# DRIVE: DECOUPLED REPRESENTATION WITH INFORMED VALUE ESTIMATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Model-based reinforcement learning (MBRL) promises sample-efficient continuous control, yet existing methods face a fundamental tension between representation quality and training stability. Reconstruction-based approaches allocate encoder capacity to value-irrelevant features, while joint dynamics-value supervision lets noisy bootstrapped gradients distort the learned latent geometry. We introduce **DRIVE** (**D**ecoupled **R**epresentation with **I**nformed **V**alue **E**stimation), a framework built on one principle: *representation learning should be decoupled from value-driven optimization*. DRIVE isolates the encoder from all critic and actor gradients, training it exclusively through structured supervision, including dynamics prediction, contrastive temporal consistency, reward estimation, and cumulative return prediction on the unit hypersphere. The critic fits values on the resulting well-conditioned latent space, and the actor combines single-step Q-gradients with a multi-step *soft-decay rollout* through the learned dynamics, internalizing trajectory-level planning at training time with no additional inference cost. Across 28 high-dimensional continuous control tasks, DRIVE matches or exceeds state-of-the-art model-based and model-free baselines. Ablation studies confirm that the soft-decay rollout and structured encoder supervision provide complementary gains. Crucially, our analysis reveals a fundamental failure mode in high-dimensional MBRL: local representation aliasing inevitably leads to Q-value flattening and actor gradient starvation, a pathology that DRIVE effectively cures.

## 1 INTRODUCTION

Model-based reinforcement learning (MBRL) has emerged as a promising paradigm, improving sample efficiency through learned dynamics models (Hafner et al., 2025; Hansen et al., 2024). However, as state and action spaces grow in dimensionality, such as in humanoid locomotion and dexterous manipulation, a fundamental question arises: *what should the world model learn?*

Current MBRL methods implicitly answer this question in two divergent ways. Reconstruction-based approaches such as Dreamer (Hafner et al., 2019; 2025) train the encoder to reconstruct raw observations, treating all state dimensions equally regardless of their relevance to downstream value estimation. In high-dimensional settings, this forces the encoder to allocate capacity to value-irrelevant features, a limitation that becomes more severe as observation dimensionality grows. The resulting representations, while faithful to the observation space, are poorly suited for decision-making.

Planning-based approaches take a different path: they learn compact latent dynamics models supervised jointly by dynamics, reward, and value losses. TD-MPC2 (Hansen et al., 2024) operates in continuous action spaces while EfficientZero (Ye et al., 2021; Wang et al., 2024) targets discrete domains, yet both share two structural limitations. First, joint supervision lets noisy bootstrapped value gradients propagate into the encoder, distorting a latent geometry that the dynamics loss has shaped—a *signal quality mismatch* that worsens as observation dimensionality grows. Second, both rely on expensive inference-time search (MPPI and MCTS, respectively) that scales poorly with the action space. If the learned representation were sufficiently structured for direct policy optimization, why would search be necessary at all? Subsequent distillation methods (Lin et al., 2025; Wang et al., 2025; Zhan et al., 2025) lend support to this view: by distilling the planner's output into a reactive policy, they achieve performance gains that implicitly confirm a gap between the representation's quality and the downstream task's demands—a gap that search was compensating for. DRIVE



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

targets this gap directly: by improving upstream representation quality through structured encoder supervision, the actor can optimize effectively through the latent space without search.

Despite their divergent designs, both paradigms share a common shortcoming: *neither produces representations that are both structurally stable and value-aware enough for direct policy optimization.* Dreamer's encoder captures observation structure but not value structure; TD-MPC2's encoder conflates both and corrupts the former.

We propose **DRIVE** (**D**ecoupled **R**epresentation with **I**nformed **V**alue **E**stimation), a model-based framework built on a single architectural principle: *representation learning should be decoupled from value-driven optimization*, with each component performing one well-defined role. The encoder is trained exclusively with direct supervision losses, including dynamics prediction, reward estimation, contrastive temporal consistency, and cumulative return prediction. All representations are projected onto the unit hypersphere to establish a unified metric space, and no gradients are received from the critic or actor. The critic performs value fitting on the resulting well-conditioned latent space. The actor leverages two complementary signals: a single-step Q-gradient for stable local improvement, and a multi-step soft-decay rollout through the learned model for trajectory-level planning. The soft-decay mechanism attenuates gradients as model predictions become less reliable, transitioning to Q-based terminal estimation. This yields a learned model-predictive control scheme without online search and with zero inference overhead. Our main contributions are:

- **Gradient-isolated encoder-actor-critic architecture.** A decoupled architecture in which the encoder receives no gradients from either the critic or the actor, preventing noisy bootstrapped value signals from distorting the learned representation geometry and addressing the signal quality mismatch in jointly trained MBRL systems.
- **Value-informed geometric representation learning.** Beyond standard dynamics and reward prediction, we introduce InfoNCE-based contrastive learning that prevents representation collapse by requiring *identity discrimination* across the batch on the unit hypersphere, and a cumulative return prediction head that grounds the latent geometry in value structure.
- **Soft-decay rollout for learned model-predictive control.** We propose a differentiable rollout mechanism with exponential gradient decay that bridges model-based planning and value-based estimation. Unlike BPTT (Dreamer) which suffers from compounding gradient errors, or gradient-free search (TD-MPC2) which discards model differentiability, soft-decay rollout internalizes multi-step planning at training time with zero additional inference cost.

## 2 Related Work

**World models for continuous control.** Model-based reinforcement learning leverages learned dynamics to improve sample efficiency. Dreamer (Hafner et al., 2025) learns a latent world model through observation reconstruction and optimizes policies via imagined rollouts, but the reconstruction objective allocates encoder capacity to value-irrelevant features as dimensionality grows (Hafner et al., 2025). TD-MPC2 (Hansen et al., 2024) replaces reconstruction with joint dynamics, reward, and value supervision in a compact latent space, yet this creates a signal quality mismatch: high-bandwidth dynamics gradients dominate the latent geometry while noisy bootstrapped value signals can destabilize it. TD-MPC2 further relies on MPPI sampling at inference time, discarding gradient information that a differentiable model could provide; subsequent distillation approaches (Lin et al., 2025; Wang et al., 2025; Zhan et al., 2025) improve warm-start quality but leave the encoder's mixed-gradient issue unaddressed. In the discrete-action setting, EfficientZero (Ye et al., 2021) shows that auxiliary self-supervised representation losses improve sample efficiency, though its reliance on MCTS incurs high computational cost. In the visual domain, JEPA-based world models (Nam et al., 2026; Maes et al., 2026) address representation collapse through distributional regularization, confirming that anti-collapse mechanisms are essential across modalities. DRIVE pursues an analogous anti-collapse goal in the state-based setting: by isolating the encoder from value gradients entirely and projecting representations onto a unit hypersphere with contrastive supervision, it supports direct policy optimization through differentiable rollouts without online search.

**Representation learning in reinforcement learning.** Self-supervised objectives have been widely adopted to improve sample efficiency in RL, including contrastive methods such as CURL and

SPR (Laskin et al., 2020; Schwarzer et al., 2021) and statistical regularization such as VICReg (Bardes et al., 2022). These approaches improve data efficiency but are primarily designed for stationary data distributions and do not explicitly shape representations around value-relevant structure. This criterion is formalized by the Value Equivalence Principle (Grimm et al., 2020), which argues that world models need only preserve transitions relevant to the induced Bellman operators. A parallel line of work addresses the optimization challenges posed by RL's non-stationary data stream. CrossQ (Bhatt et al., 2024) applies batch normalization to the SAC (Haarnoja et al., 2018) to reduce variance under shifting target distributions; BRO (Nauman et al., 2024) scales network capacity with regularization for compute- and sample-efficient control; XQC (Palenicek et al., 2026) combines batch normalization, weight normalization, and distributional cross-entropy loss, showing that well-conditioned critic optimization naturally bounds gradient norms under non-stationary bootstrapping; and SimBa-v2 (Lee et al., 2025) projects representations onto the unit hypersphere via weight normalization, stabilizing high update-to-data ratio training. Recent work further integrates representation learning with RL-specific objectives. TCRL (Zhao et al., 2023) introduces temporal contrastive objectives with a decoupled encoder, and MRQ (Fujimoto et al., 2025) pairs a decoupled state-action encoder with direct supervision losses to achieve strong multi-task performance, primarily demonstrated on low-dimensional benchmarks. DRIVE builds on both threads: it enforces strict gradient isolation between the encoder and the critic, projects all latent codes onto the unit hypersphere to unify the metric space across dynamics, contrastive, and rollout objectives (Appendix B), and channels value information to the encoder through differentiable actor rollouts rather than through the critic loss.

**Actor optimization in model-based RL.** How the actor exploits learned dynamics defines a spectrum of approaches. At one end, Dreamer and SVG (Amos et al., 2021) backpropagate through imagined rollouts (BPTT), providing rich gradient information but suffering from compounding errors that necessitate aggressive gradient clipping and short horizons. At the other end, TD-MPC2 and PETS (Chua et al., 2018) rely on sampling-based planning (CEM, MPPI), which avoids gradient instability but scales poorly with action dimensionality and incurs inference-time cost. C51 (Bellemare et al., 2017) recasts value regression as classification over a discrete support, providing well-conditioned gradients on the probability simplex, a principle DRIVE applies to reward and cumulative return prediction in the encoder. Geometric action-space constraints (Lin, 2026) bound the policy's Lipschitz constant, limiting out-of-distribution queries to the critic. DRIVE's soft-decay rollout occupies a middle ground: differentiable multi-step planning with geometrically attenuating gradient flow ($\alpha^h$ at step $h$), exploiting model predictions where reliable and deferring to Q-based terminal estimation where they are not (Appendix E), with zero additional inference cost.

## 3 METHOD

We present DRIVE, a model-based reinforcement learning framework built on a single architectural principle: *representation learning should be decoupled from value-driven optimization*. Each component is assigned a well-defined role in the learning pipeline (Figure 1). DRIVE consists of three decoupled components: 1) an encoder trained exclusively with direct supervision losses to produce smooth and informative latent representations; 2) a critic that performs value fitting on the latent space without propagating gradients to the encoder; and 3) an actor trained with two signals, single-step Q-gradients and multi-step soft-decay rollouts through the learned model.

We consider the standard reinforcement learning setting formulated as a Markov decision process $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S} \subseteq \mathbb{R}^n$ is the state space, $\mathcal{A} \subseteq \mathbb{R}^m$ is the continuous action space, $P(s'|s,a)$ is the transition dynamics, $r(s,a)$ is the reward function, and $\gamma \in (0,1)$ is the discount factor.

The central design choice in DRIVE is strict gradient isolation: the encoder provides representations *to* the critic and actor but never receives gradients *from* them. This allows the encoder to act as a *representation smoother*, transforming the raw high-curvature state manifold into a well-conditioned latent space where Bellman-based value fitting is stable (formalized in Appendix F).

This separation is motivated by the value equivalence principle (Grimm et al., 2020): the encoder defines the model class $\mathcal{M}$, while the critic defines the value function class $\mathcal{V}$. Joint optimization risks conflating the model with its evaluation criterion; gradient isolation ensures that the encoder's representational geometry is shaped solely by structured supervision.

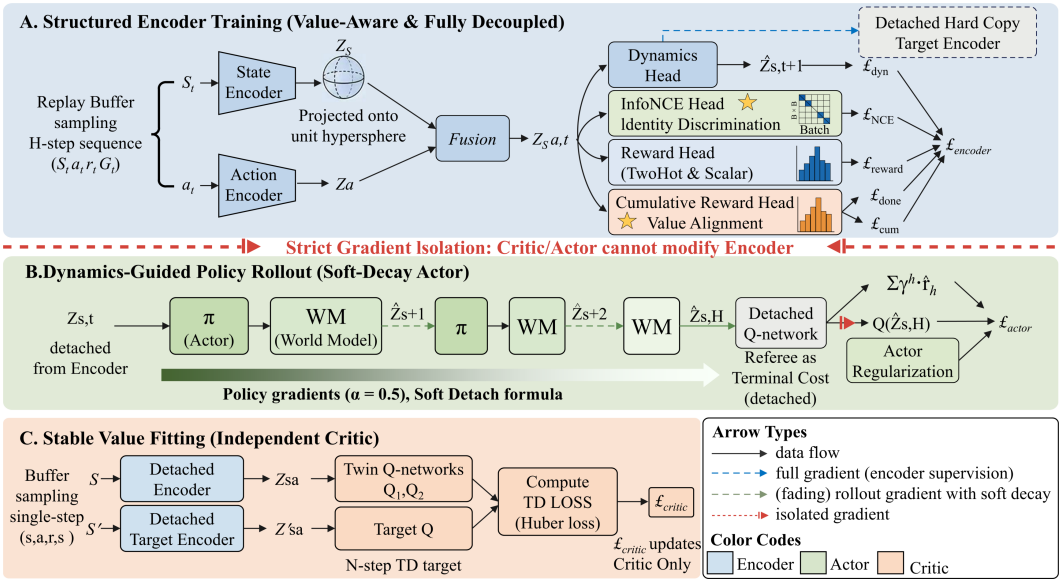



Figure 1: **DRIVE architecture. (A)** Encoder maps observations to the unit hypersphere, trained by five losses (dynamics, InfoNCE, reward, done, cumulative return) with gradient isolation from actor and critic. **(B)** Actor combines single-step Q-gradients with soft-decay rollout guidance ($\alpha$=0.5). **(C)** Critic fits twin Q-networks on detached encoder representations via $n$-step Huber TD loss.

Concretely, the encoder $f_\theta$ maps observations to latent states on the unit hypersphere:

$$z_t^s = \frac{\tilde{z}_t^s}{\|\tilde{z}_t^s\|_2}, \qquad \tilde{z}_t^s = \mathrm{MLP}_\theta(s_t), \tag{1}$$

so that $z_t^s \in \mathcal{S}^{d_s-1}$, where $d_s$ is the latent state dimension. This normalization provides a unified geometric foundation: the dynamics, contrastive, and rollout objectives operate under the same cosine metric on the sphere, ensuring geometric consistency across supervision signals (Appendix B).

Actions are projected to a latent space $z_t^a = g(a_t) \in \mathbb{R}^{d_a}$, where $g$ is a linear projection, and fused with the state representation via a three-layer MLP $h$ to form the joint representation $z_t^{sa} = h(z_t^s, z_t^a) \in \mathbb{R}^{d_{sa}}$. The actor $\pi_\phi(z_t^s)$ and critic $Q_\psi(z_t^{sa})$ consume encoder outputs through stop-gradient connections:

$$\pi_\phi(\mathtt{sg}(z_t^s)), \qquad Q_\psi(\mathtt{sg}(z_t^{sa})), \tag{2}$$

where $\mathtt{sg}(\cdot)$ denotes the stop-gradient operator.

### 3.1 Structured Encoder Training

The encoder is trained by unrolling an $H$-step autoregressive trajectory in latent space. Given a trajectory segment $(s_t, a_t, r_t, d_t, \ldots, s_{t+H})$ from the replay buffer, we encode $z_t^s = f_\theta(s_t)$ and predict forward:

$$\hat{z}_{t+h+1}^s,\ \hat{r}_{t+h},\ \hat{d}_{t+h},\ \hat{R}_{t+h}^{\mathrm{cum}} = \mathtt{WM}_\theta(\hat{z}_{t+h}^s,\ a_{t+h}), \quad h = 0, \ldots, H-1, \tag{3}$$

where $\hat{z}_t^s \triangleq z_t^s$ and $\mathtt{WM}_\theta$ represents the world model, which comprises four prediction heads: dynamics ($\hat{z}^s$), distributional reward ($\hat{r}$), termination ($\hat{d}$), and cumulative return ($\hat{R}^{\mathrm{cum}}$), sharing a common state-action representation $z^{sa}$. A fifth scalar reward head $\hat{r}^{\mathrm{sc}} = \mathtt{WM}_\theta^{\mathrm{r_sc}}(z^{sa})$ is trained during the encoder phase via Smooth-L1 loss against ground-truth rewards ($\lambda_{\mathrm{r_sc}} = 0.1$) and consumed read-only by the actor rollout (Section 3.2), where a scalar output avoids softmax Jacobian instability during multi-step backpropagation. A target encoder $f_{\bar{\theta}}$ (hard-copied every $K$ steps) provides supervision targets $\bar{z}_{t+h+1}^s = f_{\bar{\theta}}(s_{t+h+1})$. All losses are weighted by a trajectory mask $m_h = \prod_{k=0}^{h-1}(1 - d_{t+k})$ that zeroes out post-termination steps.

**Dynamics loss.** Since predicted and target latent states lie on the unit hypersphere, the dynamics loss reduces to cosine distance:

$$\mathcal{L}_{\text{dyn}} = \sum_{h=0}^{H-1} m_h \left(1 - \cos(\hat{z}^s_{t+h+1},\ \bar{z}^s_{t+h+1})\right). \tag{4}$$

This loss serves as a causal inductive bias enforcing local temporal consistency rather than accurate long-horizon prediction. However, dynamics prediction alone is insufficient: an encoder that maps all states to a single point trivially minimizes cosine distance.

**Contrastive loss.** To prevent representational collapse, we add an InfoNCE objective whose logits are cosine similarities—sharing the same metric as the dynamics loss:

$$\mathcal{L}_{\text{NCE}} = -\sum_{h=0}^{H-1} m_h \frac{1}{B} \sum_{i=1}^{B} \log \frac{\exp\big(\cos(\hat{z}^s_{t+h+1,i},\ \bar{z}^s_{t+h+1,i})/\tau\big)}{\sum_{j=1}^{B} \exp\big(\cos(\hat{z}^s_{t+h+1,i},\ \bar{z}^s_{t+h+1,j})/\tau\big)}, \tag{5}$$

where $B$ is the batch size and $\tau$ is the temperature. The dynamics loss attracts predictions toward targets; InfoNCE adds repulsion that requires each prediction to be uniquely identifiable among batch elements, jointly forming a complete force field on the sphere (Appendix B; formal gradient analysis in Appendix C). Unlike augmentation-based contrastive methods (Laskin et al., 2020), dynamics prediction provides natural positive pairs without data augmentation.

**Reward and termination prediction.** To ground the latent space in task-relevant information, rewards are predicted via TwoHot cross-entropy over symexp-spaced bins (Hafner et al., 2025):

$$\mathcal{L}_{\text{rew}} = -\sum_{h=0}^{H-1} m_h \sum_{k=1}^{N_b} \text{TwoHot}(r_{t+h})_k \log \text{softmax}(\hat{r}_{t+h})_k. \tag{6}$$

The distributional formulation provides well-conditioned gradients on the probability simplex regardless of reward scale, unlike MSE, whose gradient magnitude is proportional to prediction error and vanishes near convergence, weakening the encoder's supervisory signal precisely when representations stabilize (Appendix D). Episode termination is predicted via MSE: $\mathcal{L}_{\text{done}} = \sum_h m_h \|\hat{d}_{t+h} - d_{t+h}\|_2^2$.

**Cumulative return prediction.** The dynamics and reward losses define equivalence with respect to single-step Bellman operators (Grimm et al., 2020). To extend this to multi-step value functions, we add a cumulative return head using the same TwoHot (Hafner et al., 2025) cross-entropy formulation:

$$\mathcal{L}_{\text{cum}} = -\sum_{h=0}^{H-1} m_h \sum_{k=1}^{N_b} \text{TwoHot}\big(\text{symlog}(R^{\text{cum}}_{t+h})\big)_k \log \text{softmax}(\hat{R}^{\text{cum}}_{t+h})_k, \tag{7}$$

where $R^{\text{cum}}_{t+h} = \sum_{k=h}^{H-1} \gamma^{k-h} r_{t+k} \prod_{l=h}^{k-1}(1 - d_{t+l})$ and $N_b$ is the number of bins. This forces the encoder to retain features with long-horizon value that dynamics prediction alone may discard.

**Combined objective.** The total encoder loss is:

$$\mathcal{L}_{\text{enc}} = \lambda_{\text{dyn}}\mathcal{L}_{\text{dyn}} + \lambda_{\text{NCE}}\mathcal{L}_{\text{NCE}} + \lambda_{\text{rew}}\mathcal{L}_{\text{rew}} + \lambda_{\text{done}}\mathcal{L}_{\text{done}} + \lambda_{\text{cum}}\mathcal{L}_{\text{cum}}. \tag{8}$$

The encoder trains in concentrated phases: every $K$ environment steps, the target encoder is hard-copied from the online encoder, followed by $K$ consecutive updates on fresh replay data, decoupling representation evolution from the actor-critic cycle.

### 3.2 Soft-Decay Rollout for Actor Guidance

DRIVE trains the actor through two complementary gradient channels at different temporal scales.

**Single-step Q-gradient.** We first describe the critic objective that provides the Q-gradient signal. The critic networks $Q_{\psi_1}, Q_{\psi_2}$ are trained on the encoder's joint representation $z^{sa}_t$ using clipped double Q-learning with $n$-step returns:

$$\mathcal{L}_Q = \sum_{k=1}^{2} \ell_\beta\Big(Q_{\psi_k}\big(\text{sg}(z^{sa}_t)\big),\ \sum_{i=0}^{n-1} \gamma^i r_{t+i} + \gamma^n (1 - d_{t+n}) \min_{j=1,2} Q_{\bar{\psi}_j}\big(\bar{z}^{sa}_{t+n}\big)\Big), \tag{9}$$

where $\bar{z}^{sa}_{t+n} = h\big(f_{\bar{\theta}}(s_{t+n}),\ \pi_{\bar{\phi}}(f_{\bar{\theta}}(s_{t+n}))\big)$ denotes the target representation computed using the target encoder $f_{\bar{\theta}}$ and target actor $\pi_{\bar{\phi}}$ and $\ell_\beta$ is the Huber loss (Huber, 1992) with threshold $\beta$. The single-step actor gradient maximizes the local Q-value:

$$\mathcal{L}_{\text{Q-actor}} = -Q_{\psi_1}\big(h(\mathrm{sg}(z^s_t),\ \pi_\phi(\mathrm{sg}(z^s_t)))\big). \tag{10}$$

**Multi-step soft-decay rollout.** To provide trajectory-level planning, the actor unrolls a differentiable $H$-step rollout through the learned dynamics with exponentially attenuating gradient flow:

$$\begin{aligned}
\hat{z}^s_{t+1} &= \mathtt{WM}^{\text{dyn}}_\theta(\mathrm{sg}(z^s_t),\ \pi_\phi(\mathrm{sg}(z^s_t))),\\
\tilde{z}^s_{t+h} &= \alpha \cdot \hat{z}^s_{t+h} + (1-\alpha)\cdot \mathrm{sg}(\hat{z}^s_{t+h}), \quad h = 1, \dots, H,\\
\hat{z}^s_{t+h+1} &= \mathtt{WM}^{\text{dyn}}_\theta(\tilde{z}^s_{t+h},\ \pi_\phi(\tilde{z}^s_{t+h})),
\end{aligned} \tag{11}$$

where $\alpha \in [0, 1]$ controls gradient retention. The decomposition leaves the forward pass unchanged, with both components sharing the same value, and scales the effective gradient to $\alpha^h$ at step $h$ in the backward pass (formal analysis in Appendix E). Each predicted state is $\ell_2$-normalized to remain on the unit sphere before decomposition.

The rollout accumulates scalar rewards and terminates with a detached Q-estimate:

$$\mathcal{L}_{\text{rollout}} = -\left(\sum_{h=1}^{H} \gamma^{h-1}\ \hat{r}^{\text{sc}}_{t+h}\ +\ \gamma^H\ \mathrm{sg}\big(Q_{\psi_1}(\hat{z}^{sa}_{t+H})\big)\right). \tag{12}$$

The world model handles near-horizon prediction where dynamics are locally reliable; the terminal Q-value provides stable estimation beyond the rollout horizon.

**Combined actor objective.** The actor minimizes:

$$\mathcal{L}_{\text{actor}} = \mathcal{L}_{\text{Q-actor}} + \lambda_{\text{roll}}\,\mathcal{L}_{\text{rollout}} + \lambda_{\text{reg}}\|\mathbf{a}_{\text{pre}}\|_2^2, \tag{13}$$

where $\lambda_{\text{roll}}$ controls the rollout guidance weight, $\mathbf{a}_{\text{pre}}$ denotes the actor's pre-tanh activation, and the regularization term prevents tanh saturation. The Q-gradient provides stable local direction from real transitions; the rollout loss provides multi-step awareness that remains informative even when the Q-landscape flattens around a converged policy. All target networks (encoder, actor, critic) are updated via periodic hard copy every $K$ steps, synchronized with the encoder's training phase. Complete pseudocode, network architectures, and hyperparameters are provided in Appendix A.

# 4 Experiments

## 4.1 Experimental Setup

**Environments.** We evaluate on 28 high-dimensional continuous control tasks spanning three benchmark suites: DMC Hard (Tassa et al., 2018) (12–38 action dimensions), MyoSuite (Caggiano et al., 2022) (39-dimensional muscle actuators), and HumanoidBench (Sferrazza et al., 2024) (19 action dimensions on the H1 humanoid robot). These suites cover quadruped and humanoid locomotion, dexterous musculoskeletal manipulation, and whole-body humanoid coordination across diverse morphologies. Full environment specifications are provided in Table A.2 (Appendix A).

**Baselines.** We compare against five representative methods: SAC (Haarnoja et al., 2018), BRO (Nauman et al., 2024), XQC (Palenicek et al., 2026), SimbaV2 (Lee et al., 2025), and TD-MPC2 (Hansen et al., 2024), spanning model-free and model-based families with varying architectural innovations. Appendix A.2 additionally reports Dreamer V3 (Hafner et al., 2025) and MR.Q (Fujimoto et al., 2025) for completeness. All baseline results use published numbers where available and are evaluated at 1M environment steps. Full baseline descriptions and data sources are provided in Appendix A.2.

**Protocol.** DRIVE uses a consistent set of core hyperparameters across experiments, including batch size 256, encoder horizon $H = 5$, soft-decay coefficient $\alpha = 0.5$, and periodic hard target updates every $K = 250$ steps. All experiments use a single environment with action repeat 2, yielding $5 \times 10^5$ decision steps ($10^6$ environment steps total). Evaluation uses a deterministic policy every 50K steps over 10 episodes. We report mean $\pm$ standard deviation over 3 seeds. Complete hyperparameters are provided in Appendix A.

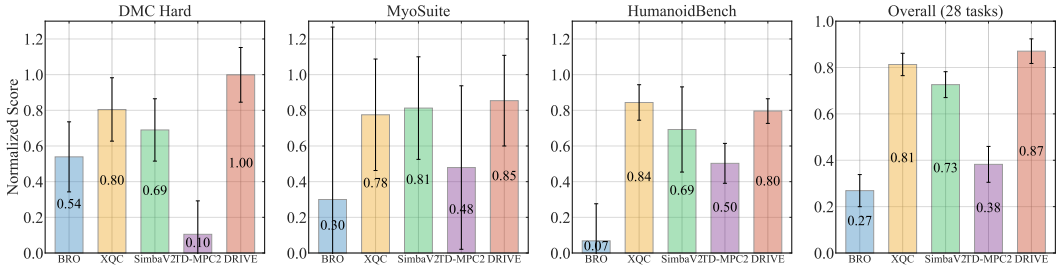



Figure 2: **Normalized performance across three benchmark suites (1M steps).** Scores are min-max normalized per task across all compared methods.

Table 1: **Benchmark results across 28 continuous tasks (1M).** DMC Hard and HumanoidBench (H1) report average return; MyoSuite reports success rate. Dominant win Catastrophic failure

| Suite | Task | SAC | BRO | XQC | SimbaV2 | TD-MPC2 | DRIVE |
|---|---|---|---|---|---|---|---|
| *DeepMind Control Suite — Hard (Average Return ↑)* | | | | | | | |
| DMC | Quadruped Run | 162.0 ± 129.4 | 856.1 ± 38.1 | 940.7 ± 9.4 | 933.9 ± 11.9 | 935.7 ± 10.7 | 951.4 ± 6.1 |
| DMC | Humanoid Stand | 339.3 ± 190.1 | 914.3 ± 14.8 | 904.8 ± 62.5 | 911.8 ± 46.7 | 638.0 ± 21.8 | 925.9 ± 12.6 |
| DMC | Humanoid Walk | 266.8 ± 217.1 | 655.0 ± 69.8 | 698.8 ± 111.9 | 642.4 ± 89.9 | 610.6 ± 13.5 | 837.1 ± 44.9 |
| DMC | Humanoid Run | 106.6 ± 41.6 | 199.5 ± 31.1 | 212.7 ± 30.0 | 190.4 ± 17.6 | 181.7 ± 14.4 | 252.0 ± 48.9 |
| DMC | Dog Stand | 97.7 ± 85.0 | 967.2 ± 13.5 | 982.8 ± 5.2 | 977.8 ± 6.7 | 779.4 ± 35.3 | 983.8 ± 3.3 |
| DMC | Dog Walk | 35.4 ± 37.9 | 926.1 ± 9.7 | 941.5 ± 14.0 | 931.4 ± 13.0 | 493.4 ± 300.2 | 940.5 ± 2.9 |
| DMC | Dog Trot | 29.5 ± 34.2 | 763.4 ± 88.7 | 908.7 ± 17.4 | 857.7 ± 132.2 | 465.8 ± 20.3 | 915.2 ± 17.4 |
| DMC | Dog Run | 38.8 ± 49.1 | 368.5 ± 48.6 | 635.4 ± 52.7 | 550.0 ± 63.6 | 160.2 ± 93.6 | 750.8 ± 22.8 |
| *MyoSuite — Musculoskeletal Control (Success Rate ↑)* | | | | | | | |
| Myo | Pen Twirl Hard | 0.1 ± 0.2 | 0.4 ± 0.3 | 0.8 ± 0.1 | 0.6 ± 0.6 | 0.4 ± 0.0 | 0.9 ± 0.1 |
| Myo | Pen Twirl | 0.5 ± 0.5 | 0.9 ± 0.3 | 1.0 ± 0.0 | 1.0 ± 0.0 | 0.9 ± 0.3 | 1.0 ± 0.0 |
| Myo | Key Turn Hard | 0.1 ± 0.2 | 0.4 ± 0.3 | 0.9 ± 0.3 | 0.6 ± 0.6 | 0.0 ± 0.0 | 1.0 ± 0.0 |
| Myo | Key Turn | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 |
| Myo | Obj Hold Hard | 0.9 ± 0.0 | 0.4 ± 0.2 | 1.0 ± 0.1 | 1.0 ± 0.0 | 0.6 ± 0.1 | 0.9 ± 0.1 |
| Myo | Obj Hold | 0.9 ± 0.2 | 0.9 ± 0.2 | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 |
| Myo | Pose | 0.9 ± 0.2 | 1.0 ± 0.0 | 0.7 ± 0.4 | 1.0 ± 0.0 | 1.0 ± 0.0 | 0.7 ± 0.5 |
| Myo | Reach | 0.2 ± 0.1 | 0.8 ± 0.3 | 0.9 ± 0.1 | 0.9 ± 0.1 | 0.9 ± 0.1 | 1.0 ± 0.0 |
| *HumanoidBench — H1 Robot (Average Return ↑)* | | | | | | | |
| H1 | Stand | 417.7 ± 252.6 | 606.1 ± 71.8 | 838.7 ± 4.2 | 801.6 ± 70.4 | 792.0 ± 12.9 | 883.0 ± 34.1 |
| H1 | Walk | 56.3 ± 25.5 | 229.1 ± 103.9 | 844.5 ± 7.8 | 843.4 ± 6.2 | 810.6 ± 3.0 | 879.7 ± 5.3 |
| H1 | Run | 47.1 ± 44.6 | 57.2 ± 21.8 | 365.4 ± 206.4 | 390.7 ± 155.6 | 770.8 ± 9.1 | 841.2 ± 6.3 |
| H1 | Sit Simple | 686.9 ± 58.1 | 811.9 ± 61.1 | 875.5 ± 5.9 | 873.5 ± 7.1 | 792.5 ± 8.8 | 952.6 ± 0.8 |
| H1 | Sit Hard | 268.0 ± 27.1 | 751.8 ± 83.0 | 827.8 ± 23.6 | 668.7 ± 199.5 | 729.2 ± 32.6 | 859.8 ± 49.6 |
| H1 | Pole | 346.8 ± 85.5 | 427.5 ± 207.3 | 800.5 ± 7.3 | 789.3 ± 9.2 | 656.3 ± 112.1 | 805.7 ± 27.3 |
| H1 | Slide | 66.5 ± 12.6 | 262.8 ± 19.2 | 557.3 ± 31.1 | 463.1 ± 117.5 | 320.8 ± 25.0 | 453.3 ± 22.3 |
| H1 | Stair | 39.7 ± 5.8 | 91.8 ± 26.0 | 528.7 ± 54.4 | 486.3 ± 47.7 | 306.4 ± 122.6 | 183.4 ± 20.4 |
| H1 | Balance Simple | 62.9 ± 12.7 | 69.4 ± 9.5 | 729.4 ± 75.1 | 704.6 ± 106.9 | 37.2 ± 13.6 | 734.8 ± 5.7 |
| H1 | Balance Hard | 50.6 ± 2.1 | 59.7 ± 13.4 | 137.6 ± 15.7 | 138.1 ± 20.6 | 38.0 ± 11.1 | 87.7 ± 9.2 |
| H1 | Crawl | 562.6 ± 20.0 | 882.4 ± 31.2 | 960.4 ± 15.3 | 943.2 ± 23.2 | 961.4 ± 3.7 | 965.8 ± 7.0 |
| H1 | Hurdle | 10.1 ± 8.0 | 49.8 ± 11.0 | 251.9 ± 13.4 | 197.5 ± 42.8 | 342.8 ± 63.7 | 107.3 ± 13.7 |

### 4.2 Main Results

Table 1 and Figure 2 summarize performance across all 28 tasks. Across all three suites, DRIVE achieves the highest overall normalized score (0.87), outperforming XQC (0.81), SimbaV2 (0.73), TD-MPC2 (0.38), and BRO (0.27).

**DMC Hard.** On the Humanoid tasks (24-dim action), DRIVE achieves the highest returns across all four tasks, with particularly large margins on Humanoid Walk (837.1 vs. 698.8 for the next best) and Humanoid Run (252.0 vs. 212.7). On the Dog tasks (38-dim action), where SAC collapses entirely and TD-MPC2 shows high variance, DRIVE achieves consistent performance across all four gaits, reaching 750.8 on Dog Run and surpassing the next best method (XQC, 635.4) by 18%. Standard deviations remain low across seeds, indicating reliable convergence rather than occasional favorable runs. DRIVE attains the highest suite-level normalized score on DMC Hard (1.13 vs. 1.05 for XQC), confirming that structured encoder training provides a clear advantage on the most demanding locomotion tasks. Full learning curves are provided in Figure H.1 (Appendix H).

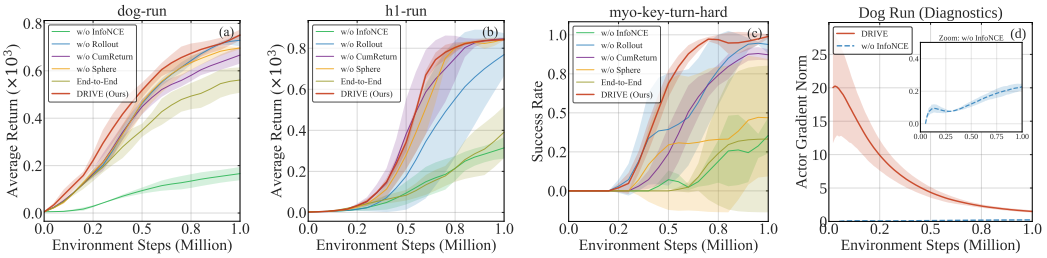



Figure 3: **Ablation study (1M steps).** Learning curves for component removal on (a) Dog Run, (b) H1-Run, (c) Myo-Key-Turn-Hard, and (d) actor gradient norm diagnostics comparing DRIVE vs. w/o InfoNCE on Dog Run. Numerical results in Table 2; full diagnostics in Appendix G.

**MyoSuite.** On musculoskeletal control tasks with 39-dimensional muscle actuators, DRIVE achieves perfect or near-perfect success rates on all tasks, matching or exceeding XQC and SimbaV2. On the challenging Key Turn Hard task, DRIVE achieves $100\%$ success rate where TD-MPC2 fails completely ($0\%$). DRIVE leads the suite-level normalized score ($1.12$ vs. $1.08$ for XQC), demonstrating that the encoder generalizes beyond rigid-body locomotion to biologically realistic muscle dynamics. Full learning curves are provided in Figure I.1 (Appendix I).

**HumanoidBench.** On the H1 humanoid (19-dim action, 12 tasks), DRIVE achieves the highest return on 8 out of 12 tasks, substantially outperforming all baselines on core locomotion. On H1 Run, DRIVE ($841.2$) surpasses the next best method (TD-MPC2, $770.8$) by $9\%$. XQC leads the suite-level normalized score ($1.16$ vs. $1.06$ for DRIVE), primarily due to its advantage on partially observable tasks (Stair, Hurdle), where end-to-end Q-gradients provide implicit temporal reasoning that DRIVE's gradient-isolated encoder cannot access. See Section 4.3 for further discussion. Full learning curves are provided in Figure J.1 (Appendix J).

**Cross-suite patterns.** DRIVE exhibits the lowest catastrophic failure rate: only 2 of 28 tasks fall substantially below the cross-method median, compared to 5 for SimbaV2 and 6 for TD-MPC2. This robustness stems from gradient isolation preventing Q-gradient corruption of the encoder. DRIVE's advantage is most pronounced on the highest-dimensional tasks (Dog Run, MyoSuite, H1 Run), consistent with the hypothesis that structured representation learning becomes increasingly important as scalar reward supervision density decreases relative to action dimensionality. This trend aligns with the representation diagnostics in Appendix G, which show that DRIVE preserves local state separability, maintaining Q-value contrast and non-vanishing actor gradients.

### 4.3 Ablation Study

We ablate five design choices, each removing one component from the full system (Table 2, Figure 3a–c).

Table 2: **Component ablation (1M steps).** Each row removes one component from the full system.

| Variant | Dog Run | H1-Run | Key-H |
|---|---|---|---|
| Full DRIVE | $750.8 \pm 22.8$ | $841.2 \pm 6.3$ | $1.0 \pm 0.0$ |
| End-to-End | $561.2 \pm 53.6$ | $391.1 \pm 126.3$ | $0.3 \pm 0.5$ |
| w/o InfoNCE | $166.3 \pm 28.1$ | $314.5 \pm 52.4$ | $0.4 \pm 0.1$ |
| w/o Sphere | $695.8 \pm 4.7$ | $845.1 \pm 6.8$ | $0.5 \pm 0.4$ |
| w/o Rollout | $728.9 \pm 25.7$ | $769.0 \pm 104.4$ | $0.9 \pm 0.1$ |
| w/o CumR | $666.4 \pm 36.1$ | $846.3 \pm 5.1$ | $0.9 \pm 0.0$ |

**Gradient isolation is essential.** End-to-End training, which allows actor and critic gradients to flow into the encoder, causes catastrophic degradation across all three tasks (Dog Run: $-25\%$, H1-Run: $-54\%$, Key-Turn-Hard: $1.0 \rightarrow 0.3$), with variance increasing by up to $20\times$. This confirms the central hypothesis: when value-fitting gradients reach the encoder, the representational geometry is destabilized by noisy Bellman targets.

**InfoNCE is the critical foundation.** Removing InfoNCE causes catastrophic failure across all three tasks (Dog Run: $750.8$ vs. $166.3$; H1-Run: $841.2$ vs. $314.5$; Key-Turn-Hard: $1.0$ vs. $0.4$). Figure 3(d) reveals the mechanism: without contrastive repulsion, the encoder exhibits local representation aliasing, reducing the actor gradient norm by $100\times$. This interpretation is further supported by

Appendix G, which shows that the failure is not due to global dimensional collapse, but to local aliasing that suppresses Q-value contrast and leads to actor gradient starvation.

**Spherical normalization stabilizes high-dimensional tasks.** Replacing spherical projection with unconstrained Euclidean representations leaves H1-Run unaffected ($845.1$ vs. $841.2$) but degrades Dog Run ($-7\%$) and severely destabilizes Key-Turn-Hard ($1.0 \rightarrow 0.5 \pm 0.4$). The benefit is not uniform performance gain but *metric consistency*: all encoder losses share the same cosine metric, eliminating scale conflicts between supervision signals.

**Cumulative return prediction provides value grounding.** Removing the cumulative return head degrades Dog Run ($-11\%$) while leaving H1-Run unchanged ($846.3$ vs. $841.2$). Under gradient isolation, the cumulative return head is the encoder's only direct access to value information—essential when dynamics prediction alone does not separate value-relevant states.

**Soft-decay rollout provides trajectory-level guidance.** Removing rollout produces modest degradation on Dog Run ($-3\%$) and more pronounced effects on H1-Run ($-9\%$). Single-step Q-gradients suffice in dense reward settings; sparser reward structures may benefit from the rollout.

**Limitation.** DRIVE underperforms on `h1-stair` and `h1-hurdle`, both of which require reacting to terrain geometry (stair edges, hurdle positions) absent from the 51-dimensional proprioceptive observation. These tasks are effectively partially observable: physically distinct situations, such as standing on flat ground versus at a stair edge, map to identical observations, making them unresolvable by any single-frame encoder. On `h1-hurdle`, the agent maintains forward velocity but adopts conservative collision-absorbing poses rather than jumping over unseen obstacles. Both failures share a common root cause: DRIVE's encoder can only extract structure present in the observation. Methods that succeed here rely on temporal context (e.g., frame stacking or recurrence) or decision-time search to implicitly infer unobserved structure, which we leave for future work.

## 5 Conclusion

We introduced **DRIVE**, a model-based RL framework built on strict gradient isolation: the encoder is shaped exclusively by structured supervision on the unit hypersphere, while the critic and actor consume its representations without distorting them. Across 28 high-dimensional continuous control tasks, DRIVE matches or exceeds state-of-the-art model-based and model-free methods without inference-time search, suggesting that representation quality, rather than model accuracy or planning complexity, is the primary bottleneck in high-dimensional MBRL.

Ablation studies confirm that gradient isolation and contrastive supervision are universally critical, while spherical normalization, cumulative return prediction, and soft-decay rollout provide complementary, task-dependent gains. Our diagnostics reveal that the critical failure mode in high-dimensional MBRL is not global dimensional collapse but *local representation aliasing*—a pathology that standard global metrics fail to detect, yet entirely silences actor gradients.

**Limitations.** DRIVE operates on proprioceptive state inputs; extending to visual observations, where the encoder must additionally handle perceptual grounding, remains untested. The soft-decay coefficient $\alpha$ and encoder loss weights are fixed throughout training and across all tasks.

**Future directions.** Visual control may be where DRIVE's gradient isolation principle has its strongest implications: reconstruction-based encoders can waste capacity on task-irrelevant pixels—precisely the failure mode DRIVE is designed to avoid. Benchmarks with rich visual distractors (Yuan et al., 2023) provide a direct testbed. An adaptive $\alpha$ schedule that increasingly trusts the model could enable deeper planning horizons without the instability of full backpropagation, while task-adaptive loss balancing remains an open direction. More broadly, DRIVE suggests that the central question in model-based control is not *how accurately to model* or *how deeply to search*, but *how information flows between learned components*—a perspective that extends beyond the present instantiation.

### AI Disclosure

Generative AI tools were used solely for language editing and readability. They were not used for research ideation, mathematical derivations, proofs, experimental design. All technical content was independently verified by the authors, who take full responsibility for the final manuscript.

## REFERENCES


Amira Abbas, David Sutter, Alessio Figalli, and Stefan Woerner. Effective dimension of machine learning models, 2021. URL https://arxiv.org/abs/2112.04807.

Brandon Amos, Samuel Stanton, Denis Yarats, and Andrew Gordon Wilson. On the model-based stochastic value gradient for continuous reinforcement learning, 2021. URL https://arxiv.org/abs/2008.12775.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning, 2022. URL https://arxiv.org/abs/2105.04906.

Marc G. Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning, 2017. URL https://arxiv.org/abs/1707.06887.

Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and Jan Peters. Crossq: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=PczQtTsTIX.

Vittorio Caggiano, Huawei Wang, Guillaume Durandau, Massimo Sartori, and Vikash Kumar. Myosuite – a contact-rich simulation suite for musculoskeletal motor control, 2022. URL https://arxiv.org/abs/2205.13600.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 1597–1607. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chen20j.html.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models, 2018. URL https://arxiv.org/abs/1805.12114.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 1587–1596, 2018.

Scott Fujimoto, Pierluca D'Oro, Amy Zhang, Yuandong Tian, and Michael Rabbat. Towards general-purpose model-free reinforcement learning. *arXiv preprint arXiv:2501.16142*, 2025.

Peiran Gao, Eric Trautmann, Byron Yu, Gopal Santhanam, Stephen Ryu, Krishna Shenoy, and Surya Ganguli. A theory of multineuronal dimensionality, dynamics and measurement. *BioRxiv*, pp. 214262, 2017.

Christopher Grimm, André Barreto, Satinder Singh, and David Silver. The value equivalence principle for model-based reinforcement learning, 2020. URL https://arxiv.org/abs/2011.03506.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, pp. 1861–1870, 2018.

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

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

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2024.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G.M. Araújo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022.

Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics: Methodology and distribution*, pp. 492–518. Springer, 1992.

Michael Laskin, Aravind Srinivas, and Pieter Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5639–5650. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/laskin20a.html.

Hojoon Lee, Youngdo Lee, Takuma Seno, Donghu Kim, Peter Stone, and Jaegul Choo. Hyperspherical normalization for scalable deep reinforcement learning. *arXiv preprint arXiv:2502.15280*, 2025.

Haotian Lin, Pengcheng Wang, Jeff Schneider, and Guanya Shi. Td-m(pc)$^2$: Improving temporal difference mpc through policy constraint, 2025. URL https://arxiv.org/abs/2502.03550.

Zhihao Lin. Beyond distributions: Geometric action control for continuous reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. URL https://arxiv.org/abs/2603.19312.

Heejeong Nam, Quentin Le Lidec, Lucas Maes, Yann LeCun, and Randall Balestriero. Causal-jepa: Learning world models through object-level latent interventions, 2026. URL https://arxiv.org/abs/2602.11389.

Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample efficient continuous control. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Daniel Palenicek, Florian Vogt, Joe Watson, Ingmar Posner, and Jan Peters. XQC: Well-conditioned optimization accelerates deep reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tx1ZvypKqS.

Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations, 2021. URL https://arxiv.org/abs/2007.05929.

Carmelo Sferrazza, Dun-Ming Huang, Xingyu Lin, Youngwoon Lee, and Pieter Abbeel. Humanoidbench: Simulated humanoid benchmark for whole-body locomotion and manipulation. *arXiv preprint arXiv:2403.10506*, 2024.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite, 2018. arXiv:1801.00690.

Shengjie Wang, Shaohuai Liu, Weirui Ye, Jiacheng You, and Yang Gao. Efficientzero v2: Mastering discrete and continuous control with limited data, 2024. URL https://arxiv.org/abs/2403.00564.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere, 2022. URL https://arxiv.org/abs/2005.10242.

Yuhang Wang, Hanwei Guo, Sizhe Wang, Long Qian, and Xuguang Lan. Bootstrapped model predictive control. *arXiv preprint arXiv:2503.18871*, 2025.

Weirui Ye, Shaohuai Liu, Thanard Kurutach, Pieter Abbeel, and Yang Gao. Mastering atari games with limited data. In *NeurIPS*, 2021.

Zhecheng Yuan, Sizhe Yang, Pu Hua, Can Chang, Kaizhe Hu, and Huazhe Xu. Rl-vigen: a reinforcement learning benchmark for visual generalization. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

Guojian Zhan, Likun Wang, Xiangteng Zhang, Jiaxin Gao, Masayoshi Tomizuka, and Shengbo Eben Li. Bootstrap off-policy with world model. *arXiv preprint arXiv:2511.00423*, 2025.

Yi Zhao, Wenshuai Zhao, Rinu Boney, Juho Kannala, and Joni Pajarinen. Simplified temporal consistency reinforcement learning. In *International Conference on Machine Learning*, pp. 42227–42246. PMLR, 2023.

# A ALGORITHM AND HYPERPARAMETERS

The implementation builds on CleanRL (Huang et al., 2022), extending TD3 (Fujimoto et al., 2018) with the decoupled encoder, structured multi-loss training, and soft-decay rollout in Section 3.

Table A.1: DRIVE hyperparameters.

| Category | Hyperparameter | Value |
|---|---|---|
| *General* | | |
| | Discount factor $\gamma$ | 0.99 |
| | Replay buffer size | $10^6$ |
| | Batch size $B$ | 256 |
| | Learning starts | 25,000 |
| | Action repeat | 2 |
| | Training steps (decision steps) | $5 \times 10^5$ |
| | Total environment steps | $10^6$ |
| *Encoder* | | |
| | Latent state dim $\lvert z^s \rvert$ | 512 |
| | Latent action dim $\lvert z^a \rvert$ | 256 |
| | Joint feature dim $\lvert z^{sa} \rvert$ | 512 |
| | Hidden dim | 512 |
| | Learning rate | $10^{-4}$ |
| | Weight decay | $10^{-4}$ |
| | Optimizer | AdamW |
| | Activation | ELU |
| | Normalization | LayerNorm (per layer) |
| | Horizon $H$ (encoder & rollout) | 5 |
| | Hard update frequency $K$ | 250 |
| | Dynamics loss weight $\lambda_{\mathrm{dyn}}$ | 1.0 |
| | Reward loss weight $\lambda_{\mathrm{rew}}$ | 0.1 |
| | Done loss weight $\lambda_{\mathrm{done}}$ | 0.1 |
| | InfoNCE loss weight $\lambda_{\mathrm{NCE}}$ | 0.3 |
| | InfoNCE temperature $\tau$ | 0.1 |
| | Cumulative return weight $\lambda_{\mathrm{cum}}$ | 0.3 |
| *Critic (Q-Network)* | | |
| | Hidden dim | 512 |
| | Learning rate | $3 \times 10^{-4}$ |
| | Optimizer | Adam |
| | Activation | SiLU |
| | Normalization | LayerNorm |
| | N-step return $n$ | 3 |
| | Gradient clip norm | 20.0 |
| | Number of critics | 2 (min ensemble) |
| *Actor* | | |
| | Hidden dim | 512 |
| | Learning rate | $3 \times 10^{-4}$ |
| | Optimizer | Adam |
| | Activation | SiLU |
| | Normalization | LayerNorm |
| | Output activation | $\tanh$ |
| | Policy delay $d_{\mathrm{policy}}$ | 2 |
| | Exploration noise $\sigma_{\mathrm{explore}}$ | 0.1 |
| | Target policy noise $\sigma_{\mathrm{target}}$ | 0.2 |
| | Noise clip $c$ | 0.5 |
| *Soft-Decay Rollout* | | |
| | Soft-decay coefficient $\alpha$ | 0.5 |
| | Rollout loss weight $\lambda_{\mathrm{roll}}$ | 0.3 |
| | Terminal value | $Q_{\psi_1}$ (stop-gradient) |
| *Target Networks* | | |
| | Target update (Encoder / Actor / Critic) | Hard copy every $K = 250$ steps |

### A.1 Reference Implementation

**Decoupled Encoder: State-Action Separation.**

```
class Encoder(nn.Module):
   def encode_state(self, state):
      """State -> zs on unit hypersphere"""
      zs = ln_activ(self.zs_mlp(state), F.elu)
      return F.normalize(zs, dim=-1)

   def forward(self, zs, action):
      """(zs, action) -> zsa (joint feature for Q)"""
      za = F.elu(self.za_proj(action))
      return self.zsa_mlp(torch.cat([zs, za], dim=-1))

   def predict_world(self, zs, action):
      """(zs, a) -> (done, next_zs, rew, cum_rew, rew_scalar)"""
      zsa = self.forward(zs, action)
      out = self.model_head(zsa) # single linear layer
      cum_r = self.cum_reward_head(zsa)
      r_sc = self.reward_scalar_head(zsa)
      return pred_done, pred_next_zs, pred_rew, cum_r, r_sc
```

**Gradient Isolation: Three-Channel Decoupling.**

```
# Critic training: encoder gradients fully stopped
with torch.no_grad():
   zs = encoder.encode_state(s_obs)
   zsa = encoder(zs, s_actions) # sg(encoder)
qf1_loss = F.smooth_l1_loss(qf1(zsa), target, beta=0.3)

# Actor training: encoder gradients fully stopped
with torch.no_grad():
   zs_for_actor = encoder.encode_state(s_obs) # sg(encoder)
policy_actions = actor(zs_for_actor)
# Actor receives gradient from Q and rollout, never encoder
```

**Structured Encoder Training: Five-Loss Objective.**

```
# Hard update targets, then train encoder K times
encoder_target.load_state_dict(encoder.state_dict())

for _ in range(K): # K = 250 concentrated updates
   states, actions, rewards, dones = rb.sample_trajectories(B, H)
   # Target encoder encodes all next states (no gradient)
   with torch.no_grad():
      target_zs_all = encoder_target.encode_state(states[:, 1:])
      cum_targets = compute_cumulative_returns(rewards, dones)

   pred_zs = encoder.encode_state(states[:, 0])
   for h in range(H): # H-step autoregressive rollout
      pred_d, pred_next_zs, pred_r, pred_cum_r, _ = \
         encoder.predict_world(pred_zs, actions[:, h])
      pred_next_zs = F.normalize(pred_next_zs, dim=-1)

      # On unit sphere: ||a-b||^2 = 2(1-cos(a,b)), see Prop. B.1
      L_dyn = masked_mse(pred_next_zs, target_zs_all[:, h], mask)
      L_rew = two_hot.ce_loss(pred_r, rewards[:, h])
      L_done = F.mse_loss(pred_d, dones[:, h])
      L_cum = two_hot.ce_loss(pred_cum_r, symlog(cum_targets[:, h]))

      # InfoNCE: identity discrimination in batch
      logits = pred_next_zs @ \
            F.normalize(target_zs_all[:, h]).T / 0.1
```

```
      L_nce = F.cross_entropy(logits, torch.arange(B))

      pred_zs = pred_next_zs # autoregressive unroll
```

**Soft-Decay Rollout: Learned MPC without Search.**

```
z = zs_for_actor    # initial latent state
rollout_reward = 0
rollout_action = policy_actions
gamma_acc = 1.0

for h in range(H):  # H = 5 imagination steps
   _, next_z, _, _, pred_r = encoder.predict_world(z, rollout_action)
   rollout_reward += gamma_acc * pred_r.squeeze(-1)
   gamma_acc *= gamma

   # Normalize to sphere, then soft detach
   next_z = F.normalize(next_z, dim=-1)
   z = alpha * next_z + (1 - alpha) * next_z.detach()
   rollout_action = actor(z)

# Q as terminal cost (stop gradient)
terminal_value = qf1(encoder(z, rollout_action)).detach()
rollout_value = rollout_reward + gamma_acc * terminal_value

# Actor dual signal: local Q + global rollout
actor_loss = -qf1(zsa_policy).mean() \
        + w_roll * (-rollout_value.mean())
```

**N-step Returns and Huber Loss (Critic).**

```
# N-step replay buffer accumulates:
# R_n = sum(gamma^k * r_{t+k} for k in range(n)), n = 3
y = R_n + (1 - d) * (gamma ** n) * min(Q1_target, Q2_target)
# Huber loss for gain compression
loss = F.smooth_l1_loss(Q_pred, y)
```

### A.2 BASELINES

All baseline results are taken from the original publications.

**SAC (Haarnoja et al., 2018).** Soft Actor-Critic is an off-policy actor-critic algorithm that jointly maximizes expected return and entropy. For DMC Hard, MyoSuite, and HBench tasks, we use the results from Lee et al. (2025), obtained by running the official repository over 10 random seeds with update-to-data (UTD) ratio set to 2.

**Dreamer V3 (Hafner et al., 2025).** Dreamer V3 learns a latent world model through observation reconstruction and optimizes policies via imagined rollouts. For all tasks, we use the results from Lee et al. (2025), obtained by running the official repository over 3 random seeds.

**BRO (Nauman et al., 2024).** BRO scales the SAC critic by integrating distributional Q-learning, optimistic exploration, and periodic resets. For DMC Hard, MyoSuite, and HBench tasks, we use the results from Lee et al. (2025), obtained by running the official repository (`https://github.com/naumix/BiggerRegularizedOptimistic`) over 10 random seeds for DMC Hard and MyoSuite, and 5 random seeds for HBench tasks. Unless stated otherwise, the UTD ratio is set to 2.

**TD-MPC2 (Hansen et al., 2024).** TD-MPC2 is a model-based algorithm that learns a decoder-free world model through joint dynamics, reward, and value prediction, and performs local trajectory optimization via MPPI in the learned latent space. For DMC Hard tasks, we use the results from Fujimoto et al. (2025), averaged over 10 seeds. For MyoSuite and HBench tasks, we use the results

from Lee et al. (2025), obtained by running the official repository (`https://github.com/nicklashansen/tdmpc2`) over 3 random seeds.

**MR.Q (Fujimoto et al., 2025).** MR.Q is a model-free algorithm that uses model-based objectives—dynamics and reward prediction with a decoupled state-action encoder—to learn rich representations for an actor-critic agent. For DMC Hard tasks, we use the results from Fujimoto et al. (2025), averaged over 10 random seeds. MR.Q results are not available for MyoSuite and HBench tasks.

**XQC (Palenicek et al., 2026).** XQC built on SAC that combines batch normalization, weight normalization, and distributional cross-entropy loss to achieve well-conditioned critic optimization. For DMC Hard, MyoSuite, and HBench tasks, we use the results provided by the authors from the official repository (`https://github.com/danielpalenicek/xqc`).

**SimBa-v2 (Lee et al., 2025).** SimBa-v2 projects representations onto the unit hypersphere via weight normalization, stabilizing high update-to-data ratio training. For DMC Hard, MyoSuite, and HBench tasks, we use the results provided by the authors from the official repository (`https://github.com/DAVIAN-Robotics/SimbaV2`).

### A.3 Environment Specifications

Table A.2: Environment specifications for all 28 benchmark tasks. DMC tasks use the DeepMind Control Suite (Tassa et al., 2018); MyoSuite tasks use the MyoSuite musculoskeletal simulator (Caggiano et al., 2022); HumanoidBench tasks use HumanoidBench (Sferrazza et al., 2024) with the H1 robot.

| Suite | Task | Obs dim | Act dim |
|---|---|---|---|
| *DeepMind Control Suite* | | | |
| DMC | Quadruped Run | 78 | 12 |
| DMC | Humanoid Stand | 67 | 24 |
| DMC | Humanoid Walk | 67 | 24 |
| DMC | Humanoid Run | 67 | 24 |
| DMC | Dog Stand | 223 | 38 |
| DMC | Dog Walk | 223 | 38 |
| DMC | Dog Trot | 223 | 38 |
| DMC | Dog Run | 223 | 38 |
| *MyoSuite* | | | |
| MyoSuite | Key Turn | 93 | 39 |
| MyoSuite | Key Turn Hard | 93 | 39 |
| MyoSuite | Obj Hold | 91 | 39 |
| MyoSuite | Obj Hold Hard | 91 | 39 |
| MyoSuite | Pen Twirl | 83 | 39 |
| MyoSuite | Pen Twirl Hard | 83 | 39 |
| MyoSuite | Pose | 108 | 39 |
| MyoSuite | Reach | 115 | 39 |
| *HumanoidBench (H1)* | | | |
| H1 | Balance Hard | 77 | 19 |
| H1 | Balance Simple | 64 | 19 |
| H1 | Crawl | 51 | 19 |
| H1 | Hurdle | 51 | 19 |
| H1 | Pole | 51 | 19 |
| H1 | Run | 51 | 19 |
| H1 | Sit Simple | 51 | 19 |
| H1 | Sit Hard | 51 | 19 |
| H1 | Slide | 64 | 19 |
| H1 | Stair | 51 | 19 |
| H1 | Stand | 51 | 19 |
| H1 | Walk | 51 | 19 |

# B Spherical Geometry and Metric Unification

**Proposition B.1** (MSE on the unit sphere reduces to cosine distance)**.** *For any two vectors $z_a, z_b \in \mathcal{S}^{d-1}$ (i.e., $\|z_a\|_2 = \|z_b\|_2 = 1$), the squared Euclidean distance is equivalent to cosine distance up to a constant factor:*

$$\|z_a - z_b\|_2^2 = 2\big(1 - \cos(z_a, z_b)\big), \tag{14}$$

*where* $\cos(z_a, z_b) = z_a^\top z_b$.

*Proof.* Expanding the squared norm:

$$\begin{aligned}\|z_a - z_b\|_2^2 &= \|z_a\|_2^2 - 2z_a^\top z_b + \|z_b\|_2^2 \\ &= 2(1 - z_a^\top z_b) = 2(1 - \cos(z_a, z_b)).\end{aligned} \tag{15}$$

□

Since both the online encoder ($\hat{z}^s$) and target encoder ($\bar{z}^s$) produce $\ell_2$-normalized outputs on $\mathcal{S}^{d_s-1}$, the dynamics loss, InfoNCE logits, and rollout trajectory all operate in the same angular metric space. We highlight three implications.

**Metric unification.** Minimizing MSE on normalized vectors is equivalent to maximizing cosine similarity, and the InfoNCE logits $\cos(\hat{z}^s, \bar{z}^s)/\tau$ are directly proportional to the negative of the dynamics loss. All encoder supervision signals share a single geometric language based on angular distance on the sphere, which prevents interference between objectives that would arise if different losses operated in incompatible metric spaces.

**Gradient coherence.** The dynamics loss gradient (before spherical projection) $\nabla_{\hat{z}^s}\mathcal{L}_{\text{dyn}} = -\bar{z}^s$ and the InfoNCE positive-pair gradient share the same direction, differing only in magnitude by a factor of $(1 - p_{ii})/\tau$, where $p_{ii}$ is the softmax probability assigned to the correct positive pair. When the positive pair is already well-discriminated ($p_{ii} \to 1$), the InfoNCE attractive component vanishes automatically, leaving only the dynamics attraction. This self-regulating property prevents over-attraction from redundant gradient signals and ensures that dynamics and contrastive objectives never produce conflicting encoder gradients.

**Bounded rollout trajectories.** Because all predicted latent states are normalized to the unit sphere, prediction errors during the soft-decay rollout (Section 3.2) can only accumulate *angularly*, and the maximum possible drift is bounded by $\pi$ (antipodal). This transforms multi-step prediction from an unbounded extrapolation problem in $\mathbb{R}^{d_s}$ into a bounded problem on a compact manifold, stabilizing gradient-based planning.

**Why project rather than penalize.** A soft penalty $\lambda(\|z\| - 1)^2$ is vulnerable to norm-minimization shortcuts: the encoder can reduce prediction losses by shrinking $\|z\|$ toward zero without learning meaningful representations (Wang & Isola, 2022). Hard projection onto $\mathcal{S}^{d_s-1}$ eliminates this degree of freedom, forcing all representational capacity into the directional structure that downstream components consume.

*Remark* B.2 (Cosine vs MSE formulation). Although MSE and cosine distance are equivalent on $\mathcal{S}^{d_s-1}$ (Proposition B.1), we formulate the dynamics loss as cosine similarity for notational consistency with InfoNCE. Both produce identical pre-normalization gradients after projection through the normalization Jacobian $J = \|\tilde{z}^s\|_2^{-1}(I - z^s z^{s\top})$, where $\tilde{z}^s$ is the pre-normalization embedding and $z^s = \tilde{z}^s/\|\tilde{z}^s\|_2$ (cf. Eq. 1). The Jacobian annihilates the radial component, making the two formulations identical in practice.

**Proposition B.3** (InfoNCE on $\mathcal{S}^{d_s-1}$ prevents representation collapse)**.** *Let $\{z_i^s\}_{i=1}^B \subset \mathcal{S}^{d_s-1}$ be $\ell_2$-normalized embeddings produced by the encoder, and let $z_i^{s+}$ denote the positive pair for $z_i^s$ (the target-encoder embedding of the corresponding next state). The InfoNCE loss with temperature $\tau > 0$ is*

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp\big(z_i^{s\top} z_i^{s+}/\tau\big)}{\sum_{j=1}^{B} \exp\big(z_i^{s\top} z_j^{s+}/\tau\big)}. \tag{16}$$

*If all embeddings completely collapse to a single point $z_i^s = c \in \mathcal{S}^{d_s-1}$ for every $i$, then the loss evaluates exactly to the trivial baseline $\mathcal{L}_{\text{InfoNCE}} = \log B$. Consequently, any configuration achieving $\mathcal{L}_{\text{InfoNCE}} < \log B$ necessarily produces non-degenerate representations.*

*Proof.* Under complete collapse $z_i^s = z_j^{s+} = c$ for all $i, j$, every pairwise cosine similarity equals one: $z_i^{s\top} z_j^{s+} = \|c\|_2^2 = 1$. Substituting into the loss:

$$\mathcal{L}_{\text{InfoNCE}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp(1/\tau)}{B\,\exp(1/\tau)} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{1}{B} = \log B. \tag{17}$$

Since $\log B$ represents the loss under complete representational collapse (or a completely uniform uninformative prior), achieving a training loss $\mathcal{L}_{\text{InfoNCE}} < \log B$ requires the numerator to be systematically larger than the average term in the denominator. This can only occur if the encoder learns to align positive pairs ($z_i^{s\top} z_i^{s+}$) more strongly than negative pairs ($z_i^{s\top} z_j^{s+}$), which structurally precludes the possibility of all states collapsing to a single point. □

*Remark* B.4 (Uniformity guarantee). Proposition B.3 establishes a minimal non-degeneracy result. A stronger characterization is provided by Wang & Isola (2022), who show that the optimal solution of InfoNCE on $\mathcal{S}^{d_s-1}$ converges to the *uniform distribution* on the hypersphere as $B \to \infty$. Combined with the alignment objective from the dynamics loss (which pulls positive pairs together), InfoNCE induces representations that are simultaneously *aligned* (temporally coherent states map to nearby points) and *uniform* (the embedding distribution covers the sphere), directly preventing both complete and dimensional collapse.

*Remark* B.5 (Comparison with distributional anti-collapse regularization). This geometric anti-collapse mechanism is complementary to distributional approaches such as SIGReg (Maes et al., 2026), which enforce an isotropic Gaussian prior on latent embeddings via the Cramér–Wold theorem. On the unit hypersphere, the two targets are asymptotically equivalent in high dimensions, as samples from $\mathcal{N}(0, I_{d_s})$ concentrate on a thin shell of radius $\sqrt{d_s}$, so the isotropic Gaussian prior and the uniform spherical prior converge. However, the spherical formulation additionally provides the three structural benefits discussed above (metric unification, gradient coherence, and bounded rollout trajectories), making it particularly well-suited for model-based RL where the encoder must simultaneously serve dynamics prediction, contrastive learning, and value estimation under non-stationary data.

## C Force Field Analysis: Dynamics and InfoNCE on the Sphere

We derive the gradient structure of the dynamics loss and InfoNCE on $\mathcal{S}^{d_s-1}$, showing that they form complementary attractive and repulsive forces with a self-regulating property.

**Dynamics loss gradient (attraction).** For a predicted-target pair $(\hat{z}^s, \bar{z}^s)$ on the unit sphere, the dynamics loss is $\mathcal{L}_{\text{dyn}} = 1 - (\hat{z}^s)^\top \bar{z}^s$. The Euclidean gradient with respect to $\hat{z}^s$ is:

$$\nabla_{\hat{z}^s} \mathcal{L}_{\text{dyn}} = -\bar{z}^s. \tag{18}$$

After projection onto the tangent plane of $\mathcal{S}^{d_s-1}$ at $\hat{z}^s$, the effective Riemannian gradient becomes $-\left(\bar{z}^s - \left((\hat{z}^s)^\top \bar{z}^s\right)\hat{z}^s\right)$. This vector points from $\hat{z}^s$ toward $\bar{z}^s$ along the geodesic with magnitude $\sin\theta$, where $\theta = \arccos\left((\hat{z}^s)^\top \bar{z}^s\right)$ is the geodesic angle between the two embeddings. This is purely attractive and blind to negative samples, and therefore cannot prevent representational collapse.

**InfoNCE gradient (attraction + repulsion).** The InfoNCE gradient for sample $i$ decomposes as:

$$\nabla_{\hat{z}_i^s} \mathcal{L}_{\text{NCE}}^{(i)} = \underbrace{-\frac{1 - p_{ii}}{\tau}\bar{z}_i^s}_{\text{attraction to positive}} + \underbrace{\frac{1}{\tau}\sum_{j \neq i} p_{ij}\bar{z}_j^s}_{\text{repulsion from negatives}}, \tag{19}$$

where $p_{ij} = \text{softmax}\left((\hat{z}_i^s)^\top \bar{z}_j^s / \tau\right)$. The attraction weakens as $p_{ii} \to 1$; the repulsion concentrates on the most confusable negatives via the softmax weighting (i.e., negatives with high cosine similarity to the anchor). Throughout all experiments we fix $\tau = 0.1$, the default value established in SimCLR (Chen et al., 2020) and widely adopted in self-supervised representation learning; we did not tune this hyperparameter.

**Complementarity.** The dynamics loss ensures temporal coherence (adjacent states are predictable) while InfoNCE ensures informational richness (distinct states remain distinguishable). Together they form a complete force field that prevents the trivial collapse solution where all states map to a single point on the sphere.

*Remark* C.1 (Self-regulating attraction). The two attractive components do not redundantly over-attract. The dynamics loss provides constant attraction $-\bar{z}_i^s$ regardless of alignment quality. The InfoNCE attraction is weighted by $(1 - p_{ii})$ and vanishes as $p_{ii} \to 1$:

$$-\frac{1 - p_{ii}}{\tau}\bar{z}_i^s \;\to\; 0 \qquad \text{as} \quad p_{ii} \to 1. \tag{20}$$

In this regime, InfoNCE contributes only repulsion, and the dynamics loss remains the sole attractive force. This self-regulating transition requires no explicit scheduling.

## D Cross-Entropy vs. MSE for Auxiliary Prediction

We briefly justify the choice of TwoHot cross-entropy over Mean Squared Error (MSE) for the reward and cumulative return prediction heads.

For MSE regression on a scalar target $y$, the gradient magnitude with respect to the prediction $\hat{y}$ is precisely $|\hat{y} - y|$. This gradient vanishes linearly as the prediction improves, weakening the supervisory signal to the encoder precisely when representations begin to stabilize.

Conversely, for TwoHot cross-entropy over $N_b$ bins, the gradient with respect to the pre-softmax logits is a vector $(\hat{p} - q) \in \mathbb{R}^{N_b}$, where $\hat{p}$ is the predicted probability distribution and $q = \mathrm{TwoHot}(y)$ is the target distribution. Because cross-entropy operates in probability space, even when the expected value of $\hat{p}$ closely approximates $y$, the network's uncertainty (entropy) ensures that small distributional discrepancies produce non-vanishing gradients. With $N_b = 65$, this formulation expands a shrinking scalar error into a rich, 65-dimensional supervisory signal (see Table D.1).

Table D.1: Gradient magnitude comparison near convergence ($y = 0.50$, expected prediction $\hat{y} \approx 0.48$). The TwoHot CE gradient norm is computed analytically from the logit gradient $\|\hat{p} - q\|_2$ over $N_b = 65$ bins.

| **Loss Formulation** | **Gradient Dimensionality** | **Approximate $\|\nabla\|_2$** |
|---|---|---|
| Standard MSE | 1 | 0.02 |
| TwoHot Cross-Entropy | 65 | 0.15–0.25 |

This order-of-magnitude difference in gradient norm persists throughout training. It ensures that the TwoHot formulation serves as a consistently active auxiliary task, maintaining rich representation-shaping gradients even in late stages of training where MSE supervision would effectively vanish.

## E Soft-Decay Rollout: Gradient Analysis and Comparison

We provide a formal analysis of the soft-decay rollout mechanism, establishing its gradient attenuation properties and comparing it with alternative model-based actor training strategies.

**Proposition E.1** (Exponential gradient decay on latent paths)**.** *Under the soft-decay mechanism (Eq. 11), gradient paths from rollout step $h$ that carry temporal credit through earlier latent transitions accumulate an attenuation factor of $\alpha^{h-1}$ ($h = 1, \ldots, H$), where $\alpha \in [0, 1]$ is the soft-decay coefficient. The direct gradient from the predicted reward $\hat{r}^{\mathrm{sc}}_{t+h}$ through the local action $\pi_\phi(\tilde{z}^s_{t+h})$ is not attenuated by soft-decay.*

*Proof.* At each rollout step, the soft-decay operation decomposes the predicted latent state as:

$$\tilde{z}^s_{t+h} = \alpha \cdot \hat{z}^s_{t+h} + (1 - \alpha) \cdot \mathrm{sg}(\hat{z}^s_{t+h}). \tag{21}$$

The Jacobian of $\tilde{z}^s_{t+h}$ with respect to $\hat{z}^s_{t+h}$ is:

$$\frac{\partial \tilde{z}^s_{t+h}}{\partial \hat{z}^s_{t+h}} = \alpha \cdot I, \tag{22}$$

since the stop-gradient term contributes zero gradient. Let $J_h = \frac{\partial \hat{z}^s_{t+h+1}}{\partial \tilde{z}^s_{t+h}}$ denote the Jacobian of the world model dynamics at step $h$. By the chain rule, the gradient of the rollout objective with respect to the initial action decomposes as a sum of per-step contributions:

$$\frac{\partial \mathcal{L}_{\text{rollout}}}{\partial \phi} = \sum_{h=1}^{H} \gamma^{h-1} \frac{\partial \hat{r}^{\text{sc}}_{t+h}}{\partial \phi}, \tag{23}$$

where the terminal Q-value is detached and contributes no gradient to the actor. Since the actor recomputes actions at each rollout step $h$, the gradient $\partial \hat{r}^{\text{sc}}_{t+h}/\partial \phi$ decomposes into two types of paths: (1) a *direct path* through the local action $\pi_\phi(\tilde{z}^s_{t+h})$, which does not pass through any prior soft-decay operation and is therefore unattenuated; and (2) *indirect paths* that carry credit through earlier latent transitions $\tilde{z}^s_{t+1} \to \cdots \to \tilde{z}^s_{t+h}$, each passing through $h-1$ soft-decay operations and accumulating an attenuation factor of $\alpha^{h-1}$. The soft-decay mechanism thus explicitly attenuates long-range temporal credit assignment while preserving local action gradients at full strength. □

*Remark* E.2 (Jacobian boundedness from spherical constraint). The spherical geometry of the latent space provides a structural argument for why the dynamics Jacobians $J_h$ do not systematically amplify gradients. Since both the input $\tilde{z}^s_{t+h} \in \mathcal{S}^{d_s-1}$ and the output $\hat{z}^s_{t+h+1}$ (after normalization) lie on the unit sphere, the normalization operation projects any radial component of the Jacobian to zero, leaving only the tangential component. This prevents systematic norm amplification across rollout steps, making $\alpha^{h-1}$ the dominant attenuation factor on indirect gradient paths empirically. However, this should be interpreted as designed attenuation rather than a tight theoretical bound: the full gradient magnitude on indirect paths still depends on the learned dynamics, reward-head, and actor Jacobians, while the direct action gradient at each step is unaffected by $\alpha$.

### E.1 Comparison with Alternative Strategies

Figure E.1 compares the effective gradient magnitude across rollout steps for three model-based actor training strategies. Full BPTT (Dreamer) passes gradients through $H$ composed dynamics Jacobians $\prod_h J_h$, which is susceptible to vanishing and exploding gradients. Sampling-based planning (TD-MPC2/MPPI) provides zero gradient to the actor. Soft-decay rollout contributes $\alpha^{h-1}$ at step $h$, naturally aligning gradient magnitude with prediction reliability.

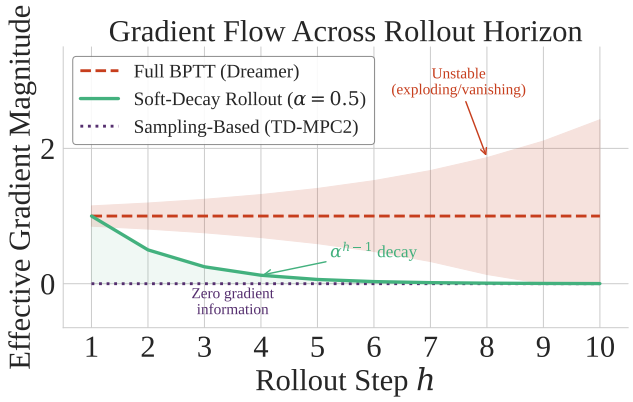



Figure E.1: **Effective gradient magnitude across rollout steps.** Full BPTT maintains unit gradient in theory but is unstable in practice (shaded region). Sampling-based planning provides zero gradient to the actor. Soft-decay rollout ($\alpha = 0.5$) provides geometrically attenuating gradients aligned with prediction reliability.

The coefficient $\alpha$ places DRIVE on a continuous spectrum: $\alpha = 0$ recovers model-free actor-critic (no model gradient), $\alpha = 1$ recovers full BPTT. At $\alpha = 0.5$, step 1 contributes $100\%$ of its gradient to the actor, step 2 contributes $50\%$, step 3 contributes $25\%$, and step 5 contributes $6.25\%$, strongest where the model is locally accurate and naturally suppressed where compounding errors accumulate. Unlike sampling-based planning, this trajectory-level guidance is internalized into the actor during training, yielding zero additional inference cost.

## F Representation Smoothness and Bellman Error Propagation

We formalize the intuition that the encoder acts as a *representation smoother*, showing that when the encoder is Lipschitz continuous, Bellman backup errors are tightly controlled by the encoder's smoothness and the world model's prediction accuracy in latent space.

**Setup.** Let $f_\theta : \mathcal{S} \to \mathcal{Z}$ denote the encoder and $Q_\psi : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}$ the critic. We assume both are Lipschitz continuous:

$$\|f_\theta(s) - f_\theta(s')\|_Z \le L_f \|s - s'\|_S, \qquad |Q_\psi(z,a) - Q_\psi(z',a)| \le L_Q \|z - z'\|_Z. \tag{24}$$

**Lemma F.1** (Value sensitivity bound)**.** *For any two states $s, s'$ and action $a$:*

$$|Q_\psi(f_\theta(s), a) - Q_\psi(f_\theta(s'), a)| \le L_Q \, L_f \, \|s - s'\|_S. \tag{25}$$

*Proof.* By composition of Lipschitz bounds: $|Q_\psi(f_\theta(s), a) - Q_\psi(f_\theta(s'), a)| \le L_Q \|f_\theta(s) - f_\theta(s')\|_Z \le L_Q \, L_f \, \|s - s'\|_S$. □

The product $L_Q L_f$ governs the critic's sensitivity in raw state space. A smoother encoder (smaller $L_f$) yields more stable value estimates under state perturbations.

**Proposition F.2** (Bellman error bound)**.** *Let $\mathcal{T}Q(s,a) = r(s,a) + \gamma \, \mathbb{E}_{s'}[Q_\psi(f_\theta(s'), a')]$ be the Bellman backup using the true next-state encoding, where $z = f_\theta(s)$ and $a' = \pi_{\bar{\phi}}(f_\theta(s'))$ is the action selected by the target policy. Let $\hat{\mathcal{T}}Q(z,a) = \hat{r}(z,a) + \gamma \, Q_\psi(\hat{z}', a')$ be the backup using the learned world model, where $\hat{z}'$ is the latent next-state predicted by the dynamics head. If the reward prediction error is bounded by $\epsilon_r$, then:*

$$|\mathcal{T}Q(s,a) - \hat{\mathcal{T}}Q(z,a)| \le \epsilon_r + \gamma \, L_Q \, \mathbb{E}_{s'}\big[\|f_\theta(s') - \hat{z}'\|_Z\big]. \tag{26}$$

*Proof.* By the triangle inequality:

$$\begin{aligned} |\mathcal{T}Q - \hat{\mathcal{T}}Q| &\le |r(s,a) - \hat{r}(z,a)| + \gamma \, \mathbb{E}_{s'}\big[|Q_\psi(f_\theta(s'), a') - Q_\psi(\hat{z}', a')|\big] \\ &\le \epsilon_r + \gamma \, L_Q \, \mathbb{E}_{s'}\big[\|f_\theta(s') - \hat{z}'\|_Z\big]. \end{aligned}$$

□

**Interpretation.** The bound reveals two channels through which representation quality governs value estimation accuracy:

1. The term $\|f_\theta(s') - \hat{z}'\|_Z$ is precisely the latent dynamics prediction error that DRIVE's $\mathcal{L}_{\text{dyn}}$ minimizes.
2. The coefficient $L_Q$ reflects the critic's sensitivity in latent space, controlled by LayerNorm and Huber loss in the critic architecture.

In DRIVE, $\mathcal{Z} = \mathcal{S}^{d_s - 1}$ and $\|\cdot\|_Z$ reduces to angular distance (Proposition B.1), so the dynamics prediction error $\|f_\theta(s') - \hat{z}'\|_Z$ is precisely the cosine distance minimized by $\mathcal{L}_{\text{dyn}}$. The critic in DRIVE consumes the joint representation $z^{sa} = h(z^s, z^a)$ rather than $(z^s, a)$ directly; the Lipschitz bound applies to the composed mapping $Q_\psi \circ h$.

Crucially, the encoder's smoothness $L_f$ enters *indirectly*: a smooth encoder maps nearby raw states to nearby latent states, making the dynamics prediction task easier and reducing $\|f_\theta(s') - \hat{z}'\|_Z$. A high-curvature encoder amplifies small state differences into large latent distances, increasing the prediction burden.

**Why gradient isolation helps.** In jointly trained architectures, value gradients can increase $L_f$ along directions that reduce immediate TD error but distort global representational geometry—the value loss may increase local curvature, making dynamics prediction harder and ultimately increasing the Bellman error bound. DRIVE eliminates this conflict: the encoder is trained solely to produce smooth, predictable representations via structured world model losses, and the critic adapts to whatever geometry the encoder provides.

This design choice is supported by independent empirical findings in prior work. TCRL (Zhao et al., 2023) shows that jointly training the value function and dynamics model causes the latent space magnitude to grow unboundedly, leading to training instability and performance degradation on challenging locomotion tasks; in contrast, training them separately maintains a stable latent space (see their Figure 5). Similarly, MRQ (Fujimoto et al., 2025) demonstrates that removing the target encoder, thereby allowing value-related gradients to corrupt the representation, causes

catastrophic performance drops across 118 environments spanning four diverse benchmarks (see their Table 2, "No target encoder" variant). These observations are consistent with our bound: uncontrolled value gradients increase the encoder's local curvature $L_f$, amplifying $\|f_\theta(s') - \hat{z}'\|_Z$ and ultimately loosening the Bellman error bound in Proposition F.2.

# G Representation Health Diagnostics

We define three diagnostic metrics to verify that DRIVE's encoder avoids representational collapse while maintaining task-relevant structure. All metrics are computed on mini-batches of encoded states $\{z_i^s\}_{i=1}^B$ from the replay buffer, with $Z \in \mathbb{R}^{B \times d_s}$ denoting the centered representation matrix.

**Off-diagonal cosine similarity.** Following the covariance regularization criterion of VICReg (Bardes et al., 2022), we measure $\bar{S}_{\text{off}} = \frac{1}{B(B-1)} \sum_{i \neq j} (z_i^s)^\top z_j^s$. Values near 0 indicate diverse representations, while values approaching 1 indicate dimensional collapse.

**Effective dimension.** Defined via the Shannon entropy of the normalized singular value spectrum (Abbas et al., 2021): $d_{\text{eff}} = \exp\big(-\sum_k \tilde{\sigma}_k \log \tilde{\sigma}_k\big)$, where $\tilde{\sigma}_k = \sigma_k / \sum_j \sigma_j$ and $\{\sigma_k\}$ are the singular values of $Z$. Higher values indicate better utilization of representational capacity.

**Participation ratio.** Computed from the per-dimension variances $\{v_k\}_{k=1}^{d_s}$ of $Z$ as $\text{PR} = (\sum_k v_k)^2 / \sum_k v_k^2$ (Gao et al., 2017). This moment-based metric ranges from 1 (a single dominant dimension) to $d_s$ (perfectly uniform utilization across all dimensions).

**Downstream functional metrics.** Beyond representation-level statistics, we track three metrics that capture the downstream impact of representation quality on Q-learning and policy optimization: (i) the number of *similar pairs* (state pairs with cosine similarity $> 0.7$), which detects local clustering in latent space; (ii) the mean *Q-value difference* among similar pairs, which measures whether the critic can distinguish states that the encoder maps nearby; and (iii) the *actor gradient norm*, which reflects whether the policy receives informative learning signals from the critic.

## G.1 Empirical Evidence: Dog Run

Figure G.1 compares DRIVE against its ablation without InfoNCE on Dog Run ($\dim(\mathcal{S}) = 223$, $\dim(\mathcal{A}) = 38$), using 3 random seeds. We report both episodic return and all six diagnostic metrics.

Three key observations emerge, revealing a causal chain from representation quality to downstream learning failure.

**Global representation health shows moderate differences.** Panels (b)–(d) show that removing InfoNCE produces only modest changes in global metrics: off-diagonal similarity increases from $\sim$0.10 to $\sim$0.19, effective dimension decreases slightly, and participation ratio drops from $\sim$480 to $\sim$440. These differences, while consistent, might suggest that InfoNCE is merely beneficial rather than essential.

**Local representation aliasing reveals the true failure mode.** However, panel (f) exposes a far more severe pathology: without InfoNCE, the number of similar pairs increases by approximately $10\times$ (from $\sim$10–70 to $\sim$70–700). This indicates that the encoder develops *local clustering*, where many distinct states are mapped to nearly identical latent positions, even though the global average similarity remains moderate. Crucially, panel (g) shows that the Q-value difference among these aliased pairs drops from $\sim$1.5–2.5 to $\sim$0.2–0.4, confirming that the critic loses its ability to distinguish states that the encoder collapses together.

**Actor gradient starvation completes the causal chain.** Panel (h) demonstrates the downstream consequence: the actor gradient norm in the ablated variant is effectively zero ($< 0.23$ throughout training), compared to $\sim$1.5–20 for DRIVE. Without fine-grained Q-value differences to generate informative gradients, the policy receives no learning signal and stagnates. This is consistent with the

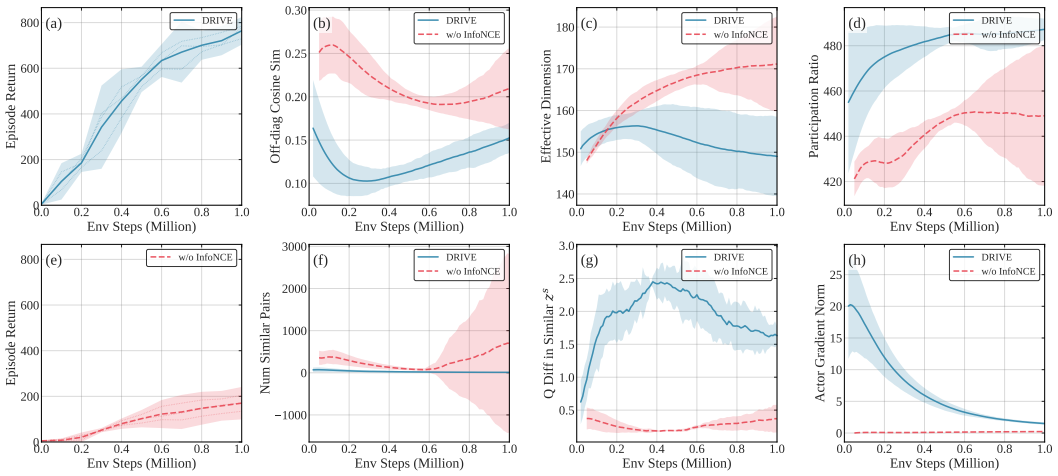



Figure G.1: **Representation diagnostics on Dog Run.** Top row: DRIVE episodic return and representation health metrics (off-diagonal cosine similarity, effective dimension, participation ratio). Bottom row: ablation without InfoNCE episodic return and downstream functional metrics (number of similar pairs, Q-value difference among similar states, actor gradient norm). Solid blue: DRIVE; dashed red: w/o InfoNCE. Shaded regions indicate 95% confidence intervals over 3 seeds.

$4\times$ performance gap observed in panels (a) vs. (e): DRIVE achieves $\sim$764 mean return while the ablation reaches only $\sim$170.

**Summary.** These results establish that InfoNCE's primary contribution is not preventing global dimensional collapse, which dynamics loss alone partially addresses, but rather preventing *local representation aliasing*. The causal chain is: local clustering $\rightarrow$ Q-value flattening $\rightarrow$ actor gradient starvation $\rightarrow$ performance degradation. This finding highlights that batch-internal negative samples in InfoNCE provide essential local repulsive forces that dynamics prediction alone cannot supply.

**Computational cost.** Table G.1 compares computational cost across methods. At inference, TD-MPC2 requires MPPI sampling (64 trajectories $\times$ 5 iterations) at every decision step; DRIVE uses a single forward pass through the actor network, identical to model-free methods. Training time differences are even more pronounced: on Dog Run, TD-MPC2 requires approximately 25 hours while DRIVE completes in approximately 3 hours on the same hardware, an $8\times$ speedup.

Table G.1: **Computational cost comparison** on Dog Run (RTX 3090, 1M steps). DRIVE incurs no planning cost at inference and trains $8\times$ faster than TD-MPC2.

| Method | Inference | Training | Planning |
|---|---|---|---|
| SAC / TD3 | $\sim$0.1 ms/step | $\sim$3 h | None |
| XQC / SimbaV2 | $\sim$0.1 ms/step | $\sim$4 h | None |
| DRIVE (Ours) | $\sim$0.1 ms/step | $\sim$4 h | None (training only) |
| TD-MPC2 | $\sim$5–10 ms/step | $\sim$25 h | MPPI (64$\times$5) |

## H FULL RESULTS: DMC HARD

Table H.1: **Average return on DMC Hard tasks (1M steps).** Mean ± std over 3 seeds. Best in **bold**.

| Task | Dreamer V3 | SAC | TD-MPC2 | BRO | MR.Q | XQC | SimbaV2 | DRIVE (Ours) |
|---|---|---|---|---|---|---|---|---|
| Quadruped-Run | 471.3 ± 229.8 | 162.0 ± 129.4 | 935.7 ± 10.7 | 856.1 ± 38.1 | 947.8 ± 7.7 | 940.7 ± 9.4 | 933.9 ± 11.9 | **951.4 ± 6.1** |
| Humanoid-Stand | 5.1 ± 1.9 | 339.3 ± 190.1 | 638.0 ± 21.8 | 914.3 ± 14.8 | 869.7 ± 44.9 | 904.8 ± 62.5 | 911.8 ± 46.7 | **925.9 ± 12.6** |
| Humanoid-Walk | 1.3 ± 0.3 | 266.8 ± 217.1 | 610.6 ± 13.5 | 655.0 ± 69.8 | 646.6 ± 92.2 | 698.8 ± 111.9 | 642.4 ± 89.9 | **837.1 ± 44.9** |
| Humanoid-Run | 0.9 ± 0.1 | 106.6 ± 41.6 | 181.7 ± 14.4 | 199.5 ± 31.1 | 195.4 ± 51.2 | 212.7 ± 30.0 | 190.4 ± 17.6 | **252.0 ± 48.9** |
| Dog-Stand | 55.9 ± 19.1 | 97.7 ± 85.0 | 779.4 ± 35.3 | 967.2 ± 13.5 | 965.7 ± 8.6 | 982.8 ± 5.2 | 977.8 ± 6.7 | **983.8 ± 3.3** |
| Dog-Walk | 23.4 ± 11.0 | 35.4 ± 37.9 | 493.4 ± 300.2 | 926.1 ± 9.7 | 908.5 ± 17.2 | **941.5 ± 14.0** | 931.4 ± 13.0 | 940.5 ± 2.9 |
| Dog-Trot | 10.2 ± 3.6 | 29.5 ± 34.2 | 465.8 ± 20.3 | 763.4 ± 88.7 | 873.4 ± 47.6 | 908.7 ± 17.4 | 857.7 ± 132.2 | **915.2 ± 17.4** |
| Dog-Run | 15.7 ± 8.8 | 38.8 ± 49.1 | 160.2 ± 93.6 | 368.5 ± 48.6 | 560.7 ± 36.8 | 635.4 ± 52.7 | 550.0 ± 63.6 | **750.8 ± 22.8** |

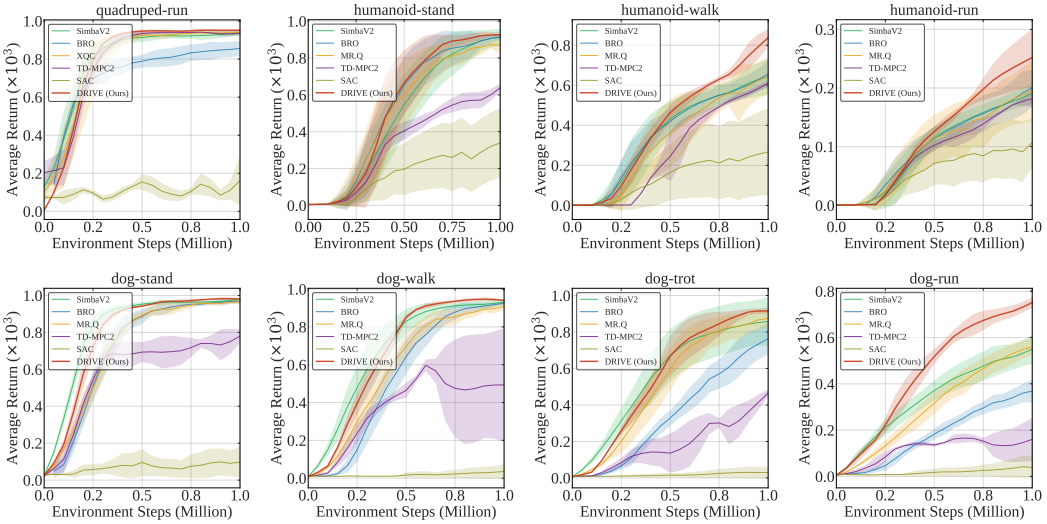



Figure H.1: **Learning curves on DMC Hard tasks (1M environment steps).**

# I FULL RESULTS: MYOSUITE

Table I.1: **Success rate on MyoSuite tasks (1M steps).** Mean ± std over 3 seeds. Best in **bold**.

| Task | Dreamer V3 | SAC | TD-MPC2 | BRO | XQC | SimbaV2 | DRIVE (Ours) |
|---|---|---|---|---|---|---|---|
| myo-pen-twirl-hard | 0.5 ± 0.2 | 0.1 ± 0.2 | 0.4 ± 0.0 | 0.4 ± 0.3 | 0.8 ± 0.1 | 0.6 ± 0.6 | **0.9 ± 0.1** |
| myo-pen-twirl | 1.0 ± 0.1 | 0.5 ± 0.5 | 0.9 ± 0.3 | 0.9 ± 0.3 | 1.0 ± 0.0 | 1.0 ± 0.0 | **1.0 ± 0.0** |
| myo-key-turn-hard | 0.0 ± 0.0 | 0.1 ± 0.2 | 0.0 ± 0.0 | 0.4 ± 0.3 | 0.9 ± 0.3 | 0.6 ± 0.6 | **1.0 ± 0.0** |
| myo-key-turn | 0.9 ± 0.2 | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 | 1.0 ± 0.0 | **1.0 ± 0.0** |
| myo-obj-hold-hard | 0.1 ± 0.0 | 0.9 ± 0.0 | 0.6 ± 0.1 | 0.4 ± 0.2 | 1.0 ± 0.1 | **1.0 ± 0.0** | 0.9 ± 0.1 |
| myo-obj-hold | 0.3 ± 0.6 | 0.9 ± 0.2 | 1.0 ± 0.0 | 0.9 ± 0.2 | 1.0 ± 0.0 | 1.0 ± 0.0 | **1.0 ± 0.0** |
| myo-pose | **1.0 ± 0.0** | 0.9 ± 0.2 | **1.0 ± 0.0** | **1.0 ± 0.0** | 0.7 ± 0.4 | **1.0 ± 0.0** | 0.7 ± 0.5 |
| myo-reach | **1.0 ± 0.0** | 0.2 ± 0.1 | 0.9 ± 0.1 | 0.8 ± 0.3 | 0.9 ± 0.1 | 0.9 ± 0.1 | **1.0 ± 0.0** |

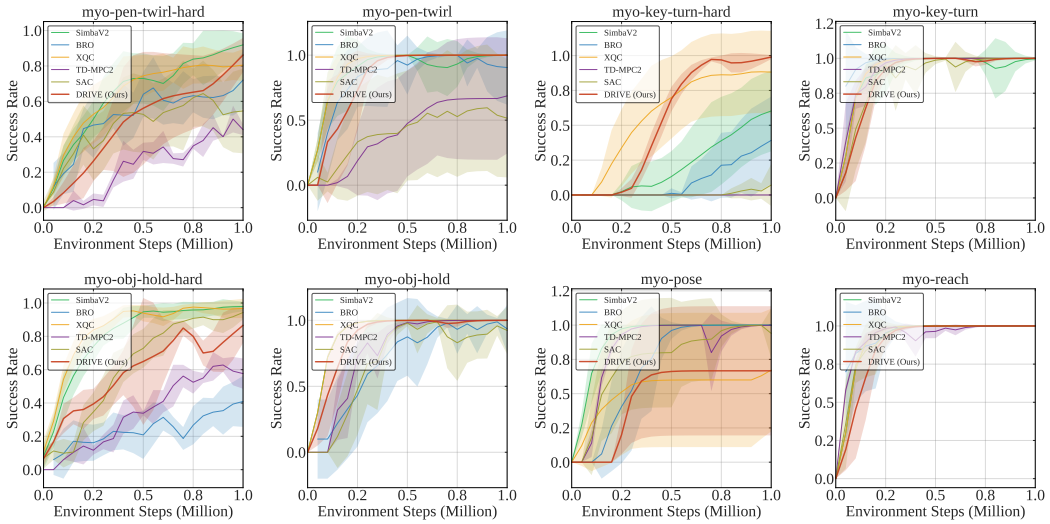



Figure I.1: **Learning curves on MyoSuite tasks (1M environment steps).**

# J FULL RESULTS: HUMANOIDBENCH (H1BENCH)

Table J.1: **Average return on H1Bench tasks (1M steps).** Mean ± std over 3 seeds. Best in **bold**.

| Task | Dreamer V3 | SAC | TD-MPC2 | BRO | XQC | SimbaV2 | DRIVE (Ours) |
|---|---|---|---|---|---|---|---|
| h1-stand | 224.8 ± 189.9 | 417.7 ± 252.6 | 792.0 ± 12.9 | 606.1 ± 71.8 | 838.7 ± 4.2 | 801.6 ± 70.4 | **883.0 ± 34.1** |
| h1-walk | 87.9 ± 67.5 | 56.3 ± 25.5 | 810.6 ± 3.0 | 229.1 ± 103.9 | 844.5 ± 7.8 | 843.4 ± 6.2 | **879.7 ± 5.3** |
| h1-run | 60.2 ± 32.7 | 47.1 ± 44.6 | 770.8 ± 9.1 | 57.2 ± 21.8 | 365.4 ± 206.4 | 390.7 ± 155.6 | **841.2 ± 6.3** |
| h1-sit-simple | 30.4 ± 21.2 | 686.9 ± 58.1 | 792.5 ± 8.8 | 811.9 ± 61.1 | 875.5 ± 5.9 | 873.5 ± 7.1 | **952.6 ± 0.8** |
| h1-sit-hard | 7.5 ± 8.3 | 268.0 ± 27.1 | 729.2 ± 32.6 | 751.8 ± 83.0 | 827.8 ± 23.6 | 668.7 ± 199.5 | **859.8 ± 49.6** |
| h1-pole | 202.4 ± 144.2 | 346.8 ± 85.5 | 656.3 ± 112.1 | 427.5 ± 207.3 | 800.5 ± 7.3 | 789.3 ± 9.2 | **805.7 ± 27.3** |
| h1-slide | 17.1 ± 10.8 | 66.5 ± 12.6 | 320.8 ± 25.0 | 262.8 ± 19.2 | **557.3 ± 31.1** | 463.1 ± 117.5 | 453.3 ± 22.3 |
| h1-stair | 34.3 ± 8.5 | 39.7 ± 5.8 | 306.4 ± 122.6 | 91.8 ± 26.0 | **528.7 ± 54.4** | 486.3 ± 47.7 | 183.4 ± 20.4 |
| h1-balance-simple | 11.3 ± 0.6 | 62.9 ± 12.7 | 37.2 ± 13.6 | 69.4 ± 9.5 | 729.4 ± 75.1 | 704.6 ± 106.9 | **734.8 ± 5.7** |
| h1-balance-hard | 12.9 ± 2.2 | 50.6 ± 2.1 | 38.0 ± 11.1 | 59.7 ± 13.4 | 137.6 ± 15.7 | **138.1 ± 20.6** | 87.7 ± 9.2 |
| h1-crawl | 634.7 ± 119.4 | 562.6 ± 20.0 | 961.4 ± 3.7 | 882.4 ± 31.2 | 960.4 ± 15.3 | 943.2 ± 23.2 | **965.8 ± 7.0** |
| h1-hurdle | 13.1 ± 2.4 | 10.1 ± 8.0 | **342.8 ± 63.7** | 49.8 ± 11.0 | 251.9 ± 13.4 | 197.5 ± 42.8 | 107.3 ± 13.7 |

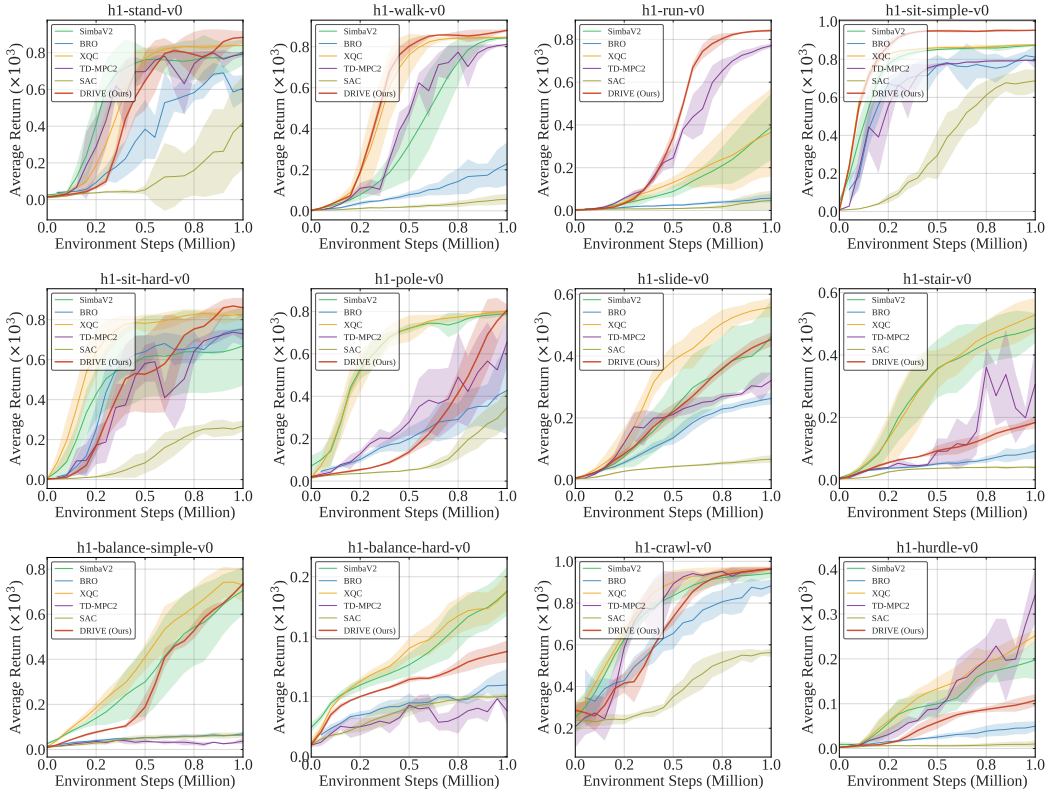



Figure J.1: **Learning curves on H1Bench tasks (1M environment steps).**