribution to the lowest-cost samples, iterating until the distribution concentrates on a single action sequence. The initial states and goal states are specified by images and scored by L_2 distance in embedding space, avoiding a per-task reward. Learning the world once and solving tasks by planning is a central motivation for JEPA-style world models, and is the capability we evaluate.

However, the appeal of this method rests on the assumption that the learned embedding space is a high-quality representation of the world that captures the state relevant to the task. Further, the reported successes of latent world models are predominantly short-horizon: the goal lies a few steps away, where only a brief rollout of the predictor is required for success. Whether the same representations support long-horizon planning remains largely under investigated.

We study this question using OGBench PointMaze as a toy environment and OGBench Scene as a setting for long-horizon robotic control [17]. Our findings are as follows:

1. **Planning collapses at long horizons.** Sweeping start–goal offsets reveals a sharp regime beyond which sampling-based planning degrades to near-failure in both environments.

   The representation objective is fundamentally misspecified for control. Mechanistic analysis of LeWorldModel embeddings shows the failure lies in

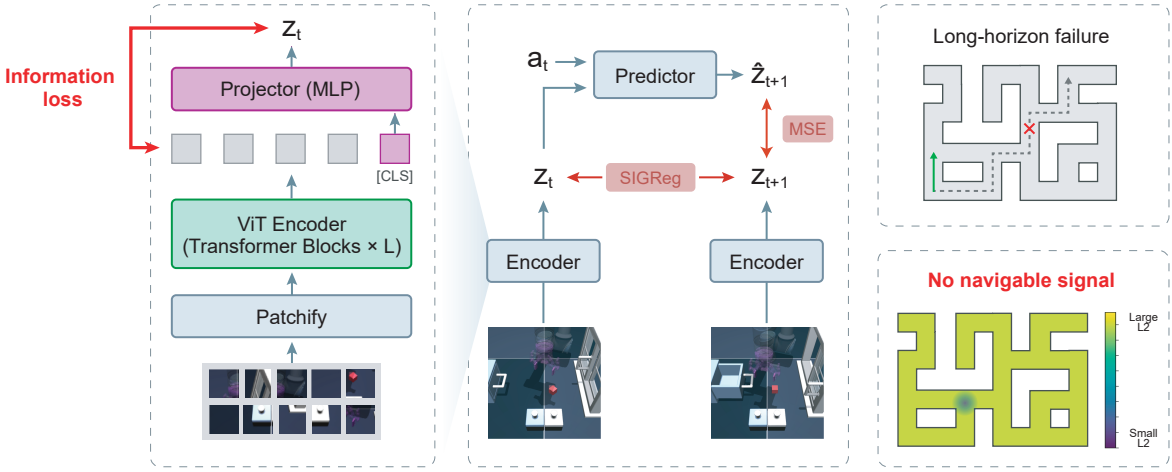

Figure 1. **LeWorldModel embeddings are misspecified for control.** *Center, adapted from [14]:* JEPA architecture, objective, and regularization. A ViT encoder maps an observation to an embedding z_t, a predictor forecasts $\hat{z}_{t+1}$ from z_t conditioned on action a_t, and the model is trained with a next-embedding MSE loss and the SIGReg isotropy penalty. *Left:* in complex environments, scene-relevant information is lost between the ViT patch tokens and the embedding z_t. *Right:* latent-space MPC collapses over long horizons (top) and the embedding L_2 cost landscape provides no navigable signal toward the goal (bottom).

the representation: in PointMaze, the task state is buried in a low-variance subspace invisible to L_2 cost, while in Scene, high-DOF states are dropped from the embedding altogether.

2. Related Work

Latent world models and JEPAs. Predicting in representation space rather than pixel space underlies image JEPAs [2] and video JEPAs [3]. A persistent challenge is representation collapse, classically addressed by asymmetric architectures, stop-gradients, or variance-covariance regularization [5, 7, 15]. Recent work introduces a sketched distributional penalty, SIGReg, that enables stable end-to-end training of a predictive encoder from scratch [4], which is the world model we study. Our contribution is orthogonal to training stability: we ask what the resulting representation contains and whether it is usable for planning.

World models for control. Reconstruction-based latent world models such as PlaNet and the Dreamer family [8, 9] plan or learn policies in a learned latent space, and TD-MPC [10, 11] learns a task-oriented latent via a reward and value signal. These methods couple the representation to either the observation, through a reconstruction decoder that forces it to retain the information needed to regenerate pixels, or to the task, through reward and value losses. JEPA-style planners instead use a task-agnostic embedding and a reward-free L_2 goal metric. We show this decoupling is precisely where long-horizon control breaks.

Planning as inference and goal-conditioned control. Solving tasks by optimizing actions against a learned model [20, 21] and specifying goals as images is an attractive, reward-free formulation. OGBench [16] provides offset-controllable goal-reaching tasks that we exploit to sweep planning horizons.

Probing representations. Linear probing [1] and decodability analyses are standard tools for asking what a representation encodes. We use probing to dissociate two failure modes: information that is present but metrically suppressed, versus information that is absent.

3. Preliminaries

The world model. An encoder f_θ maps an observation o_t to an embedding $z_t = f_\theta(o_t) \in \mathbb{R}^d$ and a future observation o_{t+1} to an embedding $z_{t+1} \in \mathbb{R}^d$. A predictor g_ϕ forecasts the next embedding from a short history of embeddings and actions, $\hat{z}_{t+1} = g_\phi(z_{t-k:t}, a_{t-k:t})$. Training minimizes a next-embedding prediction loss plus a Sketched-Isotropic-Gaussian Regularizer (SIGReg),

$$\mathcal{L} = ||\hat{z}_{t+1} - z_{t+1}||_2^2 + \lambda \text{SIGReg}(Z), \tag{1}$$

Crucially, no term in $\mathcal{L}$ references a task, reward, or notion of which dimensions of z are control-relevant.

SIGReg. SIGReg plays an important role in shaping the embedding space. In principle, SIGReg prevents trivial collapse and promotes feature diversity in the world model's embedding space [14]. Formally, let $Z \in \mathbb{R}^{N \times B \times d}$ denote the tensor of latent embeddings of dimension d collected over a history N and batch size B. SIGReg projects embeddings onto M random unit-norm directions $u^{(m)} \in \mathbb{S}^{d-1}$ and optimizes the univariate Epps-Pulley [6] test statistic

$T(\cdot)$ along the one-dimensional projections $h^{(m)} = Zu^{(m)}$.

$$\text{SIGReg(Z)} \triangleq \frac{1}{M} \sum_{m=1}^{M} T(h^{(m)}) \tag{2}$$

Planning. Given the current observation and a goal image o_g with embedding $z_g = f_\theta(o_g)$, control is MPC with the Cross-Entropy Method (CEM). The planner samples action sequences $a_{t:t+H}$, rolls each forward with g_ϕ to a predicted terminal embedding $\hat{z}_{t+H}$, and scores it by

$$c(a_{t:t+H}) = ||\hat{z}_{t+H} - z_g||_2, \tag{3}$$

the L_2 distance between the state and goal in embedding space. The best (elite) sequences are used to re-estimate the mean and variance of the sampling distribution (usually Gaussian). Iterating this concentrates the distribution on a low-cost action sequence. The planner then executes its first action and replans at the next observation. The first action of that sequence is taken and the whole process repeats. The cost c is the only signal coupling the representation to the task, and it weights every embedding dimension equally.

4. Methods

Environments. We use two OGBench settings. PointMaze is a 2D navigation task with low-dimensional underlying state (agent (x, y) position), which we use as a controlled probe: ground-truth task state is known and the cost landscape can be visualized directly. Scene is a tabletop manipulation environment with a drawer, a window, buttons, and a movable cube, requiring multi-step interaction and serving as our long-horizon control setting.

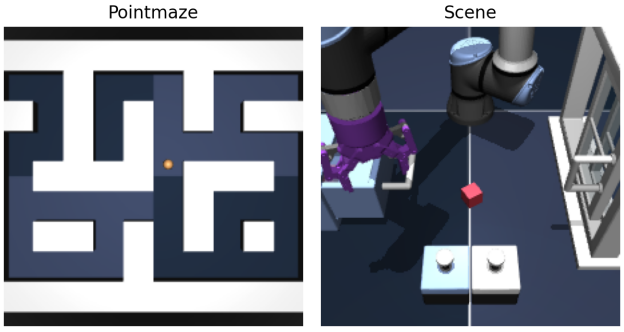

Figure 2. **OGBench environments.** We select OGBench PointMaze and OGBench Scene as settings for testing long-horizon control.

Training. We train the LeWM world model from pixels with the next embedding latent prediction plus the isotropic-Gaussian regularization term (SIGReg [4]). The image encoder is implemented as a Vision Transformer (ViT) that projects pixel inputs into a 512-dimensional embedding space. All models are implemented in PyTorch [18] and optimized using AdamW on a dataset consisting of 1 million transitions of play data collected within the target scene. We train the network for 100 epochs prior to evaluation.

Goal-offset sweep. To probe horizon, we construct goal-reaching tasks parameterized by the temporal offset Δ between the start state and the goal state along a reference trajectory, and report success rate as a function of Δ. The planning budget is scaled with offset (budget $= 3\Delta$) so that failure cannot be attributed to too few replanning steps.

Probing. To ask what the embedding encodes, we fit probes from frozen embeddings to ground-truth state variables (agent position in PointMaze; object and articulation states in scene) and report held-out R^2. We probe both the embedding z used by the planner and the spatial patch-token features before pooling, fitting linear (ridge) and nonlinear (MLP) probes with scikit-learn [19] to distinguish absent information from nonlinearly-encoded information. All figures are rendered with matplotlib [12].

Code. Our experiments build on the official LeWorld-Model codebase [14], which provides the world-model architecture, the SIGReg training objective, and the CEM-MPC planner.

5. Experiments

5.1. Long-horizon planning collapses

We first stress-test the planning horizon directly. For each goal offset $\Delta \in \{25, 50, 100, 150, 200\}$ steps, we sample start–goal pairs separated by Δ environment steps from held-out expert trajectories, so every goal is reachable and in-distribution. We plan with CEM-MPC (300 samples, 30 refinement iterations, receding-horizon replanning) using the best-validation checkpoint, allow a budget of 3Δ steps, and count an episode successful if the agent reaches the goal within that budget. We report the success rate over 50 episodes with Wilson 95% confidence intervals (Fig. 4).

Success degrades sharply and monotonically with offset in both environments. Because we scale planning budget with distance (3Δ, three times the demonstration length), the collapse cannot be a fixed step cap starving distant goals. Planning also succeeds at $\Delta = 25$ in both domains, so the dynamics, the optimizer, and low-level control are each functional in isolation; only their composition over a longer horizon breaks. With the model, planner, environment, success criterion, and budget-to-distance ratio held fixed across offsets, and the same collapse arising in both a navigation and a manipulation domain, we attribute it to the planning objective at increasing horizon rather than to any particular goal, planner, or domain.

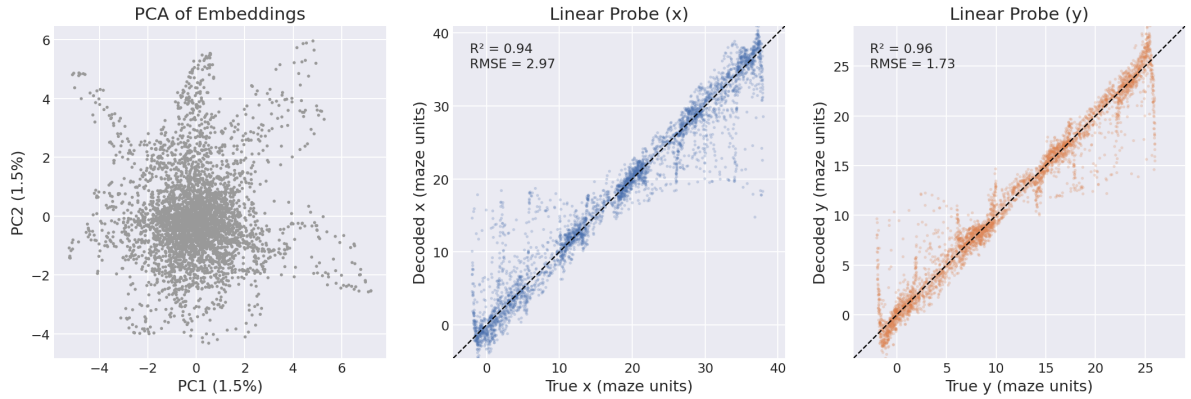

Figure 3. **Euclidean position is linearly represented in world model embeddings.** Despite a near-isotropic embedding (top-2 PCs caputure 3% of variance, left), a linear probe recovers agent position from the same space ($R^2 = 0.94$ for x, 0.96 for y; center, right). The information is present even though the raw L_2 cost is flat.

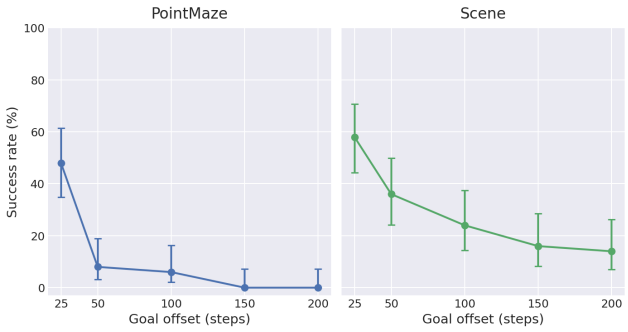

Figure 4. **Goal offset sweeps.** Success rate versus the start–goal offset Δ for PointMaze (left) and Scene (right), with the planner budget scaled as 3Δ; error bars are Wilson 95% CIs over 50 episodes. Success collapses sharply and monotonically with offset in both environments, reaching near zero in PointMaze and 14% in Scene by the largest offsets.

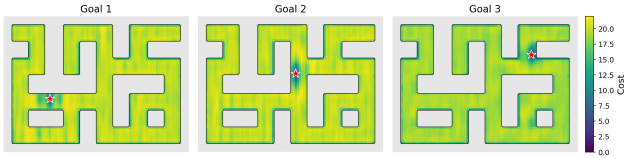

Figure 5. **PointMaze cost landscape.** Embedding cost $\|z_s - z_g\|_2$ over all free cells for three goals (stars). The same flat pattern occurs for every goal: cost stays near its maximum almost everywhere and dips only in a narrow well at the goal, leaving CEM with near-identical scores across samples and no signal to concentrate toward the target.

5.2. CEM cost is uninformative

We visualize the planner's cost landscape directly. For three goals, we render the agent at a dense grid covering all free cells, encode each state, and plot the embedding cost $\|z_s - z_g\|_2$ over the maze (Fig. 5). As a natural reference, we use the geodesic (shortest-path) distance to the goal: it respects the walls and decreases monotonically along every optimal route, so a cost landscape an agent can descend should track it.

Instead, the field is nearly uniform: cost is high and roughly constant everywhere, with only a small depression at the goal itself. The rank correlation with geodesic distance is $\rho = -0.01 \pm 0.01$ (mean and 95% CI over 64 randomly sampled goals), statistically indistinguishable from zero, and the three maps shown are representative (Fig. 5). The cost therefore carries no usable gradient toward the goal: its only structure is the shallow basin at the goal, which a sampling-based planner can descend only once the goal already lies within a short rollout. This is a key symptom of the mechanistic causes of poor long-horizon performance.

5.3. Position is linearly represented

A flat cost might suggest position is simply absent from the representation. We test this by training a linear probe to recover the agent's (x, y) position from world-model embeddings (Fig. 3). Two findings stand in tension. A 2D PCA of the embedding does not organize states by position: the top two principal components capture only 3% of the variance, reflecting the near-isotropy encouraged by the SIGReg term. Yet a single linear map decodes the agent's position with held-out $R^2 = 0.95$ (per-coordinate $R^2 = 0.94$ in x and 0.96 in y). Position is thus linearly present in the very embedding the planner uses, even though it occupies a low-variance subspace invisible to a 2D PCA. The flat cost is therefore not a missing-information problem: raw L_2 distance over the embedding simply fails to reflect the position the embedding encodes.

5.4. Linear probes recover a navigable cost

To separate "decodable" from "used by L_2", we compare three costs computed from the same states: the plan-

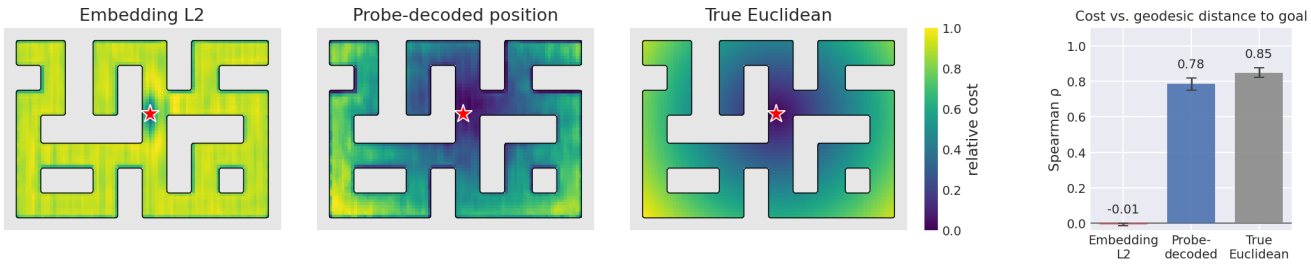

Figure 6. **Position readout recovers a navigable cost.** For a fixed goal (red star), each maze panel shows the relative cost assigned to every free cell under three readouts: the planner's raw embedding L_2 $\|z_s - z_g\|_2$ (left), the L_2 distance between linear probe-decoded positions (center), and the true Euclidean distance (right). *Far right:* Spearman correlation between each cost and geodesic (shortest-path) distance to the goal over 64 randomly sampled goals (error bars are 95% CIs). Raw embedding L_2 is uncorrelated with geodesic distance.

ner's raw embedding L_2, the L_2 between linearly decoded positions, and the true Euclidean distance, which serves as an upper bound for any Euclidean cost computed from position. For each, we measure the Spearman correlation with geodesic distance over 64 randomly sampled goals (Fig. 6). Raw embedding L_2 is flat ($\rho = -0.01 \pm 0.01$), but probe-decoded position is highly associated with geodesic distance ($\rho = 0.78 \pm 0.03$) and approaches the true Euclidean ceiling ($\rho = 0.85 \pm 0.03$). Motivated by this highly linear auxiliary cost, we further conduct a causal intervention comparing CEM planning via the standard L_2 cost to CEM via probe-decoded distance (Fig. 7).

We replace the raw embedding cost c with the L_2 distance between probe-decoded positions while holding the encoder, predictor, and CEM budget-to-distance ratio fixed. We find that this restores planning across the entire horizon range (Fig. 7). Probe-decoded planning sustains a success rate between roughly 65% and 90% for every offset $\Delta \in \{25, \ldots, 200\}$, in stark contrast to the raw embedding cost. Because both planners share the same latent rollouts and differ only in the readout used to score them, the long-horizon failure cannot be attributed to predictor drift or to an insufficient sampling budget.

5.5. Information collapse in OGBench Scene

Prompted by our mechanistic investigation of PointMaze, we extend the probing analysis to OGBench Scene, a more realistic embodied-control environment with substantially higher state dimensionality. In PointMaze, since state embeddings already represented position linearly, probing embeddings was enough to pinpoint the failure to the planning metric. We begin our analysis of Scene the same way, probing only the cost-space embedding used by the predictor and the planner. The drawer, window, and buttons are recovered almost perfectly (Fig. 8 at held-out $R^2 = 0.96$, 0.97, 0.97), but the cube and the arm are not ($R^2 = 0.41$ and 0.38). The split reflects the degrees of freedom of each object (Table 1), where the recovered objects are one-DOF articulations and the embedding fails to linearly encode high-DOF entities like the free-moving cube (3 translational DOFs with high variance and 3 rotational) and the arm (7 DOF).

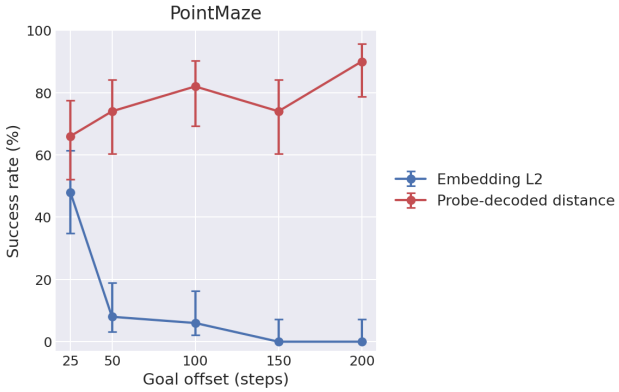

Figure 7. **Linear probes recover CEM planning**. Success rate vs goal offset (Wilson 95% CI, $n = 50$). Swapping the planner's cost from raw embedding L_2 (blue) to probe-decoded position distance (red) lifts success from 0% to $66 - 90\%$ across all offsets.

Object	DOF	State variance	Probe R^2
Arm	7	3.04 rad^2	0.38
Cube	6	258 cm^2	0.41
Drawer	1	41 cm^2	0.96
Window	1	66 cm^2	0.97
Buttons	2	0.50 (binary)	0.97

Table 1. **Object-state variance versus cost-space encodability.** The high-DOF objects (arm, cube) are exactly those the encoder fails to represent (low linear-probe R^2), while the low-DOF slides and binary buttons are recovered nearly perfectly. Variance is in each object's native units and is not comparable across rows; DOF is the unit-free axis.

Unlike PointMaze, task-relevant state information is missing from the world model's learned embeddings. This raises a sharper question: does the encoder represent the

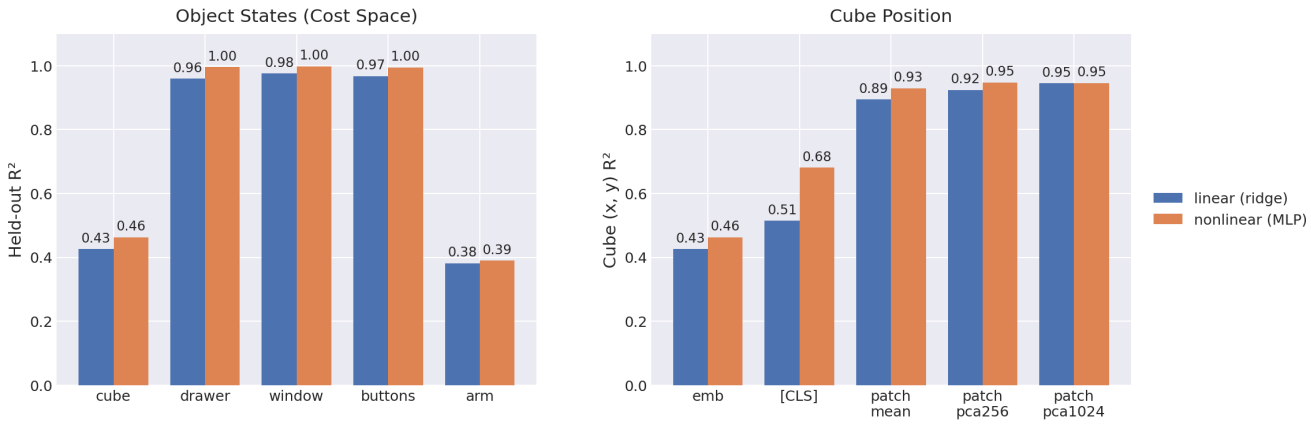

Figure 8. **OGBench-Scene representation probing.** Held-out R^2 for linear (ridge) and nonlinear (MLP) probes. *Left:* probes on the cost-space embedding. The low-DOF drawer, window, and buttons are recovered near-perfectly, but the high-DOF cube and arm decode far worse ($R^2 \approx 0.4$). The MLP barely beats the linear probe, indicating missing information rather than nonlinear encoding. *Right:* cube (x, y) is weakly represented in the embedding and the $[CLS]$ token, but strongly recoverable from the patch tokens ($R^2 \approx 0.9$).

high-DOF states anywhere, or not at all? Inspired by this question, we extend to linear and nonlinear (MLP) probes (Fig. 8), and analyze representations at the $[CLS]$ and patch tokens within the ViT backbone. Interestingly, cube position is recoverable from the patch tokens at $R^2 \approx 0.9$ (Fig. 8, right), with linear and nonlinear probes nearly tied. Thus, the information exists upstream of the embeddings, but does not survive into the embedding the objective shapes and the planner consumes. This is information collapse in the precise sense: the embedding distribution is Gaussian by construction under SIGReg, yet it carries little information about the high-DOF factors of the state.

We hypothesize that this mechanism represents a degenerate solution to the predictive objective. Specifically, the next-embedding loss term $\|\hat{z}_{t+1} - \text{sg}(z_{t+1})\|^2$ can be trivially minimized by omitting hard-to-predict features from the representation entirely rather than learning to model them. Because the high-DOF cube and robotic arm exhibit the most complex and least predictable dynamics, the most efficient path for the model to reduce prediction error is to cease encoding them altogether, while retaining only the near-deterministic, low-DOF components.

6. Conclusion

Our experiments trace the failure of LeWorldModel planning to its representation objective. The next-embedding loss is task-agnostic: it references no reward and no notion of which dimensions are control-relevant, so it is minimized as readily by discarding hard-to-predict factors as by modeling them. The high-DOF cube and arm carry the least predictable dynamics, making the cheapest way to lower the loss along their directions to stop encoding them. SIGReg does not cause this so much as conceal it: by holding the embedding isotropic and Gaussian it keeps the covariance full-rank, so the loss of control-relevant information leaves the spectrum looking healthy and escapes standard collapse diagnostics.

The failure surfaces in two forms that demand different remedies. In PointMaze, the control-relevant state survives in the embedding but occupies a low-variance subspace the uniform L_2 cost drowns out; here a better readout suffices, as our probe-decoded planner demonstrates. In Scene, the high-DOF state is absent from the embedding altogether and retained only in patch tokens, so no reweighting of that embedding can recover it. Preventing this requires the objective itself to incentivize control-relevance rather than solely predictability.

Several modifications to the objective could supply that incentive. A temporal-difference value signal would bind the representation to control directly: factors that shape predicted return must be encoded in order to predict it, forcing the model to keep control-relevant state even where its dynamics are hard to model. Alternatively, an information-preserving objective could address the representational weaknesses, whether by reconstructing the observation, by predicting the action that links consecutive states, or by enforcing a mutual-information floor between the embedding and the observation. We leave these to future work.

7. Contributions & Acknowledgements

William Peng and **Oscar Rodriguez** led problem formulation, implementation, experiments, analysis, writing, and poster preparation.

Generative AI tools were used during this project. We used Claude for code debugging, code generation, formatting in LaTeX, and brainstorming experiment design. All experimental results, analysis, and final claims were produced and verified by the authors.

References

[1] G. Alain and Y. Bengio. Understanding intermediate layers using linear classifier probes, 2018. 2

[2] Assran, M. Duval, V. Bojanowski, L. Rabbat, and Ballas. Self-supervised learning from images with a joint-embedding predictive architecture, 2023. 2

[3] M. Assran, A. Bardes, D. Fan, Q. Garrido, R. Howes, Mojtaba, Komeili, M. Muckley, A. Rizvi, C. Roberts, K. Sinha, A. Zholus, S. Arnaud, A. Gejji, A. Martin, F. R. Hogan, D. Dugas, P. Bojanowski, V. Khalidov, P. Labatut, F. Massa, M. Szafraniec, K. Krishnakumar, Y. Li, X. Ma, S. Chandar, F. Meier, Y. LeCun, M. Rabbat, and N. Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. 2

[4] R. Balestriero and Y. LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics, 2025. 1, 2, 3

[5] A. Bardes, J. Ponce, and Y. LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning, 2022. 2

[6] T. W. Epps and L. B. Pulley. A test for normality based on the empirical characteristic function. *Biometrika*, 70(3):723–726, 1983. 2

[7] Grill, A. Florian, R. Tallec, D. Buchatskaya1, Z. Avila, G. Guo, P. Bilal, M. Kavukcuoglu, and Valko. Bootstrap your own latent a new approach to self-supervised learning, 2020. 2

[8] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent dynamics for planning from pixels, 2019. 2

[9] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models, 2024. 2

[10] N. Hansen, H. Su, and X. Wang. Td-mpc2: Scalable, robust world models for continuous control, 2024. 2

[11] N. Hansen, X. Wang, and H. Su. Temporal difference learning for model predictive control, 2022. 2

[12] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007. 3

[13] Y. LeCun. A path towards autonomous machine intelligence. 2022. 1

[14] L. Maes, Q. L. Lidec, D. Scieur, Y. LeCun, and R. Balestriero. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. 2, 3

[15] Oquab, M. Darcet, V. Huy, K. Szafraniec, H. Fernandez, E.-N. Massa, B. Assran, H. Galuba, L. Huang, R. Misra, S. Sharma, J. Xu, L. Mairal1, Joulin, and Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. 2

[16] S. Park, K. Frans, B. Eysenbach, and S. Levine. Ogbench: Benchmarking offline goal-conditioned rl, 2025. 2

[17] S. Park1, K. Frans1, B. Eysenbach2, and S. Levine. Understanding intermediate layers using linear classifier probes, 2024. 1

[18] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. 3

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, A. Müller, J. Nothman, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python, 2018. 3

[20] R. Rubinstein. The Cross-Entropy Method for Combinatorial and Continuous Optimization. *Methodology And Computing In Applied Probability*, 1(2):127–190, Sept. 1999. 1, 2

[21] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou. Aggressive driving with model predictive path integral control. pages 1433–1440, 2016. 2