THREE GAINS, ONE LOOP: DIAGNOSING AND REPAIRING THE BELLMAN BOOTSTRAP IN HIGH DIMENSIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Model-free reinforcement learning is widely believed to require world models or complex architectural innovations to scale to high-dimensional continuous control. We show that this difficulty can stem from a core structural instability: in high dimensions, the Bellman bootstrap forms a positive-feedback loop that can amplify noise faster than it accumulates signal. As the agent improves, its actions move into poorly covered regions, generating out-of-distribution temporal-difference errors that can corrupt the value landscape needed for continued learning. We identify three gains governing this feedback loop, corresponding to target signal, critic-gradient geometry, and error sensitivity, and show how each can be compressed by an architecture-preserving intervention: multi-step returns, LayerNorm, and Huber loss, respectively. We term this repair Gain-Compressed Repair (GCR). While the individual components are established techniques, the gain decomposition explains their complementary roles and predicts distinct failure signatures when individual gains are left uncontrolled. Together, these modifications add negligible parameter overhead yet enable standard TD3 and SAC to achieve nearly 800 reward on Dog-Run (38-D actions) within 2M steps and strong performance on 61-dimensional H1Hand tasks. Across 12 primary high-dimensional benchmarks spanning Dog, Humanoid, and H1Hand, GCR remains competitive with strong model-based baselines, suggesting that Bellman-loop instability is a key bottleneck for model-free learning in the studied state-based settings.

1 INTRODUCTION

Reinforcement learning (RL) has achieved remarkable success in domains ranging from game playing to robotic manipulation. Yet a striking gap persists: in continuous control tasks with high-dimensional action spaces, such as quadruped and humanoid locomotion with 21 to 61 actuated degrees of freedom, model-free methods (e.g., TD3 (Fujimoto et al., 2018), SAC (Haarnoja et al., 2018)) often struggle, while recent model-based approaches (e.g., Dreamer (Hafner et al., 2025), TD-MPC2 (Hansen et al., 2024)) achieve substantially stronger performance. This empirical pattern has cemented a widely held belief: *Scaling continuous control requires world models*.

We show that this belief mistakes a symptom for a cause. Model-free RL (MFRL) fails in high dimensions not because it lacks a learned dynamics model. Instead, it fails because the Bellman bootstrap, the self-referential loop at the heart of temporal-difference (TD) learning, becomes an unstable feedback system whose gain exceeds unity (Sutton & Barto, 1998; Kumar et al., 2021). In this regime, the critic amplifies noise faster than it accumulates signal. What makes this failure particularly insidious is that it is *self-induced by the learning itself*. As the agent improves, its policy produces actions in regions of the high-dimensional action space that the critic has never evaluated (Lyle et al., 2023). In 38- to 61-dimensional spaces, broad coverage becomes impossible. Each policy improvement step pushes the actor into regions where the critic must extrapolate, generating large TD errors that reflect genuine out-of-distribution (OOD) responses rather than noise (Fujimoto et al., 2019). Under standard training, these errors propagate through the bootstrap chain and corrupt the value landscape that the agent relies on to continue improving.

We trace this instability to three gains in the Bellman feedback loop, each of which inflates as the action dimensionality grows and all of which must be simultaneously controlled for stable learning. The first is the **signal gain** of the bootstrap target. In a one-step Bellman update, the

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.

target $r + \gamma Q(s', a')$ is almost entirely composed of the critic's own recursive estimate. The single scalar reward provides only weak external grounding. *Multi-step returns* inject a window of real environmental reward into the target, providing segment-level supervision that anchors the critic against recursive drift while preserving the Bellman learning framework (Peng & Williams, 1994; Hessel et al., 2018; Yarats et al., 2021; Hernandez-Garcia & Sutton, 2019).

The second is the **gradient gain** of the critic. The actor updates by ascending $\nabla_a Q(s, a)$. In high-dimensional spaces, the local Lipschitz constant of an unconstrained critic grows through chain-rule Jacobian multiplication, and this direction becomes unreliable under small action perturbations. *LayerNorm* (Ba et al., 2016) acts as a Jacobian conditioner, dynamically compressing the critic's gain so that $\nabla_a Q(s, a)$ becomes a bounded and approximately isotropic direction field that the actor can follow.

The third is the **error gain** of the self-referential update. When the agent improves and its new actions produce large TD errors in the critic's blind spots, the squared loss translates these errors into proportionally large gradients. Because the updated critic generates the next round of targets, a single OOD spike can cascade through the bootstrap chain. The *Huber loss* caps the gradient magnitude for large errors, allowing genuine learning signal to pass while preventing OOD spikes from destroying the value landscape that the other two components have made informative and navigable (Huber, 1992; Mnih et al., 2015).

The aforementioned three interventions (multi-step returns, LayerNorm, and Huber loss) add no additional networks or architectural changes to the base algorithm, requiring only negligible parameter overhead (LayerNorm affine parameters). While these components are individually established, our contribution lies in the gain-based decomposition that assigns them distinct roles in the Bellman feedback loop and yields testable predictions about their effects. In particular, the framework predicts distinct failure signatures when different gain controls are removed, a gap between critic normalization and simple magnitude clipping, and stability under substantially longer training. On 12 primary high-dimensional control benchmarks, including Dog (38-dim) and Humanoid (21-dim) from DeepMind Control (Tassa et al., 2018), and H1Hand (61-dim) from HumanoidBench (Sferrazza et al., 2024), structurally repaired TD3 and SAC achieve strong performance and are competitive with state-of-the-art model-based methods including TD-MPC2.

The main contributions of this work are summarized as follows:

1. We identify the Bellman bootstrap as a self-referential feedback system in high-dimensional action spaces and decompose its instability into three gain mechanisms associated with target grounding, critic sensitivity, and error amplification.

2. We present *Gain-Compressed Repair* (GCR), a lightweight, architecture-preserving instantiation using multi-step returns, LayerNorm, and Huber loss, each targeting a distinct gain. The framework explains their complementary roles and predicts distinct failure modes when individual gains are left uncontrolled.

3. We evaluate these predictions across 12 primary high-dimensional tasks and six supplementary HumanoidBench-H1 tasks using TD3, with SAC for cross-algorithm validation, together with component ablations, long-horizon training, width variation, and clipping controls. The results show that simple model-free agents can attain strong performance up to 61-dimensional control.

2 Related Work

Model-free continuous control. Off-policy actor-critic methods including DDPG (Lillicrap et al., 2015), TD3 (Fujimoto et al., 2018), and SAC (Haarnoja et al., 2018) establish the paradigm for continuous control. TD3 introduces mechanisms that implicitly address aspects of the instability studied in this work: clipped double Q-learning suppresses value overestimation, target policy smoothing provides limited robustness against OOD critic evaluations, and delayed policy updates ensure the critic stabilizes before the actor follows its gradients. However, these mechanisms were developed for low-dimensional benchmarks where the instability remains subcritical. Recent work has revisited critic architecture to improve stability. CrossQ (Bhatt et al., 2024) adopts BatchNorm in the critic, while BRO (Nauman et al., 2024) and XQC (Palenicek et al., 2026) likewise rely on normalization layers in the critic; these works confirm the stabilizing effect of critic normalization without identifying

the specific failure mode it addresses. SimbaV2 (Lee et al., 2025) stabilizes high update-to-data ratio training by projecting weights and features onto the unit hypersphere, at the cost of a substantially more complex network topology than the standard LayerNorm. FastTD3 (Seo et al., 2025) combines parallel simulation, large-batch updates, and a distributional critic to accelerate humanoid training, focusing on engineering scalability rather than diagnosing the underlying instability. Our work unifies these observations under a single diagnostic framework: identifying which gain a modification compresses explains why it helps and where it remains insufficient in isolation (Appendix F).

Components in isolation. Each of our three interventions has appeared in prior work, though in different contexts and without a unifying rationale. Huber loss was used in the original DQN (Mnih et al., 2015) for discrete control and was largely abandoned when the field moved to continuous action spaces, where MSE became the default; recent work by MRQ (Fujimoto et al., 2025) revisits Huber loss alongside multi-step returns and LayerNorm for continuous control, though without the unifying gain-compression diagnosis we develop here. Multi-step returns appear in Rainbow (Hessel et al., 2018) for discrete control and in DrQ-v2 (Yarats et al., 2022) for image-based continuous tasks, understood primarily through the bias-variance lens. Geometric policy constraints such as GAC (Lin, 2026) compress feedback-loop gain at the action boundary, limiting the magnitude of OOD actions the critic must evaluate. Our contribution is not any individual component but the identification of *why* they are simultaneously necessary: each compresses a distinct gain in the Bellman feedback loop, and the multiplicative structure of the loop stability condition (Appendix B) means that no single intervention suffices.

Model-based RL. The perceived failure of model-free methods in high-dimensional control has motivated world-model approaches such as DreamerV3 (Hafner et al., 2025) and planning-based methods such as TD-MPC2 (Hansen et al., 2024). Hybrid approaches including TCRL (Zhao et al., 2023), BMPC (Wang et al., 2025), and BOOM (Zhan et al., 2025) combine learned dynamics with policy optimization. From the perspective of our analysis, these methods address the signal-gain problem through a complementary mechanism: learned rollouts generate extended sequences of synthetic supervision that smooth the training signal reaching the critic, at the cost of compounding model error over the rollout horizon. Our results show that when the Bellman feedback loop is directly stabilized, model-free methods match or exceed model-based performance without incurring this additional source of error.

Beyond Bellman. A separate line of work circumvents the difficulties of continuous Bellman updates by discretizing the action space (Tang & Agrawal, 2020), training policies via supervised regression on high-return trajectories (Scannell et al., 2024), or sequence modeling over offline datasets (Chen et al., 2021). These approaches sidestep the feedback-loop instability we identify but also forfeit the bootstrap's core advantage: the ability to propagate value information beyond observed returns and to generalize across states. Our work instead takes the position that the Bellman framework is not fundamentally broken: its instability in high dimensions reflects a specific structural flaw that can be repaired.

3 METHODOLOGY

Our *Gain-Compressed Repair* (GCR) scheme adds no networks, learned components, or architectural changes to standard actor-critic algorithms. It consists of three modifications (multi-step returns, LayerNorm, and Huber loss) to the Bellman training loop, each compressing one gain in the feedback system that TD learning forms.

We briefly recapitulate the feedback structure before describing each intervention. Let Q_θ denote the critic parameterized by θ, $Q_{\bar{\theta}}$ the target critic whose parameters $\bar{\theta}$ track θ via exponential moving average, and π_ϕ the actor. The Bellman target $y = r + \gamma Q_{\bar{\theta}}(s', \pi_\phi(s'))$ feeds into a loss that updates Q_θ, whose gradients in turn guide π_ϕ. As the actor improves, it produces novel actions that the critic must evaluate in regions it has never been trained on. The quality of learning depends on three quantities along this loop: how much real information the target contains (*signal gain*), how faithfully the critic's action-gradient reflects value structure (*gradient gain*), and how aggressively the loss translates TD errors into parameter changes (*error gain*). In high-dimensional action spaces, all three gains inflate simultaneously, and the loop becomes self-amplifying rather than self-correcting. The

composite stability condition governing these three gains is formalized in Appendix B.4. Our three interventions each compress exactly one gain.

3.1 SIGNAL GAIN: MULTI-STEP RETURNS

In the standard one-step target $y = r+\gamma Q_{\bar{\theta}}(s', a')$, the scalar reward r is the only external information entering the update; the bootstrap term is the critic's own recursive estimate. Without bootstrap, TD learning cannot propagate the value beyond immediate reward. With bootstrap alone, the critic trains predominantly on its own predictions. In high-dimensional spaces where the critic is necessarily inaccurate, this recirculation degenerates into noise amplification.

Multi-step returns resolve this tension by replacing n rounds of recursive self-evaluation with n steps of observed environmental reward, as follows:

$$y^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n Q_{\bar{\theta}}(s_{t+n}, a_{t+n}). \tag{1}$$

Here and throughout, rewards r_{t+k} are collected under the behavior policy and stored in the replay buffer. The bootstrap action is given by $a_{t+n} = \pi_\phi(s_{t+n})$, i.e., the current policy evaluated at the next state. We use this standard uncorrected off-policy n-step target without importance weighting. Each reward r_{t+k} provides an independent observation of the trajectory segment $[t, t+n)$ that is absent from the bootstrap term. This goes beyond a bias–variance trade-off: it injects *segment-level supervision* into a system otherwise starved of external signal. The bootstrap mechanism remains, preserving the Bellman framework's ability to generalize beyond observed returns, but is now anchored by real environmental feedback rather than recursive estimates. As a result, the signal gain of the bootstrap target decreases approximately linearly with n (Proposition B.1, Appendix B). An information-theoretic interpretation is provided in Appendix D.

Remark 3.1. Injecting richer signal into the target shifts the bottleneck. With multi-step returns, the critic receives meaningful supervision; the question becomes whether it can *reliably process* that supervision. This leads directly to the next intervention.

3.2 GRADIENT GAIN: CRITIC LAYERNORM

The actor improves by ascending the critic's action gradient $\nabla_a Q_\theta(s, a)|_{a=\pi(s)}$. For this gradient to be informative, it must reflect relative value of nearby actions. Small perturbations in action should therefore produce proportionally small and geometrically consistent changes in the predicted value. For an L-layer critic, the action gradient decomposes via the chain rule:

$$\nabla_a Q_\theta = W_1^\top\big|_{\text{action}} D_1 \, W_2^\top \cdots D_{L-1} \, w_L, \tag{2}$$

where W_i denotes the weight matrix of layer i, $z_i = W_i h_{i-1} + b_i$ is the pre-activation vector with h_0 being the network input (state-action concatenation), $D_i = \mathrm{diag}(\sigma'(z_i))$ the diagonal matrix of activation derivatives, and w_L the final-layer weight vector (since Q is scalar). The spectral norm of this product is bounded by $\prod_i \|W_i\|\|D_i\|$, which in an unconstrained network can grow arbitrarily with depth and input dimensionality. In high-dimensional action spaces, this implies that individual action dimensions may be amplified by orders of magnitude relative to others, causing the gradient to align with these amplified directions rather than with directions of genuine value improvement.

A subtler issue arises from the interaction with the optimizer. The learning rate is a fixed scalar (e.g., $\eta = 3 \times 10^{-4}$ for Adam (Kingma & Ba, 2014)), while gradient magnitudes vary widely as the critic encounters OOD actions in new regions of the value landscape. Without normalization, the same step size that yields a mild update in well-explored regions can shatter the critic's feature geometry in novel regions where Jacobian norms are an order of magnitude larger.

LayerNorm addresses both problems simultaneously. It is applied after each hidden linear transformation and before the nonlinearity, i.e., Linear $\to$ LayerNorm $\to$ SiLU. For the pre-activation $z_i = W_i h_{i-1} + b_i$, LayerNorm gives $\hat{z}_i = (z_i - \mu_{z_i})/\sigma_{z_i} \odot g_i + c_i$, followed by $h_i = \mathrm{SiLU}(\hat{z}_i)$, where g_i and c_i are learnable scale and shift parameters. The factor $1/\sigma_{z_i}$ adaptively rescales the layerwise Jacobian, mitigating activation-scale growth and uncontrolled amplification through the Jacobian product (Proposition C.1, Appendix C).

Remark 3.2. LayerNorm is not a generic training stabilizer in this context—it is a *Jacobian conditioner*. It dynamically compresses the critic's gain so that the action-gradient reflects value structure faithfully across the full range of training conditions. With the signal anchored (multi-step) and the gradient geometry corrected (LayerNorm), the remaining question is whether the optimization process can update the critic without destroying the landscape it has built.

3.3 Error Gain: Huber Loss

The Bellman target depends on the critic being updated. A large TD error produces a large gradient, which changes the critic's parameters and shifts the target distribution for the next batch. In supervised learning this self-referential structure is absent, and large errors are simply informative. In bootstrapped RL they can be catastrophic.

The risk is greatest when the agent is improving. As the actor discovers better actions, these actions fall in regions of the action space where the critic has been trained on few or no samples. Predictions in such regions are extrapolations, and the resulting TD errors can be large. This does not imply that the critic is globally incorrect; rather, it has never been supervised in this neighborhood.

Under squared loss, such errors produce gradients proportional to their magnitude. The resulting update distorts the critic's value landscape in already well-learned regions. Because the actor relies on this landscape for the next improvement step, it can be misdirected. In extreme cases the improvement is reversed and progress undermines its own foundation. The Huber loss represented by

$$\ell_\beta(\delta) = \begin{cases} \dfrac{\delta^2}{2\beta}, & |\delta| \le \beta \\ |\delta| - \dfrac{\beta}{2}, & |\delta| > \beta \end{cases} \tag{3}$$

breaks the destructive bootstrap chain, where $\delta = y^{(n)} - Q_\theta(s, a)$ is the TD error and $\beta > 0$ controls the transition between the quadratic and linear regimes. For small TD errors ($|\delta| \le \beta$), the loss is quadratic with curvature scaled by $1/\beta$, preserving sensitivity to genuine value misestimation. For large errors ($|\delta| > \beta$), typically arising from OOD evaluations during policy improvement, the gradient magnitude is capped at unity, preventing excessive energy injection into the bootstrap chain.

Formally, this limits the geometric error amplification possible under MSE to self-limiting arithmetic growth for large errors (Proposition B.2, Appendix B). A detailed analysis of this transition in high-dimensional action spaces is provided in Appendix E.

Remark 3.3. The three interventions play complementary roles: multi-step returns improve target grounding, LayerNorm conditions critic geometry, and Huber loss limits the influence of large TD errors. Ablations in Section 4.3 show that removing any component degrades performance, consistent with the small-gain perspective in Appendix B.4.

3.4 Instantiation

The three modifications apply identically to any off-policy actor-critic method: LayerNorm after each hidden critic layer, Huber loss replacing MSE, and n-step cumulative rewards stored in the replay buffer. We collect transitions from 8 parallel environments for batch diversity (Appendix B.5). No changes are made to the actor, the policy loss, or the exploration mechanism. The resulting architecture is deliberately minimal: a two-layer MLP with 256 hidden units per layer for both actor and critic, implemented within the CleanRL framework (Huang et al., 2022). All hyperparameters follow standard defaults except the three structural choices motivated above (n, β, and the use of LayerNorm). We instantiate on both TD3 and SAC using identical structural modifications; for SAC, only the entropy term in the target differs. Full hyperparameter tables are provided in Appendix A; the complete algorithm is summarized in Figure 1.

Algorithm: Structurally Repaired TD3

Input: environment with d_a-dimensional actions, replay buffer $\mathcal{B}$
Output: actor π_ϕ, critics $\{Q_{\theta_j}\}_{j=1}^{2}$
Modifications (highlighted): N-step buffer, LayerNorm critic, Huber loss ($\beta = 0.3$)

Critic architecture:
Linear → LayerNorm → SiLU → Linear → LayerNorm → SiLU → Linear

Data collection:
1. Execute $a_t = \pi_\phi(s_t) + \epsilon$, observe r_t, s_{t+1}
2. Accumulate n-step transition: store $(s_t, a_t, R_t^{(n)}, s_{t+n}, d_{t+n})$
   where $R_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k}$ and $d_{t+n} \in \{0, 1\}$ is the terminal indicator
   At episode boundaries, flush remaining transitions using available steps.

Training (each gradient step):
3. Sample minibatch from $\mathcal{B}$
4. Compute target action with policy smoothing:
   $a' \leftarrow \pi_{\phi'}(s_{t+n}) + \text{clip}(\epsilon', -c, c)$
5. Compute n-step TD target:
   $y \leftarrow R_t^{(n)} + \gamma^n (1 - d_{t+n}) \cdot \min_j Q_{\theta'_j}(s_{t+n}, a')$
6. Update critics with Huber loss:
   $\theta_j \leftarrow \theta_j - \eta \nabla_{\theta_j} \ell_\beta\big(Q_{\theta_j}(s_t, a_t) - y\big)$
7. Every $d_{\text{policy}} = 2$ steps:
   Update actor: $\phi \leftarrow \phi + \eta \nabla_\phi Q_{\theta_1}(s, \pi_\phi(s))$
   Soft-update all targets: $\theta'_j \leftarrow \tau\theta_j + (1-\tau)\theta'_j$, $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$

Figure 1: Structurally repaired TD3 with n-step returns, LayerNorm critic, and Huber loss. Actor, exploration, and policy loss are unchanged. The SAC variant differs only by the entropy term.

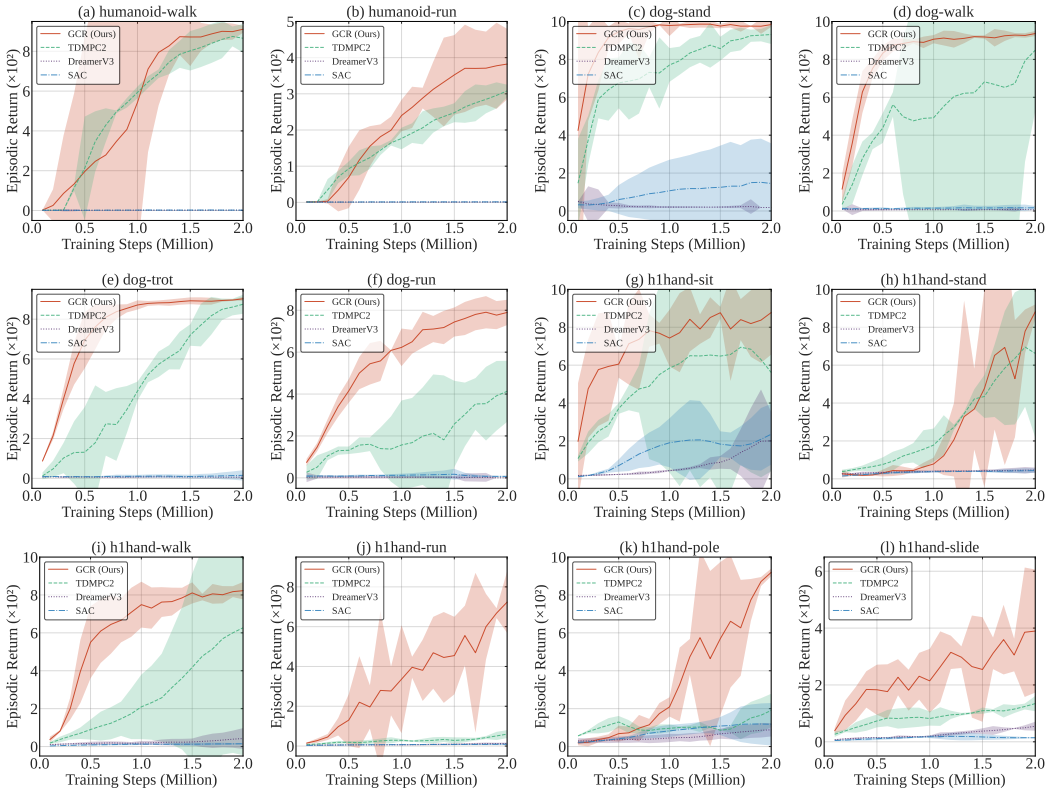

Figure 2: **Learning curves on 12 high-dimensional continuous-control tasks.**

Table 1: **Average Return on 6 DMC Hard and 6 H1Hand tasks.** mean ± 95% CI over 5 seeds.

Task	SAC	DreamerV3 (2M)	~(10M)	TD-MPC2 (2M)	~(4M)	GCR (Ours)
DMC Hard (6 tasks)						
Humanoid-Walk	1.9 ± 0.9	1.4 ± 0.8	755.6 ± 25.2	865.7 ± 72.0	884.8 ± 8.3	$\mathbf{889.7 \pm 62.4}$
Humanoid-Run	1.3 ± 0.2	0.9 ± 0.5	353.5 ± 33.2	308.0 ± 17.3	316.2 ± 9.2	$\mathbf{387.2 \pm 55.5}$
Dog-Stand	145.2 ± 211.8	18.2 ± 0.0	35.4 ± 10.8	931.2 ± 50.0	936.4 ± 7.6	$\mathbf{987.7 \pm 9.8}$
Dog-Walk	16.9 ± 9.7	5.8 ± 0.0	9.1 ± 0.6	848.4 ± 323.9	885.0 ± 74.8	$\mathbf{925.8 \pm 37.4}$
Dog-Trot	15.2 ± 25.9	6.2 ± 1.8	8.4 ± 0.9	875.0 ± 48.0	884.4 ± 22.2	$\mathbf{905.1 \pm 9.0}$
Dog-Run	6.9 ± 3.1	0.0 ± 0.0	4.3 ± 3.2	412.9 ± 146.0	427.0 ± 57.9	$\mathbf{791.0 \pm 28.2}$
H1Hand (6 tasks, 61-dim action)						
H1Hand-Sit	235.4 ± 160.3	200.4 ± 152.4	699.1 ± 177.2	561.3 ± 521.9	693.7 ± 249.9	$\mathbf{903.9 \pm 105.4}$
H1Hand-Stand	43.5 ± 11.8	46.4 ± 13.2	845.4 ± 27.3	662.0 ± 444.4	728.7 ± 121.9	$\mathbf{961.0 \pm 19.1}$
H1Hand-Walk	14.6 ± 2.8	42.3 ± 54.9	744.0 ± 28.7	626.6 ± 672.4	644.2 ± 281.1	$\mathbf{821.6 \pm 120.4}$
H1Hand-Run	9.2 ± 2.4	12.7 ± 4.8	622.4 ± 66.7	50.9 ± 20.7	66.1 ± 8.1	$\mathbf{643.1 \pm 228.4}$
H1Hand-Pole	117.6 ± 112.1	88.2 ± 35.1	577.4 ± 62.3	190.6 ± 87.9	207.5 ± 35.6	$\mathbf{841.2 \pm 177.3}$
H1Hand-Slide	14.2 ± 1.3	54.8 ± 15.8	367.6 ± 29.7	134.4 ± 26.9	141.3 ± 15.6	$\mathbf{498.1 \pm 218.8}$

4 EXPERIMENTS

We evaluate the proposed GCR scheme on 12 high-dimensional control tasks to address: (i) Performance parity, (ii) Scalability (21D→61D), and (iii) Component necessity.

4.1 EXPERIMENTAL SETUP

Environments. We select 12 tasks spanning three embodiments of increasing action-space complexity: Humanoid (21-dim) from DeepMind Control Suite (Tassa et al., 2018), Dog (38-dim) from DeepMind Control Suite, and H1Hand (61-dim) from Humanoid Bench (Sferrazza et al., 2024). These tasks are chosen because vanilla model-free methods fail catastrophically on all of them, making them diagnostic benchmarks for the instability we analyze.

Baselines. We compare GCR method against three representative methods spanning the major algorithmic families: **SAC** (Haarnoja et al., 2018) as the standard model-free baseline; **DreamerV3** (Hafner et al., 2025) as the state-of-the-art reconstruction-based world model; and **TD-MPC2** (Hansen et al., 2024) as the state-of-the-art latent-space model-based method with online planning. For DreamerV3 and TD-MPC2, we report results at both 2M steps (matching our training budget) and their extended training budgets (10M iterations for DreamerV3; 4M iterations for TD-MPC2). For completeness, Appendix H provides a supplementary comparison with recent model-free methods including BRO, SimBaV2, and XQC on the 19-dimensional HumanoidBench-H1 suite.

Protocol. GCR is instantiated on TD3 with identical hyperparameters across all 12 environments: 8 parallel environments, batch size 256, hidden dimension 256, learning rate 3×10^{-4}, $\gamma = 0.99$, $\tau = 0.0025$, $n = 3$ for multi-step returns, and Huber $\beta = 0.3$. Unless otherwise noted, experiments are trained for 2M steps and evaluated using a deterministic policy every 100K steps over 10 episodes. For the main comparison in Table 1, GCR results are reported as mean returns with 95% confidence intervals over 5 seeds.

4.2 MAIN RESULTS

Table 1 and Figure 2 summarize performance across all 12 tasks.

DMC Hard: 21–38 dimensional actions. On the Humanoid domain (21-dim), SAC and DreamerV3 at 2M steps both fail to learn meaningful locomotion, achieving near-zero returns. TD-MPC2 succeeds at 2M steps on Humanoid-Walk (865.7) but plateaus on Humanoid-Run (308.0). GCR attains the highest mean returns on both tasks (889.7 and 387.2), with means above TD-MPC2 under its extended training budget (884.8 and 316.2).

On the Dog domain (38-dim), the instability we diagnose is fully exposed. SAC collapses on all four tasks, with Dog-Run achieving only 6.9 ± 3.1. DreamerV3 fails at 2M steps and remains near zero even at 10M iterations, confirming that reconstruction-based world models struggle with high-dimensional proprioceptive control. TD-MPC2 at 2M steps achieves strong performance on

Dog-Stand (931.2) and Dog-Trot (875.0) but shows high variance on Dog-Walk (848.4 ± 323.9) and moderate performance on Dog-Run (412.9 ± 146.0). GCR attains the highest mean returns on all four Dog tasks, with Dog-Run reaching 791.0 ± 28.2, nearly double TD-MPC2's 2M-step result and exceeding its extended training result (427.0).

H1Hand: 61-dimensional actions. The H1Hand domain represents the most extreme test of our framework, with 61 actuated degrees of freedom including dexterous hand control. SAC and DreamerV3 at 2M steps produce negligible returns on most tasks. TD-MPC2 at 2M steps shows highly variable performance. For example, it achieves 561.3 ± 521.9 on H1Hand-Sit and 662.0 ± 444.4 on H1Hand-Stand, while remaining low on H1Hand-Run (50.9 ± 20.7) and H1Hand-Slide (134.4 ± 26.9).

GCR attains the highest mean return on all six H1Hand tasks. On H1Hand-Pole, GCR reaches 841.2 ± 177.3 compared to TD-MPC2's 190.6 ± 87.9. On H1Hand-Run, GCR achieves 643.1 ± 228.4 compared to TD-MPC2's 50.9 ± 20.7. These results show that the three-component repair remains effective at the highest action dimensionality tested, where model-based methods with learned dynamics and online planning show inconsistent performance.

Long-horizon stability. To test whether GCR merely delays instability, we extend four representative tasks to 10M environment steps. Performance remains high and continues to improve over longer training: Dog-Run increases from 791.0 to 859.8, Dog-Walk from 925.8 to 943.7, H1Hand-Run from 643.1 to 917.9, and H1Hand-Walk from 821.6 to 926.9. Performance remains high at 10M steps, providing no evidence of late-training collapse in these runs. These results indicate that GCR remains effective over substantially longer training horizons.

4.3 Ablation Studies

We evaluate the necessity of each component and the robustness of the Huber threshold on Dog-Run (38-dimensional actions) at 1M steps. Figure 3 and Table 2 summarize the results.

Table 2: **Ablation results on Dog-Run** (1M steps, 3 seeds). Δ: change from full GCR.

Configuration	Return	Δ
Component ablation (TD3)		
GCR (Full)	**622.9**$\pm$**89.8**	—
− Huber	563.5$\pm$49.6	−59
− LN-Q	437.4$\pm$80.0	−186
− n-step	398.7$\pm$53.9	−224
TD3 (vanilla)	92.1$\pm$121.4	−531
Huber threshold β		
$\beta = 0.3$	622.9$\pm$89.8	—
$\beta = 0.6$	679.3$\pm$32.2	+56
$\beta = 1.0$	638.6$\pm$85.8	+16
Cross-algorithm (SAC, 1M)	Dog-Run	H1Hand-Walk
GCR (SAC)	**700.2**$\pm$**14**	**735.0**$\pm$**173.7**
TD-MPC2	156.9$\pm$213	178.1$\pm$87
DreamerV3	3.1$\pm$5.4	39.8$\pm$5.9
SAC	11.9$\pm$8.2	37.7$\pm$3.0

Each component is individually necessary. Removing any single intervention degrades performance substantially (Table 2, top). Removing n-step returns (-224) or LayerNorm (-186) produce the largest drops, confirming that the signal gain and gradient gain are the primary bottlenecks. Removing Huber loss (-59) produces a smaller but meaningful degradation; notably, the w/o Huber configuration has not yet plateaued at 1M steps, and we expect the gap to widen at longer training horizons as the self-induced instability of MSE becomes more pronounced in the high-performance regime (Appendix B). Vanilla TD3 achieves only 92.1, confirming that the instability is catastrophic without any structural repair.

Huber threshold is robust. Performance is stable across $\beta \in \{0.3, 0.6, 1.0\}$ (Table 2, middle), with all three values achieving >620 reward. This robustness is expected from the stability analysis: the reward-error scale ($\sim$0.3) and bootstrap-error scale ($\sim$3.0) are separated by an order of magnitude, creating a wide operating range for β between them (Appendix E).

LayerNorm versus gradient clipping. We further test whether the benefit of LayerNorm can be reproduced by directly limiting gradient magnitude. Keeping the multi-step target and Huber loss unchanged, we replace critic LayerNorm with separately tuned critic-gradient clipping or action-gradient clipping on four representative tasks. LayerNorm achieves the highest return on all four tasks (Table 3), indicating that its benefit is not explained by magnitude control alone. This result is consistent with the proposed role of LayerNorm as a conditioner of the critic's local geometry rather than merely a scalar gain limiter.

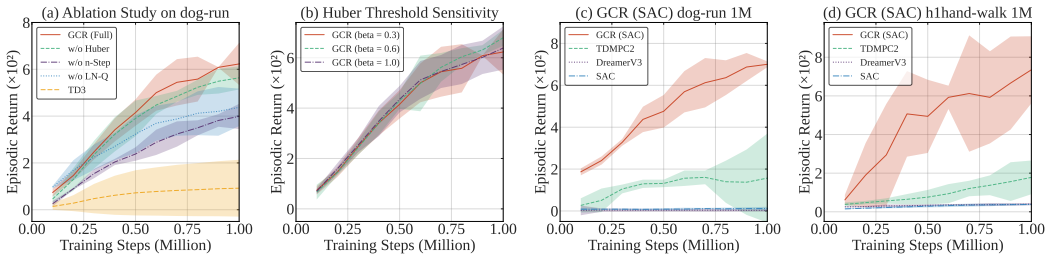

Figure 3: **Ablation and cross-algorithm validation on Dog-Run (38-dim).** (a) Component ablation: removing any element degrades performance, with vanilla TD3 near zero. (b) Huber threshold sensitivity: performance is stable for $\beta \in \{0.3, 0.6, 1.0\}$. (c) SAC-GCR on Dog-Run reaches 700+ reward, while vanilla SAC and DreamerV3 fail. (d) SAC-GCR on H1Hand-Walk (61-dim) shows the same cross-algorithm generalization.

Table 3: LayerNorm versus clipping controls at 1M steps. Mean ± 95% CI over 3 seeds. Clipping thresholds are tuned separately for each variant.

Task	LayerNorm	Critic clip	Action-grad. clip
Dog-Run	**632.6±73.8**	448.5±5.0	386.4±28.2
Dog-Walk	**917.8±10.9**	786.2±46.7	798.5±58.6
H1Hand-Run	**375.7±183.0**	361.3±98.9	335.0±250.4
H1Hand-Walk	**776.0±67.2**	621.6±389.1	543.4±68.6

Cross-algorithm generalization. Applying the identical three modifications to SAC yields dramatic improvements on both Dog-Run and H1Hand-Walk (Table 2, bottom). GCR (SAC) achieves 700.2 on Dog-Run, surpassing TD-MPC2 (156.9) by $4.5\times$ and vanilla SAC (11.9) by $59\times$, and achieves 735.0 on the 61-dimensional H1Hand-Walk, where all baselines remain below 180. These results indicate that the instability stems from the Bellman bootstrap rather than algorithm-specific design, and that the three-gain compression benefits off-policy methods in general.

5 Conclusion

This paper identifies a key source of failure for MFRL in high-dimensional continuous control. Our results suggest that a central bottleneck is not simply the lack of a world model, but a structural deficiency in the Bellman bootstrap: as action dimensionality grows, the recursive TD update can simultaneously starve itself of external signal, corrupt critic gradients through OOD extrapolation, and amplify errors through unchecked self-reference. We show that this cycle can be substantially mitigated by a simple, architecture-preserving repair: multi-step returns, LayerNorm on the critic, and Huber loss target the corresponding stages of the feedback loop, enabling model-free TD3 and SAC to achieve strong performance across 21- to 61-dimensional action spaces on a single consumer GPU, matching or surpassing model-based approaches on the studied tasks under comparable budgets. The three gains characterize whether the Bellman backup can be learned reliably by iterative regression in continuous, high-dimensional spaces: signal gain determines whether the target carries sufficient external grounding; gradient gain determines whether the learned approximation transmits value structure faithfully; and error gain determines whether residual corrections remain controlled rather than self-amplifying.

Limitations and future directions. Our analysis focuses on the state-based setting; extending the gain-compression perspective to settings where observations are mediated by a learned encoder (e.g., image-based control) is a natural extension. In addition, the interaction among the three gains is currently supported mainly by ablations. The small-gain perspective developed in Appendix B.4 suggests that principled algorithm design may benefit from explicitly monitoring and compressing loop gains during training. More broadly, understanding the signal structure of the training loop, namely how information enters, where it is destroyed, and how it can be preserved, may prove as productive as scaling model capacity.

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

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

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 *International conference on learning representations*, volume 2024, pp. 55293–55311, 2024.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

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, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

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.

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, 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.

J Fernando Hernandez-Garcia and Richard S Sutton. Understanding multi-step deep reinforcement learning: A systematic study of the dqn target. *arXiv preprint arXiv:1901.07510*, 2019.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

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.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Aviral Kumar, Rishabh Agarwal, Tengyu Ma, Aaron Courville, George Tucker, and Sergey Levine. Dr3: Value-based deep reinforcement learning requires explicit regularization. *arXiv preprint arXiv:2112.04716*, 2021.

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.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

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

Clare Lyle, Zeyu Zheng, Evgenii Nikishin, Bernardo Avila Pires, Razvan Pascanu, and Will Dabney. Understanding plasticity in neural networks. In *International Conference on Machine Learning*, pp. 23190–23211. PMLR, 2023.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015. doi: 10.1038/nature14236.

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.

Jing Peng and Ronald J Williams. Incremental multi-step q-learning. In *Machine Learning Proceedings 1994*, pp. 226–232. Elsevier, 1994.

Aidan Scannell, Kalle Kujanpää, Yi Zhao, Mohammadreza Nakhaei, Arno Solin, and Joni Pajarinen. iqrl–implicitly quantized representations for sample-efficient reinforcement learning. *arXiv preprint arXiv:2406.02696*, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. arXiv:1707.06347.

Younggyo Seo, Carmelo Sferrazza, Haoran Geng, Michal Nauman, Zhao-Heng Yin, and Pieter Abbeel. FastTD3: Simple, fast, and capable reinforcement learning for humanoid control. *arXiv preprint arXiv:2505.22642*, 2025.

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

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*, volume 1. MIT Press, Cambridge, MA, 1998.

Yunhao Tang and Shipra Agrawal. Discretizing continuous action space for on-policy optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5981–5988, 2020.

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.

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

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations*, 2021.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_SJ-_yyes8.

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

All experiments use identical hyperparameters across all 18 environments with no per-task tuning. The implementation is based on CleanRL (Huang et al., 2022) with three modifications to the standard TD3 codebase: LayerNorm added after each hidden linear transformation of the critic, before the SiLU activation, the replay buffer replaced with an n-step variant, and MSE loss replaced with Huber loss ($\beta = 0.3$). All other hyperparameters retain CleanRL defaults. Table A.1 lists the complete configuration; Figure 1 provides pseudocode.

A.1 Hyperparameters

Table A.1: Hyperparameters. All values are shared across TD3 and SAC variants and across all 18 environments. Entries marked with † are SAC-specific.

Parameter	Value
Architecture	
Hidden layers	2
Hidden dimension	256
Critic activation	SiLU
Critic normalization	LayerNorm (per hidden layer)
Actor activation	ReLU
Actor normalization	None
Optimization	
Optimizer	Adam
Learning rate (actor & critic)	3×10^{-4}
Batch size	256
Critic loss	Huber ($\beta = 0.3$)
RL-specific	
Discount γ	0.99
Target network update τ	0.0025
N-step returns n	3
Replay buffer size	10^6
Learning starts	25,000 steps
Parallel environments	8
TD3-specific	
Policy noise	0.2
Noise clip	0.5
Exploration noise	0.1
Policy update frequency	2
SAC-specific†	
Entropy coefficient	Auto-tuned
Target entropy	$-d_a$
Evaluation	
Evaluation frequency	Every 100,000 steps
Evaluation episodes	10
Evaluation policy	Deterministic (no exploration noise)
Seeds	5

A.2 Reference Implementation

The three structural modifications require fewer than 40 lines of code changes relative to the CleanRL TD3 baseline. The architecture remains deliberately minimal: both actor and critic use the standard two-layer MLP with 256 hidden units per layer, and the rest of the training pipeline is unchanged.

For reproducibility, we provide the complete single-file training script in the supplementary material. The key modifications are highlighted below.

Critic LayerNorm.

```
# QNetwork: Linear -> LayerNorm -> SiLU per hidden layer
self.fc1 = nn.Linear(obs_dim + action_dim, 256)
self.ln1 = nn.LayerNorm(256) # added
self.fc2 = nn.Linear(256, 256)
self.ln2 = nn.LayerNorm(256) # added
self.fc3 = nn.Linear(256, 1)

# Forward: x -> fc1 -> ln1 -> silu -> fc2 -> ln2 -> silu -> fc3
```

N-step returns.

```
# Replay buffer accumulates n-step transitions:
# R_n = sum(gamma**k * r_{t+k} for k in range(n))
# stores (s_t, a_t, R_n, s_{t+n}, d_{t+n})
rb = NStepReplayBuffer(..., n_step=3, gamma=0.99)

# TD target with n-step bootstrap:
y = R_n + (1 - d) * (gamma ** n) * min_Q_target
```

Huber loss.[1]

```
# Replace: loss = F.mse_loss(Q_pred, y)
loss = F.smooth_l1_loss(Q_pred, y, beta=0.3)
```

The training script (`gcr_td3.py`, 500 lines) and training logs are included in SM.

A.3 Environment Specifications

Table A.2: Environment specifications for all 18 benchmarks. DMC tasks use the DeepMind Control Suite (Tassa et al., 2018); H1Hand and H1 tasks use HumanoidBench (Sferrazza et al., 2024).

Suite	Embodiment	Tasks	Obs dim	Act dim
DMC	Humanoid	Walk, Run	67	21
DMC	Dog	Stand, Walk, Trot, Run	223	38
HumanoidBench	H1Hand	Sit, Stand, Walk, Run, Pole, Slide	151	61
HumanoidBench	H1	Sit Simple, Stand, Walk, Run, Pole	51	19
HumanoidBench	H1	Slide	64	19

B Feedback Loop Stability Analysis

This appendix formalizes the Bellman feedback loop as an input–output system, derives a composite stability condition via the small-gain theorem, and explains why the resulting instability remained hidden in low-dimensional benchmarks.

B.1 The Self-Referential Dynamics of Bellman Learning

Temporal-difference learning differs from supervised learning in that the training target $y = r + \gamma Q_{\bar{\theta}}(s', \pi_\phi(s'))$ depends on the model being trained: the critic's output at one iteration becomes part of its own supervision at the next. A large TD error therefore triggers a parameter update that *changes the target distribution itself*. Whether this change is beneficial or harmful depends on the *loop gain*,

[1] We use PyTorch's `F.smooth_l1_loss`, i.e. the classical Huber loss rescaled by $1/\beta$; the rescaling is absorbed by Adam's adaptive step size and does not affect training.

the factor by which a perturbation is amplified or attenuated through one full update cycle. When the loop gain is below unity the system is self-correcting; when it exceeds unity, errors compound and the critic settles into an uninformative equilibrium that reflects its own noise rather than the environment's reward structure.

B.2 Three Gains Along the Loop

We decompose the loop gain into three factors, each corresponding to a stage of the Bellman update and each independently measurable during training.

Signal gain G_{sig}. The signal gain measures the ratio of self-referential to external information in the Bellman target. For the n-step target $y^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n Q_{\bar{\theta}}(s_{t+n}, a_{t+n})$, where $a_{t+k} = \pi_\phi(s_{t+k})$ throughout this appendix, we define

$$G_{\text{sig}} = \frac{\text{Var}[B_t^{(n)}]}{\text{Var}[R_t^{(n)}]}, \tag{4}$$

where $R_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k}$ is the reward sum and $B_t^{(n)} = \gamma^n Q_{\bar{\theta}}(s_{t+n}, \pi_\phi(s_{t+n}))$ is the bootstrap term. When $G_{\text{sig}} \gg 1$, the target's variability is dominated by the critic's recycled estimate, and the update carries negligible fresh information from the environment.

Under the first-order approximation that per-step rewards have approximately equal variance σ_r^2 and bootstrap variance σ_B^2 is approximately invariant to small changes in n (a reasonable assumption when the target network changes slowly), we obtain the following scaling:

Proposition B.1 (Signal gain scaling)**.** *Under the assumptions above,*

$$G_{\text{sig}}(n) \approx \frac{G_{\text{sig}}(1)}{n}. \tag{5}$$

That is, the ratio of self-referential to external variance in the Bellman target decreases approximately linearly with the number of return steps.

Proof. The reward-sum variance is $\text{Var}[R_t^{(n)}] = \text{Var}[\sum_{k=0}^{n-1} \gamma^k r_{t+k}] \approx n\,\sigma_r^2$, using approximate equal variance and uncorrelatedness. By assumption, $\text{Var}[B_t^{(n)}] \approx \sigma_B^2$. Therefore $G_{\text{sig}}(n) = \sigma_B^2/(n\,\sigma_r^2) = G_{\text{sig}}(1)/n$. □

Multi-step returns thus reduce the signal gain approximately linearly in n, not by suppressing bootstrap variance but by injecting n independent environmental observations that are entirely absent from the one-step target.

Gradient gain G_{grad}. The gradient gain is the local Lipschitz constant of the critic with respect to its action input:

$$G_{\text{grad}}(s) = \left\| \nabla_a Q_\theta(s, a)\big|_{a=\pi(s)} \right\|. \tag{6}$$

This quantity determines whether the actor's update direction is geometrically faithful to the true value landscape. When G_{grad} is large or highly anisotropic, the actor's gradient is dominated by amplified dimensions that reflect the critic's spectral artifacts rather than genuine value structure.

For an L-layer critic, the chain rule gives $\nabla_a Q = W_L^\top D_{L-1} W_{L-1}^\top \cdots D_1 W_1^\top|_{\text{action}}$, whose spectral norm is bounded by $\prod_i \|W_i\|\|D_i\|$. Without normalization, this product grows uncontrollably with both network depth and input dimensionality. With LayerNorm, each layer's effective spectral contribution is bounded by $\|g_i\|_\infty/\sigma_{h_i}$, which is controlled by the normalization regardless of how large $\|W_i\|$ grows during training (Appendix C, Proposition C.1).

Error gain G_{err}. The error gain captures how aggressively the loss function converts TD errors into parameter changes:

$$G_{\text{err}}(\delta) = |\ell'_\beta(\delta)|. \tag{7}$$

Under MSE ($\ell(\delta) = \delta^2$), $G_{\text{err}}(\delta) = 2|\delta|$, so the response grows linearly with the error magnitude. Under the smooth ℓ_1 loss used in our implementation, $G_{\text{err}}(\delta) = |\ell'_\beta(\delta)| = \min(|\delta|/\beta,\ 1)$, so the response is capped at unity for large errors.

In the self-referential Bellman setting, this distinction is critical. A large TD error under MSE produces a large update that shifts the target distribution, potentially generating even larger errors at the next iteration. Huber loss breaks this positive feedback by ensuring that no single error, however large, can inject more than a bounded amount of energy into the bootstrap chain.

B.3 Perturbation Propagation Analysis

We now formalize how perturbations propagate through the Bellman feedback loop, establishing the mechanism by which each gain contributes to instability.

Suppose the critic parameters at iteration t are perturbed by $\Delta\theta_t$ from some reference trajectory. We trace how this perturbation completes one full cycle.

Step 1: Target perturbation. The perturbation $\Delta\theta_t$ affects the target network (via Polyak averaging with coefficient τ) and perturbs the bootstrap target:

$$\Delta y_t = \gamma^n \nabla_{\bar{\theta}} Q_{\bar{\theta}}(s_{t+n}, a_{t+n})^\top \cdot \tau \Delta\theta_t. \tag{8}$$

Multi-step returns attenuate this perturbation in two ways: γ^n replaces γ, and the bootstrap term influences a smaller fraction of the total target.

Step 2: TD error perturbation. The perturbed TD error is:

$$\Delta\delta_t = \Delta y_t - \nabla_\theta Q_{\theta_t}(s, a)^\top \Delta\theta_t. \tag{9}$$

Step 3: Gradient perturbation. The perturbation to the parameter update depends on the loss function's second derivative:

$$\Delta(\nabla_\theta \mathcal{L}) = \ell''_\beta(\delta_t) \cdot \Delta\delta_t \cdot \nabla_\theta Q_{\theta_t}(s, a). \tag{10}$$

For MSE, $\ell''(\delta) = 2$ (constant, always active). For the smooth ℓ_1 loss with threshold β:

$$\ell''_\beta(\delta) = \begin{cases} 1/\beta, & |\delta| \le \beta, \\ 0, & |\delta| > \beta. \end{cases} \tag{11}$$

This is the sharpest distinction between the two losses: for large TD errors ($|\delta| > \beta$), the smooth ℓ_1 loss zeroes out the second-order loss coupling $\ell''_\beta = 0$, eliminating the dominant pathway through which perturbations are amplified. A residual pathway through the network Hessian $\nabla^2_\theta Q$ persists but is bounded and typically small relative to the first-order term.

Step 4: Parameter perturbation at $t+1$. Combining the above, the perturbation after one full iteration satisfies:

$$\|\Delta\theta_{t+1}\| \le \|\Delta\theta_t\| + \eta\,|\ell''(\delta_t)| \cdot |\Delta\delta_t| \cdot \|\nabla_\theta Q_{\theta_t}\|. \tag{12}$$

The consequence for error propagation across iterations is qualitatively different under the two losses:

Proposition B.2 (Error propagation: MSE vs. Huber)**.** *Consider successive Bellman updates where the next TD error depends on the current update through the bootstrap: $|\delta_{t+1}| \le |\delta_t| + c\|\Delta\theta_t\|$ for Lipschitz constant $c > 0$. The parameter perturbation satisfies $\|\Delta\theta_t\| = \eta\,|\ell'(\delta_t)| \cdot \|\nabla_\theta Q_{\theta_t}\|$. Let $c' = c\eta\|\nabla_\theta Q_{\theta_t}\|$. Then:*

Under MSE, $|\ell'(\delta_t)| = 2|\delta_t|$, so $\|\Delta\theta_t\| = 2\eta\,|\delta_t| \cdot \|\nabla_\theta Q_{\theta_t}\|$, giving

$$|\delta_{t+1}| \le (1 + 2c')\,|\delta_t|, \tag{13}$$

permitting geometric amplification in the corresponding worst-case bound when $c' > 0$.

Under smooth ℓ_1 loss with threshold β, $|\ell'_\beta(\delta_t)| = 1$ for $|\delta_t| > \beta$, so $\|\Delta\theta_t\| = \eta\,\|\nabla_\theta Q_{\theta_t}\| = c'/c$, giving

$$|\delta_{t+1}| \le |\delta_t| + c', \quad \textit{for } |\delta_t| > \beta, \tag{14}$$

*yielding **arithmetic (linear) growth**, which is self-limiting.*

Proof. $\|\Delta\theta_t\| = \eta\,|\ell'(\delta_t)| \cdot \|\nabla_\theta Q_{\theta_t}\| = 2\eta\,|\delta_t| \cdot \|\nabla_\theta Q_{\theta_t}\|$ under MSE, so $c\|\Delta\theta_t\| = 2c'\,|\delta_t|$ and therefore $|\delta_{t+1}| \le |\delta_t| + 2c'\,|\delta_t| = (1+2c')\,|\delta_t|$. Under smooth ℓ_1 with $|\delta_t| > \beta$, $\|\Delta\theta_t\| = \eta \cdot 1 \cdot \|\nabla_\theta Q_{\theta_t}\|$, so $c\|\Delta\theta_t\| = c'$ and $|\delta_{t+1}| \le |\delta_t| + c'$. □

The distinction is fundamental. Under MSE, a single OOD spike at iteration t propagates as $|\delta_t| \to (1+2c')|\delta_t| \to (1+2c')^2|\delta_t| \to \cdots$—geometric growth that can destabilize the critic. Under Huber, the same spike propagates as $|\delta_t| \to |\delta_t| + c' \to |\delta_t| + 2c' \to \cdots$—arithmetic growth with bounded influence, which prevents runaway amplification once the underlying cause dissipates.

Complementarity with multi-step returns. This analysis explains why multi-step returns *require* Huber loss to be effective. Multi-step targets inject richer environmental signal but also widen the tail of the target distribution (the longer reward sum occasionally produces large deviations). Under MSE, these wider tails are *squared* and fed back through the bootstrap chain with amplified force—the improved signal is destroyed by its own variance. Huber loss linearizes the tails, allowing the richer signal through while preventing its variance from triggering the geometric amplification cascade.

B.4 Composite Stability Condition

We now combine the three gains into a single stability criterion. Our analysis is motivated by the classical *small-gain theorem* for input–output stable feedback systems: if a collection of causal subsystems with finite gains γ_i forms a feedback interconnection satisfying $\prod_i \gamma_i < 1$, the closed loop is finite-gain stable. We do not claim a global small-gain theorem for deep actor–critic training, whose policy, critic, and replay distribution are nonlinear and time-varying. Instead, we use small-gain theory as a *local diagnostic principle*: high-dimensional Bellman learning becomes unstable when the effective product of three operator gains exceeds unity.

Operator decomposition. A single Bellman update cycle can be decomposed into three causal operators acting on perturbation sequences:

$$\Delta\theta_t \xrightarrow{T_t} \Delta\delta_t \xrightarrow{L_t} \Delta g_t \xrightarrow{U} \Delta\theta_{t+1}, \tag{15}$$

where T_t maps a parameter perturbation to a TD-error perturbation through the bootstrap target, L_t maps the TD-error perturbation to a gradient perturbation through the loss, and U applies the learning-rate update. Each operator has a well-defined induced gain:

$$\|T_t\| \le C_Q\,(1+\gamma^n\tau), \tag{16}$$

$$\|L_t\| = |\ell'_\beta(\delta_t)| \cdot C_Q, \tag{17}$$

$$\|U\| = \eta, \tag{18}$$

where $C_Q = \|\nabla_\theta Q_{\theta_t}(s,a)\|_2$ is the critic's parameter-gradient norm. Multi-step returns compress $\|T_t\|$ via $\gamma^n \ll \gamma$; LayerNorm bounds C_Q uniformly over training (Proposition C.1); Huber loss sets $|\ell''_\beta(\delta)| = 0$ for $|\delta| > \beta$, *severing the perturbation pathway entirely* for large OOD errors.

Note that the operator gain $\|L_t\|$ involves the second derivative $|\ell''_\beta|$, which governs perturbation amplification, whereas the error gain $G_{\text{err}} = |\ell'_\beta|$ (Section B.2) controls the gradient magnitude. Both are compressed by the smooth ℓ_1 loss: $|\ell'_\beta|$ is capped at 1, and $|\ell''_\beta|$ vanishes for $|\delta| > \beta$.

Proposition B.3 (Loop stability)**.** *Define the effective loop gain*

$$\Gamma_t = \|T_t\| \cdot \|L_t\| \cdot \|U\| = \eta\,C_Q^2\,(1+\gamma^n\tau) \cdot |\ell''_\beta(\delta_t)|. \tag{19}$$

If $\sup_t \Gamma_t < 1$*, the per-iteration amplification factor is below unity, indicating that the Bellman feedback loop is operating in a locally contractive regime. While gradient descent is a discrete-time process whose global dynamics are not captured by the algebraic small-gain bound, the condition* $\Gamma < 1$ *serves as a practical diagnostic: when it holds, perturbations are attenuated at each iteration; when it is violated, the loop admits exponential perturbation growth.*

This result is a *sufficient* condition for stability, not a necessary one; actual training may remain stable even when Γ_t occasionally exceeds unity, provided the time-averaged gain remains subcritical. The condition should be read as identifying the *structural* requirement for stability, namely that all three operator gains must be simultaneously controlled, rather than as a sharp quantitative threshold.

A deeper role for LayerNorm. Beyond compressing G_{grad}, LayerNorm plays a more fundamental role by ensuring that the finite-gain assumption required by the small-gain framework holds. Without LayerNorm, the critic gradient norm C_Q can grow unbounded during training (Section C.3), leading to $\sup_t \|T_t\| = \infty$ and $\sup_t \|L_t\| = \infty$. In this regime, the loop gain Γ is undefined and the small-gain criterion is inapplicable. LayerNorm is therefore not merely a gain compressor but a precondition for the stability framework.

Our three interventions each compress one factor:

$$\text{Multi-step:} \quad G_{\text{sig}} \to G_{\text{sig}}/n \quad \text{(Prop. B.1)} \tag{20}$$

$$\text{LayerNorm:} \quad G_{\text{grad}} \to O(\|g\|_\infty/\sigma_h) \quad \text{(Prop. C.1)} \tag{21}$$

$$\text{Huber:} \quad G_{\text{err}} \to \min(|\delta|/\beta,\ 1) \quad \text{(capped at unity)} \tag{22}$$

When all three are applied, the composite gain is compressed below the stability threshold across the full range of training conditions, restoring the self-correcting property of the Bellman loop even in 38–61 dimensional action spaces.

Why all three are necessary. If G_{sig} alone is reduced (multi-step without LayerNorm or Huber), the improved signal is corrupted by an ill-conditioned landscape (G_{grad} large) and amplified by unchecked error propagation (G_{err} large), so the product still exceeds 1. Similarly, reducing G_{grad} alone yields a smooth but uninformative landscape, and reducing G_{err} alone stabilizes optimization of a corrupted objective. Only simultaneous compression of all three factors brings the product below the critical threshold.

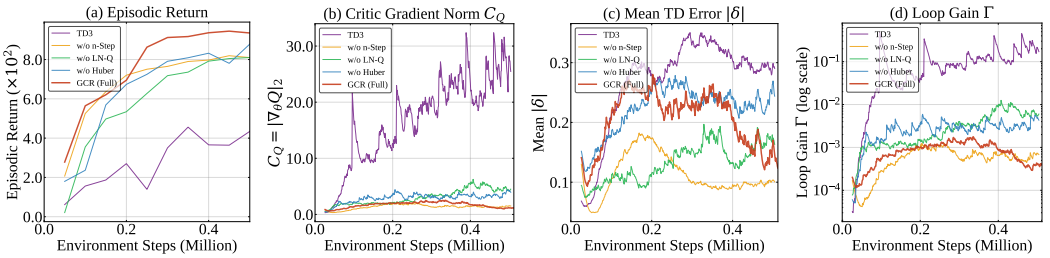

Figure B.1: **Loop gain diagnostics on Dog-Run.** (a) Episodic return across ablations. (b) Critic gradient norm $C_Q = \|\nabla_\theta Q\|_2$: TD3 grows unbounded; LayerNorm bounds C_Q, satisfying the finite-gain condition (Proposition B.3). (c) Mean absolute TD error $|\delta|$: removing Huber increases error magnitude. (d) Effective loop gain Γ: GCR remains low, while TD3 exhibits substantially larger and more frequent gain spikes.

Empirical validation. Figure B.1 tracks the loop gain components during training on Dog-Run across four configurations. The diagnostics confirm three predictions: (i) LayerNorm bounds C_Q, satisfying the finite-gain condition; (ii) Huber loss suppresses large TD errors that would otherwise amplify Γ through $|\ell''|$; (iii) the no n-step configuration attains low Γ trivially, as the critic is stable but starved of signal, consistent with the stagnation pathology in Appendix B.6.

B.5 Gain Coupling and Regime Transitions

The loop gain Γ_t (Eq. 19) treats the three operator gains as independent. In practice they are coupled: policy improvement pushes the actor into OOD regions, raising G_{err} and shifting the replay distribution that determines G_{sig}. This coupling can create regime transitions, where the system moves into a higher-gain regime as policy improvement pushes the actor further into the critic's extrapolation zone.

The three interventions play complementary roles: multi-step returns and LayerNorm set the *learning rate* (how fast signal accumulates), while Huber loss sets the *ceiling* (how far the agent can progress before self-induced OOD errors halt convergence). Only together do they sustain fast learning into the high-performance regime.

Parallel environments. While multi-step returns improve target *quality*, parallel collection (8 environments) improves batch *diversity* by reducing temporal correlation. This avoids aggressive replay reuse that can erode the Q-landscape the other three interventions have stabilized.

B.6 Empirical Signatures of Individual Gain Failures

Each ablation produces a qualitatively distinct pathology on Dog-Run (d_a=38), confirming that the three interventions address different mechanisms:

- **n-step** (high G_{sig}): smooth but uninformative landscape; gradients are faithful but point nowhere useful. Low reward, low Q-loss, and the critic converges to a self-consistent fixed point devoid of environmental signal.
- **LayerNorm** (high G_{grad}): meaningful signal reaches the critic, but the action-gradient is geometrically unreliable due to dimensional collapse. Similar low-reward plateau as the no-n-step case.
- **Huber** (high G_{err}): continued but less stable progress. Q-loss exhibits spikes coinciding with reward improvement, consistent with increased sensitivity to large TD errors in the high-performance regime (Proposition B.2).

These distinct failure modes are quantified in Section 4.3 and visualized in Figure G.1.

B.7 Why the Instability Was Hidden in Low Dimensions

The failure modes identified above have been latent in continuous actor-critic methods since their inception. Individual components addressing each gain have appeared in prior work, including LayerNorm and related normalizations in CrossQ (Bhatt et al., 2024) and SimBaV2 (Lee et al., 2025), Huber loss in MRQ (Fujimoto et al., 2025) and DQN (Mnih et al., 2015), and multi-step returns in DrQ-v2 (Yarats et al., 2022) and Rainbow (Hessel et al., 2018), but without a unifying diagnosis explaining why they are simultaneously necessary. This subsection explains why the standard benchmark suite, the default loss function, and the implicit robustness mechanisms of TD3 and SAC all conspire to keep the instability subcritical in low-dimensional tasks.

B.7.1 MSE Was the Right Choice for Low Dimensions

The universal adoption of MSE loss in continuous control is not an accident. In low-dimensional tasks ($d_a \leq 6$), the critic achieves broad coverage of the action space relatively quickly, and large TD errors predominantly signal genuine value structure, including discontinuities between stable and unstable behaviors, reward boundaries, and contact events. Under MSE, these large errors generate proportionally large gradients that rapidly imprint critical structure into the critic's landscape.

The transition occurs when action dimensionality grows large enough that the dominant source of large TD errors shifts from genuine value structure to OOD extrapolation. In low-dimensional spaces, the action manifold is approximately Euclidean: two actions that appear similar component-wise are genuinely close in ℓ_2 distance, and the critic can interpolate reliably between observed evaluations. In high-dimensional spaces, the geometry changes qualitatively. The volume of the action space grows exponentially with dimension, and two action vectors that differ only slightly per component can have large Euclidean distance ($\|\Delta a\|_2 \approx \sigma\sqrt{d_a}$ for per-component perturbation σ). The critic is forced to extrapolate rather than interpolate, and the resulting large TD errors carry no useful information, as they are geometric artifacts of high-dimensional action geometry rather than signals of value structure.

MSE amplifies these artifacts through the bootstrap chain; Huber loss attenuates them. In this sense, Huber loss serves as an implicit trust-region mechanism for the critic: it preserves the hard-won correspondence between learned representations and experienced data by preventing OOD spikes from overwriting well-learned regions of the value landscape. The analogy to PPO's ratio clipping is instructive: just as PPO prevents the policy from moving too far from the data that justified the current update, Huber loss prevents the critic from responding too aggressively to errors that arise from extrapolation beyond its training distribution.

The field never encountered this transition because the standard benchmark suite (Hopper d_a=3, Walker d_a=6, HalfCheetah d_a=6) never crossed the dimensional threshold at which OOD extrapolation dominates. The loss function that was optimal for $d_a \leq 8$ becomes catastrophic for $d_a \geq 20$.

B.7.2 TD3's Implicit Robustness Degrades with Dimension

TD3 incorporates two mechanisms that provide implicit robustness against the instabilities we identify: *target policy smoothing* and *delayed policy updates*. Both degrade in high dimensions.

Target policy smoothing adds Gaussian noise to the target action: $a' = \pi_{\phi'}(s') + \text{clip}(\epsilon, -c, c)$, $\epsilon \sim \mathcal{N}(0, \sigma^2 I_{d_a})$. The intended effect is to smooth the critic's value estimate over a local neighborhood of the target action, providing robustness against narrow peaks in the Q-landscape.

Proposition B.4 (Smoothing radius grows with dimension)**.** *For $\epsilon \sim \mathcal{N}(0, \sigma^2 I_d)$,*

$$\mathbb{E}\|\epsilon\|_2 = \sigma\sqrt{2}\,\frac{\Gamma((d{+}1)/2)}{\Gamma(d/2)} \approx \sigma\sqrt{d}. \tag{23}$$

With $\sigma = 0.2$: at $d_a = 3$, $\mathbb{E}\|\epsilon\|_2 \approx 0.35$ (a small local perturbation); at $d_a = 38$, $\mathbb{E}\|\epsilon\|_2 \approx 1.23$ (a large non-local displacement); at $d_a = 61$, $\mathbb{E}\|\epsilon\|_2 \approx 1.57$ (nearly spanning the full action range).

The mechanism that provides *local smoothing* in 3 dimensions becomes *non-local OOD evaluation* in 38+ dimensions. The critic is asked to evaluate actions far from any training data, producing extrapolation artifacts that are then averaged into the target—injecting noise rather than smoothing it.

A deeper interpretation of target policy smoothing clarifies this failure. The smoothing operation replaces a point evaluation $Q(s', \pi(s'))$ with a local integral $\mathbb{E}_\epsilon[Q(s', \pi(s') + \epsilon)]$, evaluating the critic over a neighborhood rather than at a single action. In low dimensions, this neighborhood is compact and well-covered by the replay buffer; the integral acts as a Laplacian-like smoother that reduces the Q-function's curvature and suppresses narrow peaks. In high dimensions, concentration of measure forces the perturbation onto a thin shell at radius $\approx \sigma\sqrt{d_a}$, far from the center. The "local integral" becomes a non-local surface average over a region where the critic has no training data. The smoothing mechanism, intended to reduce the critic's sensitivity to local artifacts, instead queries the critic in its extrapolation zone, contributing to the same OOD errors that Huber loss must absorb.

Delayed policy updates (updating the actor less frequently than the critic) provide the second implicit safeguard: they ensure the critic has time to stabilize before the actor follows its gradients. This helps when the critic's instability is transient (a few extra gradient steps suffice), but is insufficient when the instability is structural (the critic cannot stabilize because it is continuously receiving OOD evaluations from target policy smoothing and from the improving actor's novel actions).

B.7.3 SAC's Entropy Regularization Does Not Scale

SAC's entropy bonus $J_{\text{SAC}} = \mathbb{E}[Q(s,a) - \alpha \log \pi(a|s)]$ encourages *distributional smoothing*, i.e., diverse action sampling, but not *geometric smoothing*, i.e., a well-conditioned critic landscape. In high dimensions, this distinction is critical. A high-entropy Gaussian in $d_a = 61$ concentrates samples on a thin shell at radius $\approx \sigma_\pi\sqrt{d_a}$. As a result, the critic is evaluated at distant and isolated points, where predictions are dominated by extrapolation. More fundamentally, entropy constrains the *policy distribution* but does not constrain the geometry of Q. The critic may still exhibit sharp peaks, large curvature, and anisotropic gradients. LayerNorm and Huber loss address these geometric properties directly, whereas entropy regularization does not.

C LayerNorm as Jacobian Conditioner

LayerNorm is widely used as a generic training stabilizer. In this appendix we show that its role in the critic is far more specific: it acts as a *Jacobian conditioner* that dynamically compresses the critic's gain, restores effective representational dimensionality, and ensures that the action-gradient $\nabla_a Q$ is a bounded, approximately isotropic direction field. We analyze three progressively deeper effects.

C.1 Effect 1: Restoration of Effective Dimensionality

Consider a critic hidden layer with $d_h = 256$ units. In the absence of normalization, the pre-activation vector $h = W[s; a] + b$ inherits the scale disparities of its input: joint angles in $[-3, 3]$, velocities in $[-10, 10]$, and actions in $[-1, 1]$ produce activations whose magnitudes vary by orders of magnitude across neurons. After a nonlinearity such as SiLU, this disparity is preserved or amplified: neurons with large activations dominate the representation, while neurons with small activations contribute negligible gradients and are effectively dormant.

The result is a *dimensional collapse* of the critic's internal representation. Although the architecture provides 256 degrees of freedom, the effective dimensionality, defined as the number of neurons that meaningfully participate in representing the value function, can collapse to a small fraction of the nominal width. In a 38-dimensional action space, the critic needs sufficient representational resolution to distinguish the value of nearby actions that differ in subtle combinations of joint torques. A critic with a few effective dimensions cannot provide this resolution; it produces a value landscape that is nearly flat in most action directions, yielding uninformative gradients for the actor.

LayerNorm eliminates this failure mode by normalizing the activation vector to zero mean and unit variance at each layer:

$$\hat{h}_i = \frac{h_i - \mu_h}{\sigma_h}, \quad \mu_h = \frac{1}{d_h}\sum_j h_j, \quad \sigma_h = \sqrt{\frac{1}{d_h}\sum_j (h_j - \mu_h)^2}. \tag{24}$$

After normalization, all neurons operate at comparable scale. The learned scale and shift parameters g_i, b_i restore expressiveness without reintroducing the scale disparity: the network can learn to emphasize certain dimensions, but it starts from a uniform baseline rather than an arbitrary one determined by input statistics.

This is why removing LayerNorm produces slow learning rather than divergence: the critic still functions, but with drastically reduced effective capacity. The same number of gradient steps extracts less information from each minibatch, and the slow learning observed in our ablation (Section 4.3) reflects the bottleneck of learning a high-dimensional value function through a low-dimensional representational keyhole.

C.2 Effect 2: Jacobian Spectral Control via Three Projections

The deeper effect of LayerNorm operates on the critic's Jacobian—the matrix that determines how input perturbations propagate through the network to affect the output. The Jacobian of the LayerNorm operation with respect to its input is:

$$J_{\text{LN}} = \frac{\text{diag}(g)}{\sigma_h} \underbrace{\left(I - \frac{1}{d_h}\mathbf{1}\mathbf{1}^\top - \frac{1}{d_h}\hat{h}\hat{h}^\top \right)}_{P}. \tag{25}$$

The matrix P applies three geometric operations simultaneously, each serving a distinct stabilizing role.

Identity: faithful local transmission. The I term preserves the local gradient structure—small perturbations in the input are transmitted faithfully to the output. This ensures that LayerNorm does not destroy the critic's ability to represent complex value landscapes.

Mean projection: preventing value drift. The term $-\frac{1}{d_h}\mathbf{1}\mathbf{1}^\top$ projects out the mean component of the gradient. Gradients cannot flow in the direction of "all neurons increase simultaneously." This prevents the Q-function from undergoing global value drift, a failure mode where the critic's predictions shift up or down uniformly without changing their relative ordering, producing zero useful signal for the actor while consuming gradient budget.

Activation projection: breaking rich-get-richer dynamics. The term $-\frac{1}{d_h}\hat{h}\hat{h}^\top$ projects out the component along the current activation pattern. Gradients cannot flow in the direction of "amplify

the existing representation." Without this projection, neurons that are already strongly activated receive disproportionately large gradient updates, further strengthening their activation, creating a positive feedback loop that accelerates dimensional collapse. This projection explicitly breaks the rich-get-richer dynamic, forcing the gradient to explore directions orthogonal to the current representation.

Proposition C.1 (LayerNorm spectral bound). *Let* $\mathrm{LN}_{g,b}(h)$ *denote LayerNorm with learned gain* $g \in \mathbb{R}^{d_h}$ *and shift* $b \in \mathbb{R}^{d_h}$. *The Jacobian of LayerNorm with respect to its input satisfies*

$$\|J_{\mathrm{LN}}\|_2 \leq \frac{\|g\|_\infty}{\sigma_h}, \tag{26}$$

where σ_h *is the empirical standard deviation of the pre-activation vector* h. *In particular, this bound is independent of the spectral norm of the preceding weight matrix.*

Proof. The Jacobian decomposes as $J_{\mathrm{LN}} = \frac{\mathrm{diag}(g)}{\sigma_h} P$, where $P = I - \frac{1}{d_h}\mathbf{1}\mathbf{1}^\top - \frac{1}{d_h}\hat{h}\hat{h}^\top$. Since the normalized activations have zero mean ($\sum_i \hat{h}_i = 0$) and unit variance ($\frac{1}{d_h}\sum_i \hat{h}_i^2 = 1$), the vectors $\mathbf{1}$ and $\hat{h}$ are orthogonal and $\|\hat{h}\|^2 = d_h$. Therefore $\frac{1}{d_h}\mathbf{1}\mathbf{1}^\top$ and $\frac{1}{d_h}\hat{h}\hat{h}^\top$ are rank-one projections onto orthogonal directions, each with eigenvalue 1. Their sum is a rank-2 orthogonal projection, so P has eigenvalue 0 in these two directions and eigenvalue 1 in the remaining $d_h - 2$ directions, giving $\|P\|_2 = 1$. Therefore $\|J_{\mathrm{LN}}\|_2 \leq \frac{\|\mathrm{diag}(g)\|_2}{\sigma_h} \cdot \|P\|_2 = \frac{\|g\|_\infty}{\sigma_h}$. □

C.3 Effect 3: Dynamic Gain Compression

The spectral bound reveals the deepest role of LayerNorm: it acts as an *automatic gain controller* on the critic's forward pass.

In an unconstrained network, the chain-rule product $\nabla_a Q = W_L^\top D_{L-1} \cdots D_1 W_1^\top$ has spectral norm bounded by $\prod_i \|W_i\|\|D_i\|$. As training progresses and weights grow, this product can increase without bound, causing the gradient gain G_{grad} to inflate and the actor's update to become erratic.

With LayerNorm applied after each linear layer, the Jacobian contribution of layer i, which in an unconstrained network is simply W_i, becomes:

$$W_i \;\to\; \frac{\mathrm{diag}(g_i)}{\sigma_{h_i}} \cdot P_i \cdot W_i. \tag{27}$$

Note that LayerNorm acts on the activations $h_i = W_i x_i$, not on the weights directly; the expression above captures the effective transformation as seen by the chain-rule Jacobian. The critical observation is the $1/\sigma_{h_i}$ factor. When W_i has large spectral norm, the pre-activation $h_i = W_i x_i$ has large variance, which means σ_{h_i} is large, which means the $1/\sigma_{h_i}$ factor *automatically compresses the effective gain of that layer*. The larger the weight, the stronger the compression.

This is dynamic gain control in the signal-processing sense: the system automatically adjusts its amplification based on the input signal level. The critic can grow its weights freely during training (adapting to the changing value landscape) without the Jacobian product growing correspondingly. The decoupling between weight magnitude and gradient magnitude is what allows the same fixed learning rate to produce stable updates throughout training, even as the critic's internal representations undergo large changes.

C.4 Quantitative Illustration

To make the preceding analysis concrete, we provide a numerical example using our actual network configuration (2 hidden layers, 256 units, SiLU activation).

Without LayerNorm. For SiLU activations, $\|D_i\|_2 \leq 1.1$ (the maximum derivative of SiLU is approximately 1.1). Since the critic outputs a scalar Q-value, the final layer's weight is a row vector $w_L \in \mathbb{R}^{1 \times d_h}$, replacing the matrix W_L used in the general L-layer notation. With weight spectral norms $\|W_i\| = 5$ for the hidden layers (a moderate value observed in practice during later stages of training), the Jacobian bound for the two-hidden-layer critic is:

$$G_{\mathrm{grad}}^{(\text{no LN})} \leq \|w_L\| \cdot (\|W_2\| \cdot \|D_2\|) \cdot (\|W_1\| \cdot \|D_1\|) \leq \|w_L\| \cdot (5 \times 1.1)^2 \approx 30\,\|w_L\|. \tag{28}$$

If the final layer $\|w_L\| \approx 1$, the gradient gain can reach $\sim$30, and action dimensions aligned with the dominant singular vectors can experience amplification far beyond this average bound.

With LayerNorm. With LayerNorm, the Jacobian chain becomes:

$$G_{\text{grad}}^{(\text{LN})} \leq \|w_L\| \cdot \prod_{i=1}^{2} \frac{\|g_i\|_\infty}{\sigma_{h_i}} \cdot \|D_i\|. \tag{29}$$

With g_i initialized to 1 and σ_{h_i} determined by the preceding layer's weights and activations, each layer's contribution is bounded by $\|g_i\|_\infty \cdot \|D_i\|/\sigma_{h_i}$. In practice, $\sigma_{h_i} \geq 1$ throughout training (empirically verified), so each layer contributes at most $\sim$1.1 to the product, giving $G_{\text{grad}}^{(\text{LN})} \leq 1.1^2 \cdot \|w_L\| \approx 1.2\,\|w_L\|$—a $25\times$ reduction from the unconstrained case.

The weight spectral norm $\|W_i\| = 5$ has been completely absorbed by the normalization. If $\|W_i\|$ grows to 50 during training, the unconstrained bound inflates to $3000\,\|w_L\|$; the LayerNorm-regulated bound remains at $\sim 1.2\,\|w_L\|$, because the larger weights produce proportionally larger σ_{h_i}. This is the dynamic gain compression in action.

C.5 Consequence for the Actor

The composite effect of all three mechanisms is that $\nabla_a Q_\theta(s, a)$ becomes a *bounded, approximately isotropic direction field* over the action space:

- No single action dimension is amplified by orders of magnitude over others (isotropy from dimensional restoration).
- The field does not undergo global shifts (stability from mean projection).
- Existing activation patterns are not self-reinforced (diversity from the activation projection).
- The overall magnitude is automatically regulated (boundedness from gain compression).

The actor navigating this direction field receives reliable, geometrically faithful information about which action perturbations increase value across all 38 or 61 dimensions simultaneously, not just the few dimensions that an unconstrained critic happens to amplify.

This is why we call LayerNorm a *Jacobian conditioner* rather than a training stabilizer: its role is not to prevent divergence (the critic does not diverge without it) but to ensure that the gradient the actor receives is *informative across the full dimensionality of the action space*.

D Multi-Step Returns as Information Injection

Multi-step returns are conventionally understood through the bias-variance lens: longer returns reduce bootstrap bias at the cost of increased reward variance. This appendix presents a deeper interpretation rooted in the feedback-loop framework of Appendix B. We argue that multi-step returns are not primarily a bias-variance mechanism but an *information injection* that structurally changes the Bellman update from self-referential recursion to partially grounded supervision. Throughout this appendix, $a_{t+k} = \pi_\phi(s_{t+k})$ as defined in Section 3.1.

D.1 Bootstrap as Self-Referential Recursion

The one-step Bellman target $y = r + \gamma Q_{\bar{\theta}}(s', a')$ contains exactly one external observation: the scalar reward r. Everything else—the value estimate $Q_{\bar{\theta}}$, evaluated at a state-action pair (s', a') that itself depends on the learned policy—is the critic's own output, recycled as supervision.

The information content of this target can be decomposed as:

$$y^{(1)} = \underbrace{r_t}_{\text{1 external observation}} + \underbrace{\gamma Q_{\bar{\theta}}(s_{t+1}, a_{t+1})}_{\text{recursive self-estimate}}. \tag{30}$$

The critic is being trained on a signal that is predominantly its own prediction. In the language of control theory, this is a feedback system with minimal external input—the loop is nearly closed, and whatever noise exists in the critic's estimate is recirculated with each update.

D.2 Multi-Step Returns: Replacing Recursion with Observation

The n-step target replaces $n-1$ rounds of recursive self-evaluation with direct environmental observation:

$$y^{(n)} = \underbrace{\sum_{k=0}^{n-1} \gamma^k r_{t+k}}_{n \text{ external observations}} + \underbrace{\gamma^n Q_{\bar{\theta}}(s_{t+n}, a_{t+n})}_{\text{recursive self-estimate}}. \tag{31}$$

The critical insight is that the first n reward terms are directly observed environmental quantities and therefore contain no critic approximation error. Learned value estimation enters only at step n, through the bootstrap term $\gamma^n Q_{\bar{\theta}}(s_{t+n}, a_{t+n})$.

Proposition D.1 (Recursion-depth reduction)**.** *In the one-step Bellman target, every step of value propagation beyond the immediate reward involves critic estimation error. In the n-step target,*

$$y_t^{(n)} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n Q_{\bar{\theta}}(s_{t+n}, \pi_\phi(s_{t+n})), \tag{32}$$

the cumulative reward term is directly observed and introduces no estimation error. Bootstrap error enters only through the final term and is attenuated by γ^n rather than γ.

This is a structural change to the learning problem, not a quantitative adjustment. The critic transitions from training on a signal that is almost entirely self-referential to training on a signal that is partially grounded in observed reality. In the feedback-loop framework, this reduces the signal gain G_{sig} by diluting the self-referential component of the target with external observations (Appendix B, Proposition B.1).

D.3 Segment-Level Supervision

The n reward terms in the multi-step target are not arbitrary observations; they form a contiguous trajectory segment $[t, t+n)$ that captures the temporal coherence of the task dynamics.

One-step TD provides *point supervision*: the critic learns how good a single instant is. Multi-step TD provides *segment supervision*: the critic learns how good a short trajectory is.

For coordination tasks such as quadruped locomotion or dexterous manipulation, where value arises from sustained sequences of correct actions rather than single-step decisions, segment-level supervision is qualitatively more informative. A single reward r_t tells the critic whether this instant of the gait cycle was good; the three-step sum $r_t + \gamma r_{t+1} + \gamma^2 r_{t+2}$ tells the critic whether this phase of the gait cycle was good, providing a much stronger learning signal for tasks whose reward structure is inherently sequential.

D.4 Multi-Step Returns as Minimal Imagination

Model-based methods such as DreamerV3 extend supervision by imagining trajectories through a learned dynamics model, gaining abundant but approximate signal that compounds model error over the rollout horizon. Multi-step returns provide a similar effect using *real* rewards with zero approximation error, but are limited to transitions in the replay buffer. In this sense, multi-step returns are the minimal model-free analogue of imagination. Our results suggest that up to 61 action dimensions, this minimal supervision, combined with gain control (LayerNorm) and error limiting (Huber loss), suffices to match model-based imagination.

D.5 Why High Dimensions Demand Multi-Step Supervision

In low-dimensional tasks, one-step bootstrap provides adequate signal. In high-dimensional tasks, the critic is inaccurate over most of the action space: the bootstrap term becomes unreliable recirculation of its own noise, converging to an uninformative fixed point. Multi-step returns break this degeneracy by injecting n observed rewards independent of the critic, providing an external anchor that prevents the loop from closing on itself.

D.6 Choice of $n = 3$

We use $n = 3$ across all experiments without per-task tuning. This choice is consistent with prior work: MRQ (Fujimoto et al., 2025) and DrQ-v2 (Yarats et al., 2022) also adopt 3-step returns. The choice reflects a structural trade-off. Increasing n improves signal quality by injecting more environmental observations (reducing G_{sig} approximately linearly per Proposition B.1), but introduces two costs. First, the reward-sum variance grows with n, widening the target distribution, which requires Huber loss to prevent geometric error amplification (Appendix B). Second, larger n increases episode-boundary truncation: when episodes terminate early, fewer than n rewards are collected, reducing effective information per sample. In environments with frequent resets, large n degrades sample utilization.

At $n = 3$, the signal gain is reduced by $\sim 3\times$ relative to one-step bootstrap, episode-boundary effects are negligible for 1000-step episodes in DeepMind Control, and reward variance remains manageable under Huber loss with $\beta = 0.3$.

Why not mix multiple step lengths? Averaging targets of different horizons (e.g. $\frac{1}{2}(y^{(1)} + y^{(3)})$) does not reduce variance, as all bootstrap terms query the same critic $Q_{\bar{\theta}}$. The dominant noise source is shared, producing near-unit correlation; mixing correlated targets cannot reduce variance. Only structurally independent information, e.g. larger n or independent critics (ensembles), can dilute bootstrap noise. We do not claim $n = 3$ is optimal, but it lies within the effective range and requires no task-specific tuning.

E Huber Loss as Implicit Trust Region Protection

The choice of loss function for temporal-difference learning has deeper implications in high-dimensional continuous control than is commonly recognized. We argue that the transition from MSE to Huber loss is not merely a robustness trick, but a necessary structural adaptation to the geometry of high-dimensional state-action spaces.

Why MSE becomes catastrophic in high dimensions. In low-dimensional tasks, large TD errors predominantly reflect genuine value structure, including discontinuities between stable and unstable behaviors, reward boundaries, and contact events, and MSE's quadratic penalty appropriately prioritizes their correction. As the action dimensionality grows, this assumption breaks down due to the *volume effect* of high-dimensional geometry. Two action vectors differing by a small amount σ per component have expected Euclidean distance $\mathbb{E}\|\Delta a\|_2 = \sigma\sqrt{d_a}$. At $d_a = 6$ (Walker) and $\sigma = 0.1$, this gives $\|\Delta a\|_2 \approx 0.24$—a small perturbation within the critic's interpolation range. At $d_a = 38$ (Dog), the same per-component difference yields $\|\Delta a\|_2 \approx 0.62$; at $d_a = 61$ (H1Hand), $\|\Delta a\|_2 \approx 0.79$. The dominant source of large TD errors shifts from genuine value discontinuities to OOD extrapolation artifacts. Under MSE, these artifacts receive *quadratic* gradient magnitudes, creating a destructive feedback loop: the critic over-corrects on spurious errors, distorting the value landscape in well-learned regions, which generates new spurious errors at the next update.

Huber loss as implicit trust region. The Huber loss transitions from quadratic to linear penalty at a threshold β:

$$\ell_\beta(\delta) = \begin{cases} \dfrac{\delta^2}{2\beta}, & |\delta| \le \beta, \\ |\delta| - \dfrac{\beta}{2}, & |\delta| > \beta. \end{cases} \tag{33}$$

In the linear regime ($|\delta| > \beta$), the gradient magnitude is capped at unity regardless of the TD error's size. Small errors (likely genuine signals within the data support) receive full quadratic correction; large errors (likely OOD artifacts) are down-weighted to linear, preventing them from overwriting the critic's learned structure. The mechanism is analogous to PPO's ratio clipping (Schulman et al., 2017): both limit single-step updates to protect the stability of a self-referential learning system. The formal consequence, namely geometric error growth under MSE versus arithmetic growth under Huber, is established in Proposition B.2.

Interaction with N-step returns. The choice of β interacts with the N-step return horizon. Longer N-step returns accumulate both real rewards and bootstrap variance; if β is too small relative to the N-step reward scale, even genuine multi-step signals are clipped. Conversely, if β is too large, OOD protection is insufficient. In our experiments, $\beta = 0.3$ provides effective OOD suppression while preserving the signal gain from $n = 3$ step returns, and we find this value to be robust across all tested environments without per-task tuning.

F Prior Methods as Partial Gain Compression

Several recent methods in continuous reinforcement learning introduce techniques that, under our gain compression framework, address subsets of the stability challenges we identify. These techniques are motivated by different intuitions, including normalization for training stability, robust losses for outlier handling, and multi-step returns for bias-variance trade-off, without a unifying diagnostic. We briefly classify each as gain compression at a specific stage of the Bellman loop.

Critic-side normalization (CrossQ, SimBaV2): gradient gain compression. CrossQ (Bhatt et al., 2024) applies Batch Normalization within the Q-network; SimBaV2 (Lee et al., 2025) uses hyperspherical normalization to constrain weights and feature representations. Both compress G_{grad} by bounding the critic's Jacobian spectral norm (the same mechanism formalized for LayerNorm in Appendix C). However, neither addresses G_{sig} (insufficient signal from one-step bootstrap) nor G_{err} (unchecked error amplification under MSE), leaving two of three instability channels open.

Huber loss in MRQ: error gain compression. MRQ (Fujimoto et al., 2025) adopts Huber loss for critic training, compressing G_{err} and converting geometric error growth to arithmetic (Proposition B.2). However, MRQ does not provide an explicit analysis of *why* Huber outperforms MSE specifically in high-dimensional action spaces, and the connection to OOD extrapolation geometry (Appendix E) is left implicit. Without simultaneous compression of G_{grad} and G_{sig}, the error gain reduction alone is insufficient: a stable optimization of a corrupted, low-signal objective.

N-step returns in DrQ-v2: signal gain compression. DrQ-v2 (Yarats et al., 2022) employs multi-step returns, framed through the classical bias-variance lens. As argued in Appendix D, the deeper role is *information injection*: n-step returns reduce G_{sig} by replacing self-referential recursion with environmental observation (Proposition B.1). Without G_{grad} and G_{err} compression, the richer signal is corrupted by an ill-conditioned landscape and amplified by unchecked error propagation.

TD3 policy smoothing and SAC entropy: implicit but insufficient OOD mitigation. TD3's target policy smoothing (Fujimoto et al., 2018) and SAC's entropy regularization (Haarnoja et al., 2018) both replace point-wise Q-evaluation with local action-space averages, providing partial OOD protection. However, as analyzed in Appendix B.7, both mechanisms degrade in high dimensions: the smoothing radius grows as $\sigma\sqrt{d_a}$, transforming local averaging into non-local OOD evaluation. Neither places any direct constraint on G_{err} or G_{sig}, as they smooth the query distribution without controlling the critic's response to the resulting errors.

The case for simultaneous compression. Each method compresses gain at one stage of the Bellman loop. Our ablations (Section 4.3) show that *simultaneous* compression of all three, G_{err} (Huber), G_{grad} (LayerNorm-Q), and G_{sig} (N-step returns), yields performance that exceeds any individual or pairwise combination. The gains interact multiplicatively (Eq. 19); compressing only one or two leaves a pathway for instability to propagate.

G Additional Empirical Analysis

This appendix provides supplementary empirical evidence supporting the feedback-loop analysis developed in the main text. We present four sets of analyses: critic loss dynamics across ablated configurations (Section G.1), Q-value magnitude and stability (Section G.2), cross-algorithm validation demonstrating that the three-component repair generalizes beyond TD3 (Section G.3), and robustness to critic network width (Section G.4). The diagnostic curves in Sections G.1–G.2 are recorded on Dog-Run (38-dimensional actions) over 3 seeds.

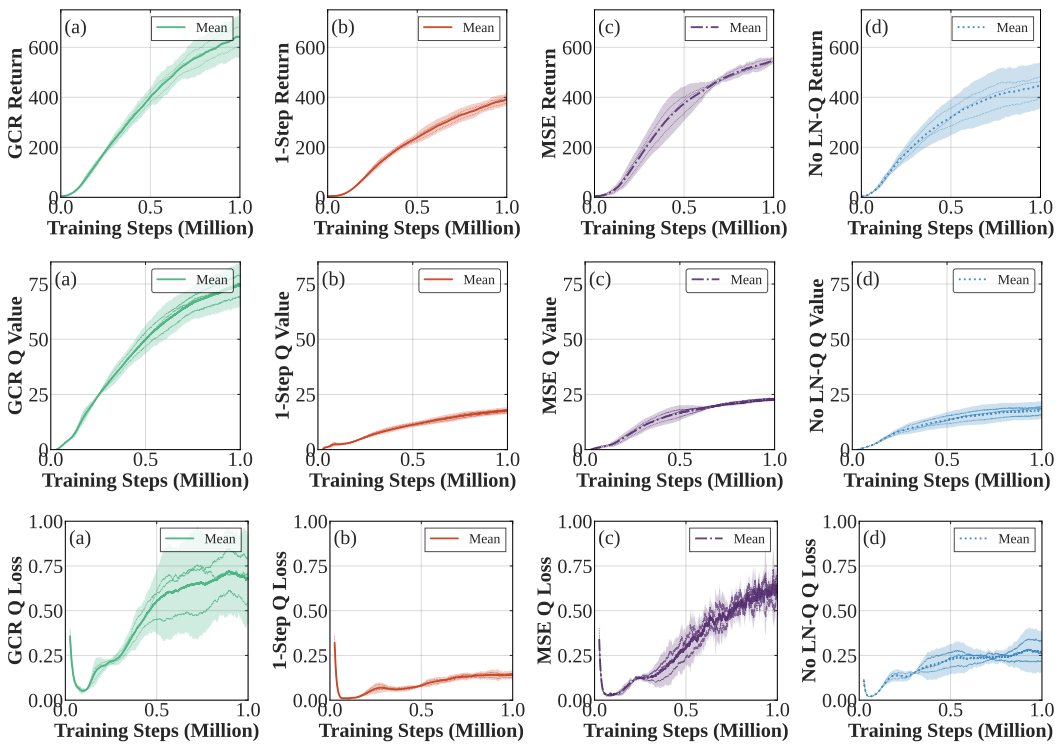

Figure G.1: **Training diagnostics on Dog-Run (38-D actions).** Each column corresponds to one configuration: (a) GCR (full), (b) − n-step, (c) − Huber (MSE), (d) − LN-Q. **Top:** episodic return. **Middle:** mean Q-value. **Bottom:** critic loss. GCR achieves high reward, high Q-value, and high loss reflecting active learning of a complex value landscape. Removing Huber (c) produces high loss with high volatility but low Q-value, indicating gradient budget consumed by OOD error correction. Removing n-step (b) or LN-Q (d) produces low loss and low Q-value, indicating convergence to an uninformative fixed point. Mean over 3 seeds; shaded regions show runs.

G.1 Critic Loss Dynamics

The bottom row of Figure G.1 compares the critic training loss across all four configurations.

Key observations. *GCR has the highest loss—because it is learning.* The full configuration exhibits critic loss rising steadily to 0.5–0.8 over training, with substantial variance across seeds. This is not a sign of instability but a consequence of the critic tracking a rapidly growing value landscape: as the agent's reward climbs from 0 to 800+, the critic must continuously revise its predictions upward, generating large TD errors that reflect genuine learning rather than noise.

MSE loss shows high loss with high volatility but low Q-value. The − Huber (MSE) configuration exhibits loss in the 0.5–0.75 range with pronounced spikes and volatility, yet its Q-value plateaus at only ∼20 (compared to GCR's ∼75). This is the signature of self-induced instability: the critic's gradient budget is consumed by correcting OOD damage rather than learning new value structure. Loss is high not because the critic is learning aggressively, but because it is repeatedly repairing landscape corruption caused by MSE's unchecked error amplification.

Removing n-step or LN-Q produces low loss and low Q-value. The − n-step and − LN-Q configurations both exhibit low, stable critic loss (0.1–0.25) throughout training. This apparent stability is misleading: it reflects convergence to an uninformative fixed point where the critic has little to learn, not successful optimization of a meaningful value function. The corresponding Q-values confirm this interpretation, reaching only ∼15–25.

Summary. The four configurations illustrate three qualitatively distinct regimes: active learning (GCR: high loss, high Q-value), destructive cycling (MSE: high loss, low Q-value), and stagnation (no

n-step / no LN-Q: low loss, low Q-value). These regimes align precisely with the three failure modes predicted by the stability analysis in Appendix B.

G.2 Q-Value Magnitude and Stability

The middle row of Figure G.1 compares the mean predicted Q-value across the four configurations.

Key observations. *GCR achieves Q-values $4\times$ higher than all ablations.* The full configuration's mean Q-value increases monotonically to $\sim$75, while ablations plateau between 15 and 25. Since Q-values estimate discounted cumulative reward, this $4\times$ gap reflects the performance difference: GCR attains the strongest overall return, while the ablated configurations reach lower performance, with the no-Huber variant remaining closest to GCR.

Gain compression does not suppress learning. A natural concern is that Huber loss and LayerNorm, by compressing the feedback loop's gain, might also suppress the critic's ability to learn high Q-values. The data refute this concern decisively: the gain-compressed configuration achieves Q-values *far higher* than any unconstrained variant. Gain compression removes destructive noise without attenuating constructive signal, so the critic learns more effectively because it is not spending gradient budget correcting self-inflicted damage.

Q-value evolves smoothly with learning. GCR's Q-value curve exhibits smooth, monotonic growth alongside the reward curve (top row of Figure G.1), indicating that the critic continues to increase its value estimates as performance improves. In contrast, the MSE configuration's Q-value grows initially but saturates while the critic loss remains high, suggesting that continued critic updates do not translate into comparable growth in its value estimates.

G.3 Cross-Algorithm Validation: SAC with GCR

The feedback-loop analysis in Section 3 identifies the instability as a property of the Bellman bootstrap, not of any specific actor-critic algorithm. If this diagnosis is correct, the same three-component repair should generalize to other off-policy methods.

We validate this prediction by applying the identical three modifications (LayerNorm on critic, 3-step returns, Huber $\beta = 0.3$) to SAC, with no additional tuning. As shown in Section 4, SAC-GCR achieves 700.2 on Dog-Run ($d_a = 38$), surpassing TD-MPC2 by $4.5\times$ and vanilla SAC by $59\times$, and reaches 735.0 on the 61-dimensional H1Hand-Walk, where all baselines remain below 180.

Entropy interaction. SAC's entropy regularization provides distributional smoothing of the policy (broadening the action sampling distribution), but as discussed in Appendix B, it does not address the geometric properties of the critic's value landscape. The three-component repair is therefore complementary to entropy regularization: SAC's entropy bonus smooths the policy distribution, while LayerNorm conditions the critic's Jacobian, Huber loss limits error amplification, and multi-step returns inject environmental signal, mechanisms that entropy cannot provide.

Implication. The generalization from TD3 to SAC supports the central claim of this paper: the failure of model-free RL in high dimensions is not algorithm-specific but structural. Any off-policy actor-critic method using the Bellman bootstrap is subject to the same three-gain instability, and the same three-component repair restores functionality regardless of the specific framework.

G.4 Robustness to Network Width

We additionally vary the critic hidden width on Dog-Run while keeping all other settings fixed. Performance remains comparable from width 256 to 1024, with returns of 791.0, 811.5, and 814.0, respectively. This suggests that the effectiveness of GCR is not specific to the default 256-unit critic and that increasing model capacity is not required to obtain strong performance in this setting.

Table G.1: Robustness to critic hidden width on Dog-Run at 2M steps. Mean ± 95% CI over 3 seeds.

Hidden width	Return
256	791.0 ± 28.2
512	811.5 ± 145.1
1024	814.0 ± 28.3

H SUPPLEMENTARY RESULTS ON HUMANOIDBENCH-H1

In addition to the main benchmark results, we also provide results on the `HumanoidBench-H1` suite as a supplementary evaluation. This benchmark involves 19-dimensional actions and covers a diverse set of coordinated humanoid control behaviors. We also compare against recent model-free methods with architectural innovations whose implementations and results are available on the H1 suite but not on the H1Hand suite used in the main evaluation.

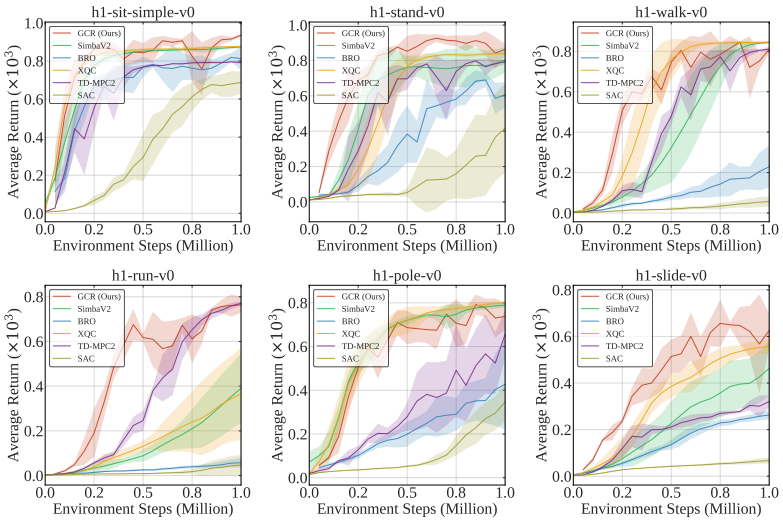

Figure H.1: **Learning curves on 6 high-dimensional continuous-control tasks.**

Table H.1: **Total Average Return on H1 tasks.** Mean ± 95% confidence interval.

Task	SAC	TD-MPC2	BRO	SimbaV2	XQC	GCR (Ours)
H1-Sit	686.9 ± 58.1	792.5 ± 8.8	811.9 ± 61.1	873.5 ± 7.1	875.5 ± 5.9	**935.3 ± 5.3**
H1-Stand	417.7 ± 252.6	792.0 ± 12.9	606.1 ± 71.8	801.6 ± 70.4	838.7 ± 4.2	**864.3 ± 31.1**
H1-Walk	56.3 ± 25.5	810.6 ± 3.0	229.1 ± 103.9	843.4 ± 6.2	**844.5 ± 7.8**	809.3 ± 19.3
H1-Run	47.1 ± 44.6	**770.8 ± 9.1**	57.2 ± 21.8	390.7 ± 155.6	365.4 ± 206.4	764.0 ± 35.4
H1-Pole	346.8 ± 85.5	656.3 ± 112.1	427.5 ± 207.3	789.3 ± 9.2	**800.5 ± 7.3**	737.9 ± 59.9
H1-Slide	66.5 ± 12.6	320.8 ± 25.0	262.8 ± 19.2	463.1 ± 117.5	557.3 ± 31.1	**625.3 ± 71.3**