From Anti-Forgetting to Fast Adaptation: Continual Reinforcement Learning with World Models

Qiyang Zhou[1,4*] **Yuliang Cai**[1*] **Peng Wang**[1] **Senquan Yi**[2]
Yanming Li[1] **Haobo Fu**[3] **Li Shen**[1,4†]

[1]Shenzhen Campus of Sun Yat-sen University
[2]Wuhan University
[3]Tencent AI Platform Department
[4]Shenzhen Loop Area Institute

Abstract

Continual Reinforcement Learning (CRL) requires agents to adapt to sequentially arriving tasks while retaining performance on previous ones. Most existing CRL methods rely on model-free frameworks, addressing catastrophic forgetting via regularization, parameter isolation, or experience replay. World models offer a promising alternative by accumulating transferable dynamics knowledge and enabling planning-based decision-making. However, existing world model approaches suffer from slow adaptation at task boundaries: abrupt reward shifts mislead the planner, fixed planning horizons accumulate prediction errors when the dynamics model is unreliable, and quality-agnostic replay dilutes training while exacerbating forgetting. We propose HERD (Horizon-adaptive Elite Replay with reward Disagreement), which addresses these challenges from two complementary perspectives: for ***how to learn***, Uncertainty-aware Reward Planning leverages ensemble disagreement to accelerate reward calibration and guide exploration, while Adaptive Planning Horizon adjusts rollout depth based on dynamics prediction error to prevent error accumulation; for ***what to learn***, Elite Experience Replay applies quality-stratified reweighting of historical trajectories to strengthen knowledge retention. Together, these components ***accelerate new-task adaptation and mitigate catastrophic forgetting.*** Experiments on Continual World, Continual Bench, and DM Control show that HERD consistently outperforms existing methods in overall performance, forgetting mitigation, and new-task adaptation.

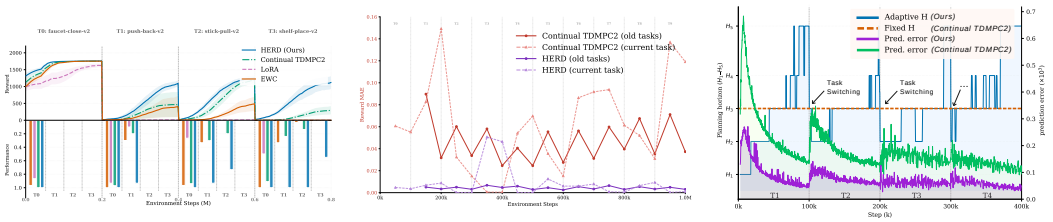
1 Introduction

Continual Reinforcement Learning (CRL) requires agents to adapt to sequentially arriving tasks without catastrophic forgetting of previously learned ones [1; 2], posing the stability-plasticity dilemma [3]. Most existing methods are model-free, mitigating forgetting through regularization [4; 5], parameter isolation [6; 7], or experience replay [8; 9], but operate primarily on policies or value functions, making limited use of shared environmental structure across tasks. World models, which learn latent dynamics, offer a stronger foundation for cross-task knowledge reuse [10; 11; 12], and recent work has shown that they can achieve forward transfer in continual settings [13; 14; 15].

However, a shared dynamics model alone is insufficient for consistent improvement in continual settings. As shown in Figure 1a, existing methods suffer from both slow new-task adaptation and severe catastrophic forgetting. We identify two key factors behind the slow adaptation, empirically verified in Figure 1b and 1c. **Reward miscalibration slows adaptation.** After a task switch, reward functions shift abruptly, causing an uncalibrated reward predictor to mislead the planner and produce

*Equal contribution.
†Corresponding author: shenli6@mail.sysu.edu.cn.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) Performance & Forgetting (b) Reward Calibration (c) Dynamics Error & Horizon

Figure 1: Empirical verification of the challenges in continual world models. **(a)** Existing methods suffer from slow adaptation and severe catastrophic forgetting. **(b)** Task switching causes sharp spikes in reward prediction error, which misleads the planner if not calibrated rapidly. **(c)** Dynamics prediction error spikes at task boundaries; fixed horizon (orange dashed) accumulates errors and does not make full use of long horizon when stable, highlighting the need of adaptive horizon.

low-quality rollouts that further delay adaptation. As shown in Figure 1b, reward prediction error spikes sharply at every task boundary and recovers slowly. **Fixed planning horizons accumulate errors.** A fixed rollout depth ignores model reliability: at task boundaries, elevated dynamics prediction error compounds over the horizon (Figure 1c), while after stabilization, the same short horizon under-utilizes the model's long-range capacity. Moreover, quality-agnostic replay further compounds the forgetting problem: standard replay treats all historical trajectories equally, allowing low-quality experiences to interfere with retained knowledge and degrade overall performance.

These observations reveal that the core challenges in continual world model learning go beyond forgetting mitigation: ***the agent must also learn to adapt efficiently***. We frame these challenges along two complementary dimensions, ***how to learn*** (the quality of the planning process) and ***what to learn*** (the selection of training experiences), and propose HERD (Horizon-adaptive Elite Replay with reward Disagreement) to address both. On the ***how to learn*** side, we propose two mechanisms that jointly govern planning quality. First, Uncertainty-aware Reward Planning (URP) measures prediction disagreement among an ensemble of reward predictors to identify under-calibrated regions. During training, this signal upweights high-uncertainty samples and combined with periodic reward parameter resets, accelerates reward calibration; during planning, it serves as an optimistic exploration bonus to guide the agent toward more informative data. Second, Adaptive Planning Horizon adjusts rollout depth based on multi-step dynamics prediction error, reducing error accumulation at task boundaries and recovering long-horizon capacity once the model stabilizes. Together, these two mechanisms ensure that the agent plans with calibrated rewards and appropriate horizons, enabling efficient adaptation from the onset of each new task. On the ***what to learn*** side, Elite Experience Replay applies quality-stratified reweighting of old-task episodes, directing training toward historical experiences most conducive to knowledge retention.

We evaluate HERD on Continual World [16], Continual Bench [15], and DM Control [17; 18], covering robotic manipulation and continuous control settings. Results show that HERD consistently outperforms existing methods in average performance, forgetting mitigation, and new-task adaptation. In summary, our contributions are threefold:

- We propose Uncertainty-aware Reward Planning (URP), which leverages ensemble disagreement for reward calibration during training and optimistic exploration during planning, accelerating adaptation after task switching.
- We design an Adaptive Planning Horizon that adjusts rollout depth based on dynamics prediction error, and Elite Experience Replay that applies quality-stratified reweighting of old-task episodes to strengthen knowledge retention.
- We conduct systematic experiments on three benchmarks, demonstrating consistent improvements over existing methods, with ablations validating each component.

2 Related Work

2.1 Continual Reinforcement Learning

Continual Reinforcement Learning (CRL) studies how agents learn across a sequence of tasks while balancing plasticity and stability [1; 2; 3]. Existing methods address catastrophic forgetting through regularization [4; 5], parameter isolation [6; 7], or experience replay [8; 9]. Beyond forgetting, recent

work identifies task interference, negative transfer [19; 20], and forward transfer [16; 21; 22] as additional challenges, framing CRL as a problem of knowledge selection and reuse[23]. Recent methods extend traditional paradigms accordingly: MT-Core [24] decouples high-level task planning from low-level policy learning via a multi-granularity transfer framework; CSP [25] builds policy subspaces for scalable knowledge reuse; DGR [26], t-DGR [27], and DISTR [28] adopt generative replay to improve the stability-plasticity balance without storing raw data.

2.2 World model for RL

World model RL builds reusable internal predictive models from environment transitions, rewards, or latent dynamics, enabling planning, policy optimization, or imagination-based training. Dyna [29] unifies interaction, model learning, and planning; World Models [10] further enables policy learning in imagined environments via compressed latent representations. Existing approaches fall into two categories. The first treats the world model as a generative simulator: World Models [10], SimPLe [30], PlaNet [31], and the Dreamer family [32; 33; 11] improve sample efficiency via latent imagination and video-based prediction. The second integrates model-based planning with value learning: TD-MPC [34] combines receding-horizon MPC with long-term TD value learning in a task-oriented latent dynamics model, and TD-MPC2 [12] further scales this framework to cross-task, cross-domain continuous control. Compared to generative world models that require explicit pixel reconstruction, the control-oriented latent dynamics of TD-MPC/TD-MPC2 more directly serve decision-making and provide the foundation for this work.

2.3 World model for CRL

World models provide CRL with transferable latent dynamics and reusable planning structures across tasks. Early methods such as HyperCRL [35] use task-conditioned hypernetworks to model non-stationary dynamics, but require explicit task boundary information. Continual-Dreamer [13] and Online Agent [15] both adopt task-agnostic settings, combining replay and shared dynamics for lifelong planning. Subsequent work improves knowledge retention through enhanced replay in WMAR [36] and uncertainty-filtered replay in UBER [37], while DRAGO [14] and DISTR [28] adopt generative replay via synthesized trajectories without storing raw data. While world models have proven effective for mitigating forgetting and enabling knowledge reuse in CRL, existing methods remain susceptible to reward shifts at task boundaries and provide limited guidance during early-stage exploration. In contrast, HERD addresses the planning process itself by calibrating reward predictions under uncertainty and adapting the planning horizon to model reliability, complementing replay-based retention with mechanisms that accelerate new-task adaptation.

3 Preliminaries

3.1 Continual Reinforcement Learning

Reinforcement learning aims to learn a policy $\pi(\mathbf{a}|\mathbf{s})$ that maximizes the expected discounted return $\mathbb{E}\pi[\sum t = 0^{\infty}\gamma^t R(\mathbf{s}_t, \mathbf{a}_t)]$ in a Markov Decision Process (MDP). An MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma, d_0)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition dynamics, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\gamma \in [0, 1)$ is the discount factor, and d_0 is the initial state distribution [38]. In CRL [1; 2], an agent faces a sequence of tasks $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K$ presented sequentially. We adopt the task-incremental setting [4; 39], where each task $\mathcal{T}_k$ is defined as an MDP $\langle \mathcal{S}, \mathcal{A}, P_k, R_k, \gamma, d_0^{(k)} \rangle$. The state and action spaces $\mathcal{S}$ and $\mathcal{A}$ are shared across all tasks, while the transition dynamics P_k and reward function R_k vary per task. During training on task $\mathcal{T}_k$, the agent collects data through online interaction with the current environment and has no access to previous task environments. The core challenge of CRL lies in the stability-plasticity tradeoff [1; 40]: agents must adapt to new tasks without degrading performance on previously learned ones, while interacting with only one task at a time [15].

3.2 Latent Space World Models

Model-based reinforcement learning augments decision-making by learning a predictive model of the environment [41]. In the latent space world model framework [10; 12; 33], an agent maintains a set of learnable components: an encoder h_θ maps observations $\mathbf{o}$ to latent states $\mathbf{z} = h_\theta(\mathbf{o})$; a latent dynamics model d_θ predicts the next latent state $\hat{\mathbf{z}}' = d_\theta(\mathbf{z}, \mathbf{a})$; a reward predictor R_θ and a value function V_θ estimate immediate rewards and long-term returns, respectively; and a policy π_θ

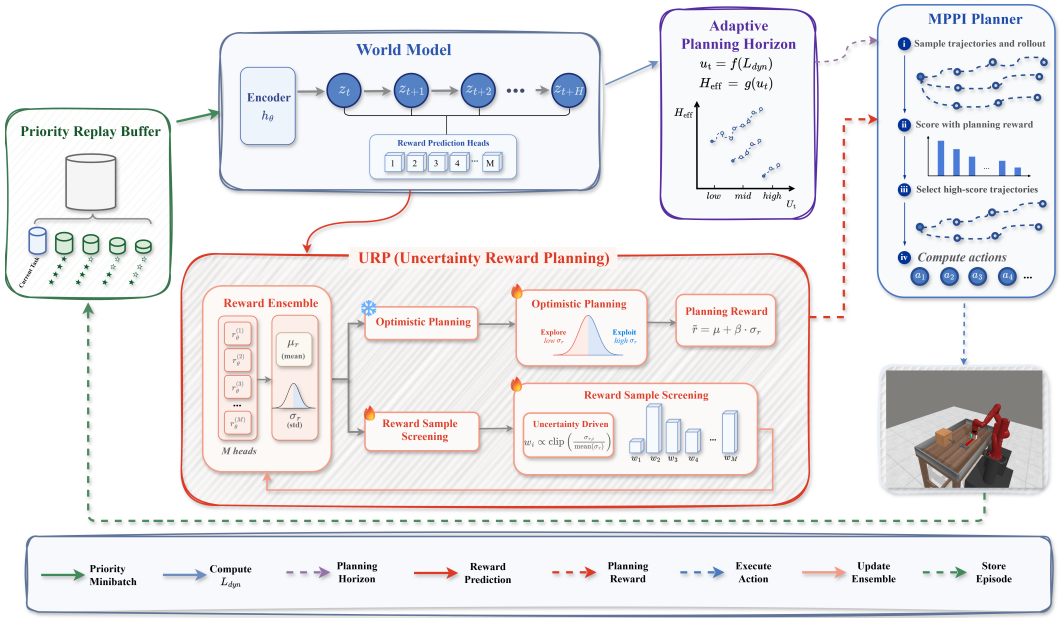


Figure 2: Overall architecture of HERD. By integrating URP and Adaptive Planning Horizon, HERD guides the agent on how to learn efficiently during task transitions, while Elite Experience Replay dictates what critical knowledge to retain.

outputs action distributions. During planning, the agent performs multi-step rollouts in latent space to evaluate candidate action sequences. Given the current latent state $\mathbf{z}_0$ and a candidate action sequence $\mathbf{a}_{0:H-1}$, the planner iteratively applies the dynamics model to generate a predicted trajectory $\{\mathbf{z}_h\}_{h=0}^{H}$ and estimates its cumulative value:

$$\hat{V}(\mathbf{z}_0, \mathbf{a}_{0:H-1}) = \sum_{h=0}^{H-1} \gamma^h R_\theta(\mathbf{z}_h, \mathbf{a}_h) + \gamma^H V_\theta(\mathbf{z}_H)$$

where H is the planning horizon. At each timestep, Model Predictive Control (MPC) repeats this process by sampling and optimizing candidate action sequences, executing the first action of the best sequence, and replanning upon observing the new state [15; 42].

4 Method

Building on the analysis in Section 1, we propose Horizon-adaptive Elite Replay with reward Disagreement (HERD), which addresses ***how to learn*** through uncertainty-aware reward planning and adaptive horizon scheduling, and ***what to learn*** through quality-stratified elite experience replay.

4.1 Uncertainty-aware Reward Planning

To accelerate reward-structure adaptation following task switches, we propose Uncertainty-aware Reward Planning (URP). URP first quantifies reward prediction uncertainty via an ensemble of M independent reward predictors, then exploits this uncertainty signal at two levels: ***training-time calibration*** and ***planning-time exploration***.

Reward Uncertainty Estimation. URP quantifies reward prediction uncertainty via an ensemble of M independent reward predictors. Each predictor $f_{\text{rew}}^{(m)}$ maps the latent state z and action a to a scalar reward estimate $\hat{r}_m(z, a)$:

$$\hat{r}_m(z,a) = g^{-1}\left(\sum_{b=1}^{B} \text{softmax}\left(f_{\text{rew}}^{(m)}([z;a])\right)_b \cdot c_b\right), \quad m = 1, \ldots, M. \tag{1}$$

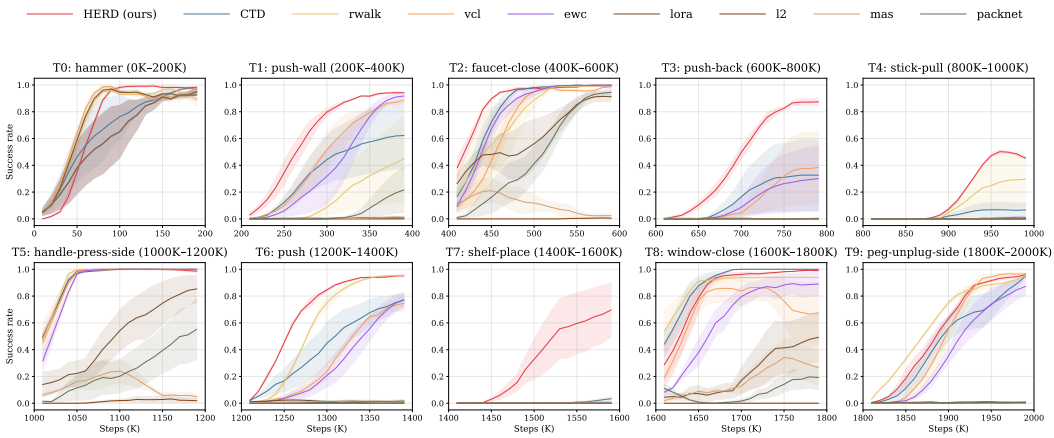


Figure 3: Per-task success rate curves on CW10. HERD (red) adapts rapidly at each task boundary and maintains high success rates on previously learned tasks, while most baselines either adapt slowly or suffer performance degradation on earlier tasks.All results are averaged over 3 random seeds.

All predictors are trained on the same supervision signal. We define reward uncertainty as the standard deviation across predictor outputs:

$$\sigma_r(z,a) = \sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(\hat{r}_m(z,a) - \bar{r}(z,a)\right)^2}, \quad \bar{r}(z,a) = \frac{1}{M}\sum_{m=1}^{M}\hat{r}_m(z,a). \tag{2}$$

High agreement among predictors indicates that the reward structure for a given sample has been sufficiently learned; high disagreement indicates that the model's estimate remains unreliable.

Training:Reward Model Calibration Regions with high σ_r indicate insufficient reward model coverage; we therefore derive adaptive sample weights from σ_r to focus reward learning on these regions. For sample $(z_h^{(i)}, a_h^{(i)})$ at rollout step h, the weight is defined as:

$$\tilde{w}_h^{(i)} = \mathrm{clip}\left(\frac{\sigma_r(z_h^{(i)}, a_h^{(i)})}{\bar{\sigma}_r}, w_{\min}, w_{\max}\right), \qquad w_h^{(i)} = \frac{\tilde{w}_h^{(i)}}{\frac{1}{H|\mathcal{B}|}\sum_{h'=0}^{H-1}\sum_{j\in\mathcal{B}}\tilde{w}_{h'}^{(j)}}. \tag{3}$$

where $\bar{\sigma}_r$ is the batch-averaged uncertainty and $[w_{\min}, w_{\max}]$ are clipping bounds. The weighted reward learning objective is:

$$\mathcal{L}_{\mathrm{rew}}^{\mathrm{URP}} = \frac{1}{H}\sum_{h=0}^{H-1}\rho^h \cdot \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}} w_h^{(i)} \cdot \ell_{\mathrm{CE}}\left(\hat{r}_h^{(i)}, r_h^{(i)}\right). \tag{4}$$

where H is the number of rollout steps, ρ is the per-step discount factor, $\mathcal{B}$ is the training batch, and $\hat{r}_h^{(i)}$, $r_h^{(i)}$ are the predicted and ground-truth rewards, respectively. This objective is activated from the second task onward, with weights w applied as stop-gradient coefficients.

Planning:Optimistic Planning Reward model calibration operates solely at training time; a planner relying exclusively on current reward predictions may over-exploit familiar regions, neglecting state-action pairs with high reward uncertainty. We therefore augment the planning-phase reward estimate with σ_r as an optimistic exploration bonus, directing the agent toward regions where the reward model remains uncertain:

$$\tilde{r}(z,a) = \bar{r}(z,a) + \beta \cdot \frac{\sigma_r(z,a)}{\max\left(\mathrm{mean}(|\bar{r}|), \epsilon\right)}. \tag{5}$$

where $\bar{r}$ is the mean reward estimate, β is the exploration coefficient, and ϵ is a small constant for numerical stability. This optimistic reward is used only during planning and does not alter the true rewards returned by the environment.

Table 1: Comparison of continual learning methods on CW10, CB, and DMC Cross. We report mean and standard deviation over three seeds for P, F, and STT. Higher P is better(↑); lower F/STT is better(↓). CTD denotes Continual TD-MPC2.

Method	CW10			CB			DMC Cross		
	P (↑)	F (↓)	STT (↓)	P (↑)	F (↓)	STT (↓)	P (↑)	F (↓)	STT (↓)
L2	$0.12_{\pm 0.08}$	$-0.04_{\pm 0.06}$	$0.93_{\pm 0.01}$	$0.14_{\pm 0.07}$	$0.03_{\pm 0.10}$	$1.25_{\pm 0.11}$	$0.12_{\pm 0.00}$	$0.01_{\pm 0.01}$	$0.86_{\pm 0.03}$
EWC	$0.77_{\pm 0.07}$	$-0.00_{\pm 0.01}$	$0.48_{\pm 0.03}$	$0.80_{\pm 0.03}$	$-0.11_{\pm 0.08}$	$0.48_{\pm 0.04}$	$0.59_{\pm 0.04}$	$-0.02_{\pm 0.04}$	$0.53_{\pm 0.02}$
MAS	$0.17_{\pm 0.03}$	$-0.03_{\pm 0.05}$	$0.69_{\pm 0.03}$	$0.18_{\pm 0.01}$	$0.01_{\pm 0.03}$	$0.98_{\pm 0.15}$	$0.18_{\pm 0.01}$	$-0.01_{\pm 0.01}$	$0.82_{\pm 0.00}$
R-Walk	$0.13_{\pm 0.00}$	$0.72_{\pm 0.04}$	$0.39_{\pm 0.02}$	$0.32_{\pm 0.12}$	$0.40_{\pm 0.04}$	$0.72_{\pm 0.14}$	$0.39_{\pm 0.01}$	$0.25_{\pm 0.01}$	$0.52_{\pm 0.01}$
VCL	$0.14_{\pm 0.02}$	$0.78_{\pm 0.13}$	$0.44_{\pm 0.03}$	$0.42_{\pm 0.06}$	$0.40_{\pm 0.04}$	$0.52_{\pm 0.06}$	$0.33_{\pm 0.00}$	$0.26_{\pm 0.04}$	$0.51_{\pm 0.01}$
LoRA	$0.25_{\pm 0.07}$	$0.15_{\pm 0.07}$	$0.67_{\pm 0.03}$	$0.22_{\pm 0.09}$	$0.36_{\pm 0.16}$	$0.77_{\pm 0.10}$	$0.10_{\pm 0.01}$	$0.05_{\pm 0.01}$	$0.73_{\pm 0.02}$
PackNet	$0.37_{\pm 0.08}$	$-0.03_{\pm 0.02}$	$0.75_{\pm 0.04}$	$0.49_{\pm 0.18}$	$-0.16_{\pm 0.24}$	$0.71_{\pm 0.09}$	$0.22_{\pm 0.09}$	$0.01_{\pm 0.14}$	$0.73_{\pm 0.02}$
CTD	$0.60_{\pm 0.07}$	$0.07_{\pm 0.07}$	$0.44_{\pm 0.09}$	$0.46_{\pm 0.03}$	$0.34_{\pm 0.17}$	$0.57_{\pm 0.06}$	$0.44_{\pm 0.03}$	$0.14_{\pm 0.03}$	$0.55_{\pm 0.04}$
HERD	$\mathbf{0.95_{\pm 0.02}}$	$\mathbf{-0.06_{\pm 0.01}}$	$\mathbf{0.26_{\pm 0.03}}$	$\mathbf{0.83_{\pm 0.00}}$	$\mathbf{-0.29_{\pm 0.10}}$	$\mathbf{0.42_{\pm 0.01}}$	$\mathbf{0.62_{\pm 0.02}}$	$\mathbf{-0.02_{\pm 0.01}}$	$\mathbf{0.50_{\pm 0.01}}$

4.2 Adaptive Planning Horizon

While URP improves single-step reward estimation, a fixed planning horizon H cannot adapt to the varying reliability of the world model after task switches, the horizon should shorten when prediction errors accumulate and extend as the model adapts. We therefore propose an Adaptive Horizon mechanism that dynamically adjusts the rollout depth based on the world model's prediction quality:

$$\mathcal{L}_{\text{dyn}}^{H_{\max}} = \sum_{h=1}^{H_{\max}} \|\hat{z}_{t+h} - \text{sg}(z_{t+h})\|_2^2 . \tag{6}$$

where $H_{\max}$ is the maximum planning horizon, $\hat{z}_{t+h}$ is the predicted latent state h steps ahead, z_{t+h} is the corresponding encoded ground-truth latent state, and $\text{sg}(\cdot)$ denotes stop-gradient, applied to prevent the encoder from being updated through this loss and causing representation drift.

At planning time, we define a reliability signal U_k to guide the choice of rollout depth. It is computed from the dynamics loss $\mathcal{L}_{\text{dyn}}^{H_{\max}}$ evaluated on the replay batch, then smoothed via exponential moving average:

$$m_k = \beta_{\text{ema}} m_{k-1} + (1 - \beta_{\text{ema}}) c_k, \qquad U_k = w c_k + (1 - w) m_k. \tag{7}$$

where c_k is the current dynamics prediction error, m_k is its exponential moving average, β_{ema} controls the smoothing rate, and w balances the current estimate against the historical average.

U_k is mapped to the effective planning horizon $H_{\text{eff}} \in [1, H_{\max}]$ via a monotone threshold table $(\tau_j, H_j)_{j=1}^{J}$, with higher uncertainty yielding shorter rollouts.

4.3 Elite Experience Replay

While URP and Adaptive Horizon address how the agent learns, catastrophic forgetting is further governed by what it learns from, specifically the quality and sampling distribution of replay experience. We therefore propose Elite Experience Replay (EER): each episode is annotated with its cumulative return R and a binary success indicator s, from which quality-stratified sampling weights are derived to concentrate training on more informative prior-task episodes.

$$w_{\text{tier}}(e) = \begin{cases} \alpha_1, & \text{if } s = 1 \text{ and } R \geq Q_3, \\ \alpha_2, & \text{if } Q_2 \leq R < Q_3, \\ \alpha_3, & \text{if } Q_1 \leq R < Q_2, \\ \alpha_4, & \text{if } R < Q_1. \end{cases} \tag{8}$$

where Q_1, Q_2, Q_3 are the 25th, 50th, and 75th percentiles of cumulative returns within each old task, and $\alpha_1 > \alpha_2 > \alpha_3 > \alpha_4$ are the tier weights. The top tier additionally requires task success ($s = 1$) to ensure that only episodes demonstrating correct task completion receive the highest replay priority, instead of high-return attempts that fail to complete the task. For K tasks with datasets $\mathcal{D}_1, \ldots, \mathcal{D}_K$, the sampling probability of episode $e \in \mathcal{D}_{k'}$ is:

$$P(e) = \underbrace{\frac{N_{k'}}{\sum_{j=1}^{K} N_j}}_{\text{inter-task ratio}} \cdot \underbrace{\frac{w(e)}{\sum_{e' \in \mathcal{D}_{k'}} w(e')}}_{\text{intra-task reweighting}}, \quad w(e) = \begin{cases} w_{\text{tier}}(e), & \text{if } k' < k_{\text{new}}, \\ 1, & \text{if } k' = k_{\text{new}}. \end{cases} \tag{9}$$

where $N_{k'}$ is the number of stored steps for task k'.

Table 2: Ablation study of key HERD components on the CW10 benchmark, reporting P, F, and STT. Higher P is better(↑); lower F/STT is better(↓). The results show that each of them plays a complementary role in improving the final performance.

Experiment	URP	AH	Elite	P (↑)	F (↓)	STT (↓)
Full Model	✓	✓	✓	$\mathbf{0.95_{\pm 0.02}}$	$\mathbf{-0.06_{\pm 0.01}}$	$\mathbf{0.26_{\pm 0.03}}$
w/o URP	×	✓	✓	$0.86_{\pm 0.02}$	$-0.05_{\pm 0.01}$	$0.32_{\pm 0.00}$
w/o AH	✓	×	✓	$0.91_{\pm 0.01}$	$-0.05_{\pm 0.02}$	$0.30_{\pm 0.02}$
w/o Elite	✓	✓	×	$0.93_{\pm 0.01}$	$-0.02_{\pm 0.01}$	$0.28_{\pm 0.03}$
w/o URP + AH	×	×	✓	$0.70_{\pm 0.01}$	$-0.10_{\pm 0.01}$	$0.43_{\pm 0.03}$
w/o URP + Elite	×	✓	×	$0.82_{\pm 0.01}$	$-0.01_{\pm 0.02}$	$0.30_{\pm 0.02}$
w/o AH + Elite	✓	×	×	$0.90_{\pm 0.02}$	$-0.03_{\pm 0.02}$	$0.28_{\pm 0.01}$

4.4 Overall Algorithm

HERD is implemented on top of a latent space world model [12]; the full pseudocode is provided in Appendix Algorithm 1. The algorithm consists of two phases.

Training. HERD samples batches $\mathcal{B} \sim P$ from the replay buffer using quality-stratified elite sampling. The world model is unrolled for $H_{\max}$ steps and optimized under the following joint objective:

$$\mathcal{L}_{\mathrm{HERD}}(\theta) = \sum_{h=0}^{H_{\max}-1} \rho^h \cdot \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \Big(\lambda_c \left\| \hat{z}_{t+h+1}^{(i)} - \mathrm{sg}(z_{t+h+1}^{(i)}) \right\|_2^2 + \lambda_r \cdot w_h^{(i)} \cdot \ell_{\mathrm{CE}}(\hat{r}_h^{(i)}, r_h^{(i)}) + \lambda_v \cdot \ell_{\mathrm{CE}}(\hat{V}_h^{(i)}, V_h^{\mathrm{tgt},(i)}) \Big). \tag{10}$$

where $\hat{z}_{t+h+1}^{(i)}$ and $z_{t+h+1}^{(i)}$ are the predicted and encoded ground-truth latent states at step h; $w_h^{(i)}$ is the uncertainty-based reward learning weight; ρ is the per-step discount factor; and $H_{\max}$ serves as the planning horizon upper bound for training long-range dynamics prediction.

Planning. Given the current latent state z_0, HERD samples N candidate action sequences via MPPI and evaluates each under the adaptive horizon H_{eff}:

$$\hat{V}(z_0, \mathbf{a}_{0:H_{\mathrm{eff}}-1}) = \sum_{h=0}^{H_{\mathrm{eff}}-1} \gamma^h \tilde{r}(z_h, a_h) + \gamma^{H_{\mathrm{eff}}} Q(z_{H_{\mathrm{eff}}}, \pi(z_{H_{\mathrm{eff}}})). \tag{11}$$

where H_{eff} is determined dynamically by the multi-step dynamics prediction error, and $\tilde{r}$ is the optimism-augmented reward that encourages exploration of uncertain regions. MPPI selects the highest-valued candidate and executes its first action.

5 Experiment

We evaluate HERD on three benchmarks: Continual World (CW) [16], Continual Bench (CB) [15], and DM Control [17; 18]. Specifically, we use the CW10 task sequence from CW, all tasks from CB, and the DMC Cross sequences from DM Control. Detailed descriptions of each benchmark are provided in Appendix B.

5.1 Baselines

All baselines are implemented on top of TD-MPC2 [12] with the same network architecture and training budget, covering the main forgetting mitigation paradigms in continual learning: **(1) Regularization:** L2, EWC [4], MAS [5] , RWalk [43], VCL [44]. **(2) Parameter Isolation:** PackNet [6], LoRA [45]. We additionally include Continual TD-MPC2 [12], which trains sequentially across tasks without any forgetting mitigation, as the naive world model baseline.

5.2 Evaluation Metrics

Traditional CRL metrics focus primarily on final steady-state performance and zero-shot behavior at task boundaries. To more directly and precisely quantify how quickly an agent adapts to each new

Table 3: Ablation study of task-order robustness on CW10. Order 1 is the default CW10 order, while Orders 2 and 3 are provided in Appendix B.

Method	Order 1			Order 2			Order 3		
	P (↑)	F (↓)	STT (↓)	P (↑)	F (↓)	STT (↓)	P (↑)	F (↓)	STT (↓)
EWC	$0.77_{\pm 0.07}$	$-0.00_{\pm 0.01}$	$0.48_{\pm 0.03}$	$0.59_{\pm 0.06}$	$0.03_{\pm 0.02}$	$0.50_{\pm 0.01}$	$0.59_{\pm 0.03}$	$-0.01_{\pm 0.06}$	$0.56_{\pm 0.04}$
R-Walk	$0.13_{\pm 0.00}$	$0.72_{\pm 0.05}$	$0.39_{\pm 0.02}$	$0.11_{\pm 0.01}$	$0.69_{\pm 0.03}$	$0.43_{\pm 0.01}$	$0.14_{\pm 0.01}$	$0.67_{\pm 0.03}$	$0.44_{\pm 0.00}$
LoRA	$0.25_{\pm 0.07}$	$0.15_{\pm 0.07}$	$0.67_{\pm 0.03}$	$0.28_{\pm 0.12}$	$0.13_{\pm 0.07}$	$0.66_{\pm 0.02}$	$0.15_{\pm 0.03}$	$0.29_{\pm 0.03}$	$0.72_{\pm 0.04}$
CTD	$0.60_{\pm 0.07}$	$0.07_{\pm 0.07}$	$0.44_{\pm 0.09}$	$0.63_{\pm 0.05}$	$0.11_{\pm 0.09}$	$0.40_{\pm 0.04}$	$0.56_{\pm 0.13}$	$0.11_{\pm 0.09}$	$0.47_{\pm 0.04}$
HERD	$\mathbf{0.95_{\pm 0.02}}$	$\mathbf{-0.06_{\pm 0.01}}$	$\mathbf{0.26_{\pm 0.03}}$	$\mathbf{0.94_{\pm 0.05}}$	$\mathbf{-0.10_{\pm 0.02}}$	$\mathbf{0.33_{\pm 0.01}}$	$\mathbf{0.89_{\pm 0.03}}$	$\mathbf{-0.07_{\pm 0.01}}$	$\mathbf{0.34_{\pm 0.02}}$

task, we introduce Steps to Threshold (STT)[46] as a complementary measure of adaptation speed. Let K be the total number of tasks, $p_i(t)$ the success rate on task i at training step t, and δ the number of training steps per task.

Average Performance (AP)[2] measures the mean success rate across all tasks after training completes: $AP = \frac{1}{K}\sum_{i=1}^{K} p_i(K \cdot \delta)$.

Forgetting (F)[16] measures the degree of performance degradation on old tasks: $F = \frac{1}{K}\sum_{i=1}^{K}\big(p_i(i \cdot \delta) - p_i(K \cdot \delta)\big)$. Lower F indicates less forgetting; negative values indicate that subsequent training improves performance on old tasks.

Steps to Threshold (STT) measures the fraction of the per-task training budget required to reach a target success rate θ on each new task: $STT = \frac{1}{K}\sum_{i=1}^{K}\frac{1}{\delta}\min\{\, t \mid \bar{p}_i(t) \geq \theta \,\}$, where $\bar{p}_i(t)$ is the average success rate on task i at step t within its dedicated training window. We set $\theta = 0.5$ throughout all experiments.Lower STT indicates faster adaptation to new tasks.

5.3 Main Results

Table 1 summarizes results across CW10, CB, and DMC Cross. HERD achieves state-of-the-art performance on all metrics across all three benchmarks.

HERD leads all baselines in average performance on all three benchmarks. Regularization methods underperform because they constrain only parameter drift, leaving reward miscalibration at task boundaries unaddressed. Parameter isolation methods avoid interference but sacrifice knowledge sharing. DMC Cross, which features more diverse locomotion morphologies, poses greater dynamics heterogeneity than CW10 and CB, depressing absolute performance for all methods.

HERD achieves negative forgetting on all benchmarks. CB shows the largest negative forgetting, as high inter-task dynamics overlap amplifies positive transfer from elite replay. In contrast, importance-based regularization methods suffer severe forgetting: constraining parameter shifts cannot prevent representation drift when the shared encoder is updated. EWC, the best among them, only passively preserves parameters, whereas HERD actively reinforces past knowledge via elite replay and indirectly consolidates prior policies through URP's high-quality rollouts.

HERD achieves the fastest new-task adaptation across all three benchmarks. Most baselines adapt slowly because uncorrected reward miscalibration at task boundaries continues to mislead the planner. R-Walk and VCL show low STT on CW10, but their final success rates are only 0.13 and 0.14, reflecting that the threshold is crossed at a poor performance level rather than genuine fast adaptation. In contrast, HERD attains the lowest STT while simultaneously achieving the highest P on all benchmarks. Figure 3 corroborates this finding: starting from the second task onward, where task switches activate URP and Adaptive Horizon, HERD rises steeply to near-optimal success rates at every task boundary, whereas most baselines exhibit prolonged plateaus or fail to learn altogether. Even on harder tasks such as T6 and T7, where most baselines stagnate near zero, HERD still reaches high success rates. This rapid adaptation stems from the joint effect of URP and Adaptive Horizon: the former accelerates reward calibration so that the planner receives accurate signals shortly after each switch, while the latter prevents compounding dynamics errors from degrading early rollouts.

5.4 Ablation Study

Component Contribution. Among all single-component removals, dropping URP incurs the largest degradation in both P and STT, confirming that rapid reward calibration is the most critical factor for post-switch adaptation. Dropping AH further reduces P, as a fixed rollout depth allows dynamics errors to compound at task boundaries. When both are removed, P falls from 0.95 to 0.70 and

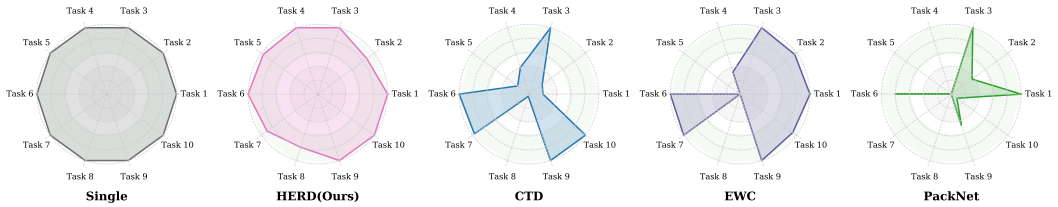


Figure 4: Final task-wise success rates on the CW10 for plasticity evaluation.The gray shaded area represents the single-task training performance, the colored polygons show each method's performance after completing the full task sequence.All results are averaged over 3 random seeds.

STT nearly doubles, indicating that planning quality accounts for the majority of HERD's gains. Elite Replay contributes along a different axis: its removal modestly affects adaptation speed but noticeably weakens forgetting mitigation. To isolate this effect, we compare the Elite-only variant (w/o URP + AH) against CTD in Table 1, which applies standard quality-agnostic replay. The consistent improvements in both P and F confirm that quality-stratified sampling strengthens knowledge retention beyond what naive replay achieves. Overall, no partial combination matches the full model, suggesting that the three components address complementary bottlenecks and benefit from joint optimization.

Robustness across Task Orders. Table 3 evaluates HERD across varying task sequences on CW10. HERD consistently achieves the best overall performance regardless of task order, maintaining higher average returns, lower forgetting, and faster adaptation. In contrast, baselines are highly sensitive to sequence permutations, exhibiting significant performance fluctuations. HERD's stability demonstrates that its effectiveness does not depend on favorable task orderings. We attribute this robustness to the principled design of its core components: rather than relying on task ids, URP and Adaptive Horizon adjust dynamically based on the current model state, while Elite Replay prioritizes experiences using order-agnostic trajectory statistics. Together, these mechanisms enable HERD to generalize seamlessly across arbitrary task sequences without order-specific tuning.

Plasticity Analysis. Figure 4 visualizes the per-task final success rates after all ten CW10 tasks have been trained. The single-task oracle (gray) provides an upper bound where each task is trained in isolation without interference. HERD (pink) closely tracks this oracle across all ten tasks, producing a nearly convex radar polygon, indicating that it achieves both high plasticity on new tasks and strong stability on old ones. In contrast, CTD (blue) collapses on multiple earlier tasks, most notably Tasks 1 and 6, reflecting its inability to retain knowledge without targeted replay. EWC (purple) preserves moderate performance overall but exhibits progressive degradation on later tasks, suggesting that parameter regularization alone cannot counteract the cumulative drift of the shared representation. PackNet (green) shows extreme imbalance, achieving near-perfect performance on a few tasks while collapsing on the majority, a consequence of rigid parameter isolation that sacrifices cross-task knowledge sharing. These results confirm that HERD's combination of planning-quality optimization (*how to learn*) and quality-stratified replay (*what to learn*) yields the most balanced stability–plasticity trade-off among all evaluated methods.

6 Conclusion

We presented HERD, a framework for continual world model learning that addresses the dual challenges of slow new-task adaptation and catastrophic forgetting through two complementary dimensions. On the *how to learn* side, Uncertainty-aware Reward Planning accelerates reward calibration via ensemble disagreement and guides exploration through optimistic planning, while Adaptive Planning Horizon dynamically adjusts rollout depth to match the current reliability of the dynamics model. On the *what to learn* side, Elite Experience Replay applies quality-stratified reweighting to direct training toward the most informative historical episodes. Experiments on Continual World, Continual Bench, and DM Control demonstrate that HERD consistently outperforms existing methods in average performance, forgetting mitigation, and adaptation speed.

Limitation. HERD has several limitations worth noting. First, the reward ensemble introduces computational and memory overhead that scales linearly with ensemble size M; more parameter-efficient uncertainty estimators could improve scalability. Second, our evaluation assumes discrete task sequences with fixed per-task budgets; extending HERD to settings with variable task durations or open-ended task streams remains future work.

Acknowledgments and Disclosure of Funding

This work is supported by NSFC Grant (No.62576364),GuangDong Basic and Applied Basic Research Foundation (2026B1515020071),Shenzhen Science and Technology Program (NO.JCYJ20241202124430041),Shenzhen Science and Technology Program (NO.SYSRD20250529113401002),Shenzhen Science and Technology Program (NO.ZDCY20250901113005006),CCF-Tencent Rhino-Bird Open Research Fund (NO.CCF-Tencent RAGR20250114) and Tencent JR2025TEG002,CAAI-Tencent Rhino-Bird Open Research Fund (NO.2026CAAI-Tencent-08).

References

[1] K. Khetarpal, M. Riemer, I. Rish, and D. Precup, "Towards continual reinforcement learning: A review and perspectives," *Journal of Artificial Intelligence Research*, vol. 75, pp. 1401–1476, 2022.

[2] D. Abel, A. Barreto, B. Van Roy, D. Precup, H. P. van Hasselt, and S. Singh, "A definition of continual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 50 377–50 407, 2023.

[3] S. Dohare, J. F. Hernandez-Garcia, Q. Lan, P. Rahman, A. R. Mahmood, and R. S. Sutton, "Loss of plasticity in deep continual learning," *Nature*, vol. 632, no. 8026, pp. 768–774, 2024.

[4] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[5] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 139–154.

[6] A. Mallya and S. Lazebnik, "Packnet: Adding multiple tasks to a single network by iterative pruning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 7765–7773.

[7] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[8] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, "Experience replay for continual learning," *Advances in neural information processing systems*, vol. 32, 2019.

[9] D. Isele and A. Cosgun, "Selective experience replay for lifelong learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[10] D. Ha and J. Schmidhuber, "World models," *arXiv preprint arXiv:1803.10122*, vol. 2, no. 3, p. 440, 2018.

[11] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse domains through world models," *arXiv preprint arXiv:2301.04104*, 2023.

[12] N. Hansen, H. Su, and X. Wang, "Td-mpc2: Scalable, robust world models for continuous control," *arXiv preprint arXiv:2310.16828*, 2023.

[13] S. Kessler, M. Ostaszewski, M. Bortkiewicz, M. Żarski, M. Wolczyk, J. Parker-Holder, S. J. Roberts, P. Mi *et al.*, "The effectiveness of world models for continual reinforcement learning," in *Conference on Lifelong Learning Agents*. PMLR, 2023, pp. 184–204.

[14] Y. Sun, H. Fu, M. Littman, and G. Konidaris, "Knowledge retention for continual model-based reinforcement learning," *arXiv preprint arXiv:2503.04256*, 2025.

[15] Z. Liu, G. Fu, C. Du, W. S. Lee, and M. Lin, "Continual reinforcement learning by planning with online world models," *arXiv preprint arXiv:2507.09177*, 2025.

[16] M. Wołczyk, M. Zając, R. Pascanu, Ł. Kuciński, and P. Miłoś, "Continual world: A robotic benchmark for continual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28 496–28 510, 2021.

[17] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. d. L. Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq *et al.*, "Deepmind control suite," *arXiv preprint arXiv:1801.00690*, 2018.

[18] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, N. Heess, and Y. Tassa, "dm_control: Software and tasks for continuous control," *Software Impacts*, vol. 6, p. 100022, 2020.

[19] S. Kessler, J. Parker-Holder, P. Ball, S. Zohren, and S. J. Roberts, "Same state, different task: Continual reinforcement learning without interference," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 7, 2022, pp. 7143–7151.

[20] H. Ahn, J. Hyeon, Y. Oh, B. Hwang, and T. Moon, "Prevalence of negative transfer in continual reinforcement learning: Analyses and a simple baseline," in *The Thirteenth International Conference on Learning Representations*, 2025.

[21] M. Wolczyk, M. Zając, R. Pascanu, Ł. Kuciński, and P. Miłoś, "Disentangling transfer in continual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 35, pp. 6304–6317, 2022.

[22] W. Chung, L. Cherif, D. Meger, and D. Precup, "Parseval regularization for continual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 37, pp. 127 937–127 967, 2024.

[23] Q. Zhou, X. Ruihang, P. Wang, W. Lu, X. Cao, N. Tan, and L. Shen, "Htac: Hierarchical task-aware composition for continual offline reinforcement learning," in *Forty-third International Conference on Machine Learning*.

[24] C. Pan, L. Ren, Y. Feng, L. Xiong, W. Wei, Y. Li, and X. Yang, "Multi-granularity knowledge transfer for continual reinforcement learning," *arXiv preprint arXiv:2401.15098*, 2024.

[25] J.-B. Gaya, T. Doan, L. Caccia, L. Soulier, L. Denoyer, and R. Raileanu, "Building a subspace of policies for scalable continual learning," *arXiv preprint arXiv:2211.10445*, 2022.

[26] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," *Advances in neural information processing systems*, vol. 30, 2017.

[27] W. Yue, B. Liu, and P. Stone, "t-dgr: A trajectory-based deep generative replay method for continual learning in decision making," *arXiv preprint arXiv:2401.02576*, 2024.

[28] F. Chen, F. Han, C. Guan, L. Yuan, Z. Zhang, Y. Yu, and Z. Zhang, "Stable continual reinforcement learning via diffusion-based trajectory replay," *arXiv preprint arXiv:2411.10809*, 2024.

[29] R. S. Sutton, "Integrated architectures for learning, planning, and reacting based on approximating dynamic programming," in *Machine learning proceedings 1990*. Elsevier, 1990, pp. 216–224.

[30] L. Kaiser, M. Babaeizadeh, P. Milos, B. Osinski, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine *et al.*, "Model-based reinforcement learning for atari," *arXiv preprint arXiv:1903.00374*, 2019.

[31] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *International conference on machine learning*. PMLR, 2019, pp. 2555–2565.

[32] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *arXiv preprint arXiv:1912.01603*, 2019.

[33] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, "Mastering atari with discrete world models," *arXiv preprint arXiv:2010.02193*, 2020.

[34] N. Hansen, X. Wang, and H. Su, "Temporal difference learning for model predictive control," in *ICML*, 2022.

[35] Y. Huang, K. Xie, H. Bharadhwaj, and F. Shkurti, "Continual model-based reinforcement learning with hypernetworks," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 799–805.

[36] L. Yang, L. Kuhlmann, and G. Kowadlo, "Augmenting replay in world models for continual reinforcement learning," *arXiv preprint arXiv:2401.16650*, 2024.

[37] A. Remonda, C. C. Terrell, E. E. Veas, and M. Masana, "Uncertainty-based experience replay for task-agnostic continual reinforcement learning," *Transactions on Machine Learning Research*, 2025.

[38] R. S. Sutton, A. G. Barto *et al.*, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

[39] Y. Yang, T. Zhou, J. Jiang, G. Long, and Y. Shi, "Continual task allocation in meta-policy network via sparse prompting," in *International Conference on Machine Learning*. PMLR, 2023, pp. 39 623–39 638.

[40] M. Malagon, J. Ceberio, and J. A. Lozano, "Self-composing policies for scalable continual reinforcement learning," *arXiv preprint arXiv:2506.14811*, 2025.

[41] T. M. Moerland, J. Broekens, A. Plaat, and C. M. Jonker, "Model-based reinforcement learning: A survey," *Foundations and Trends in Machine Learning*, vol. 16, no. 1, pp. 1–118, 2023.

[42] B. Kouvaritakis and M. Cannon, "Model predictive control," *Switzerland: Springer International Publishing*, vol. 38, no. 13-56, p. 7, 2016.

[43] A. Chaudhry, P. K. Dokania, T. Ajanthan, and P. H. Torr, "Riemannian walk for incremental learning: Understanding forgetting and intransigence," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 532–547.

[44] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner, "Variational continual learning," *arXiv preprint arXiv:1710.10628*, 2017.

[45] K. Huang, L. Shen, C. Zhao, C. Yuan, and D. Tao, "Solving continual offline reinforcement learning with decision transformer," *arXiv preprint arXiv:2401.08478*, 2024.

[46] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey." *Journal of Machine Learning Research*, vol. 10, no. 7, 2009.

[47] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on robot learning*. PMLR, 2020, pp. 1094–1100.

[48] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 5026–5033.

APPENDIX

A Algorithm Details **14**

B Benchmark Details **14**

B.1 Continual World 14

B.2 DM Control 15

B.3 Continual Bench 16

C Hyperparameters and Experimental Details **16**

D Compute Resources **16**

E More Ablation Studies **17**

E.1 Ablation study of adaptive horizon 17

E.2 Ablation study of URP 17

E.3 Ablation study of Elite Replay 18

F Task-Level Retention under Sequential Training **18**

G Comprehensive Analysis on Three Benchmarks **19**

A Algorithm Details

Algorithm 1 HERD: Horizon-adaptive Ensemble Reward Disagreement for Continual RL

Input: Task sequence $\{\mathcal{T}_k\}_{k=1}^{K}$, replay buffer $\mathcal{D}$, parameters $\theta = \{\theta_{\text{enc}}, \theta_{\text{dyn}}, \theta_{\text{rew}}, \theta_V, \theta_\pi\}$
Hyperparameters: β, $w_{\min}$, $w_{\max}$, β_{ema}, w, $\{\tau_j, H_j\}_{j=1}^{J}$, $H_{\max}$, ρ, λ_c, λ_r, λ_v
for $k = 1$ **to** K **do**
  $H_{\text{eff}} \leftarrow H_{\max}$
  for $t = 1$ **to** T_k **do**
    $z_0 \leftarrow h_{\theta_{\text{enc}}}(o_t)$
    $\tilde{r}(z,a) \leftarrow \bar{r}(z,a) + \beta \cdot \sigma_r(z,a) \,/\, \max(\text{mean}(|\bar{r}|), \epsilon)$ ▷ Optimistic planning
    $a_t \leftarrow \arg\max_\tau \text{MPPI}(z_0, \tilde{r}, H_{\text{eff}})$
    Execute a_t, observe r_t, o_{t+1}; store $(o_t, a_t, r_t, o_{t+1}) \to \mathcal{D}$
    Sample batch $\mathcal{B} \sim \mathcal{D}$ with $P(e) \propto N_{k'} \cdot w_{\text{tier}}(e) \,/\, \sum_{e'} w_{\text{tier}}(e')$ ▷ Elite replay
    for $h = 0$ **to** $H_{\max} - 1$ **do**
      $\hat{z}_{h+1} \leftarrow d_{\theta_{\text{dyn}}}(z_h, a_h)$
      $\sigma_r^{(h)} \leftarrow \text{std}\big(\{\hat{r}_m(z_h, a_h)\}_{m=1}^{M}\big)$
      $w_h \leftarrow \text{clip}\big(\sigma_r^{(h)}/\bar{\sigma}_r,\ w_{\min},\ w_{\max}\big) \,/\, \mathbb{E}\big[\text{clip}(\cdot)\big]$ ▷ Reward calibration
    end for
    $\theta \leftarrow \theta - \eta \nabla_\theta \sum_{h=0}^{H_{\max}-1} \rho^h \big[\lambda_c \mathcal{L}_{\text{dyn}}^{(h)} + \lambda_r w_h \ell_{\text{CE}}(\hat{r}_h, r_h) + \lambda_v \ell_{\text{CE}}(\hat{V}_h, V_h^{\text{tgt}})\big]$
    $\theta_\pi \leftarrow \theta_\pi - \eta_\pi \nabla_{\theta_\pi} \sum_{h=0}^{H} \rho^h \big[-\alpha \mathcal{H}(\pi(z_h)) - Q(z_h, \pi(z_h))\big]$ ▷ Policy update
    $u_t \leftarrow \beta_{\text{ema}} u_{t-1} + (1 - \beta_{\text{ema}}) \mathcal{L}_{\text{dyn}}(t)$
    $U_t \leftarrow w \cdot \mathcal{L}_{\text{dyn}}(t) + (1 - w) \cdot u_t$
    $H_{\text{eff}} \leftarrow H_{j^*}$ where $j^* = \min\{\, j \mid U_t > \tau_j \,\}$ ▷ Adaptive horizon
  end for
end for

B Benchmark Details

We evaluate HERD on three continual reinforcement learning benchmarks: Continual World [16], Continual Bench [15], and DM Control [17; 18]. These benchmarks collectively span robotic manipulation and locomotion tasks, varying in task structure, observation space, and dynamics characteristics, providing a comprehensive testbed for continual RL agents.

B.1 Continual World

Continual World [16] is a continual reinforcement learning benchmark built on Meta-World [47], in which a Sawyer robotic arm performs a sequence of simulated tabletop manipulation tasks. The benchmark defines two standard task sequences, CW10 and CW20. CW10 comprises 10 manipulation tasks presented in sequence: hammer, push-wall, faucet-close, push-back, stick-pull, handle-press-side, push, shelf-place, window-close, and peg-unplug-side. CW20 repeats these 10 tasks twice to evaluate the agent's performance when revisiting previously learned tasks. Each task provides a 12-dimensional observation space covering end-effector position, gripper state, and task-relevant object positions, and a 4-dimensional continuous action space for end-effector motion and gripper actuation. The agent trains on each task for a fixed number of steps before switching to the next, with no access to prior task data. Continual World jointly evaluates forward transfer, measured as the acceleration of new-task learning due to prior knowledge, and forgetting, measured as performance degradation on previously learned tasks.

To adequately account for task correlations, we adopt multiple task sequences from Continual World in our experiments 3, as specified below:

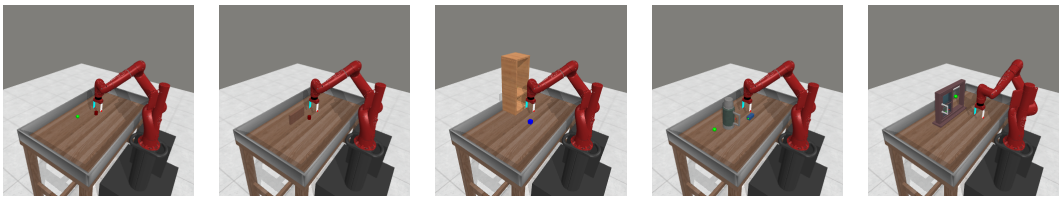
• Order 0: [hammer-v2, push-wall-v2, faucet-close-v2, push-back-v2, stick-pull-v2, handle-press-side-v2, push-v2, shelf-place-v2, window-close-v2, peg-unplug-side-v2]

• Order 1: [push-v2, window-close-v2, peg-unplug-side-v2, shelf-place-v2, handle-press-side-v2, push-back-v2, hammer-v2, stick-pull-v2, push-wall-v2, faucet-close-v2]

• Order 2: [handle-press-side-v2, peg-unplug-side-v2, push-back-v2, stick-pull-v2, push-v2, shelf-place-v2, faucet-close-v2, window-close-v2, push-wall-v2, hammer-v2]



(a) Faucet Close (b) Hammer (c) Handle Press Side (d) Peg Unplug Side (e) Push Back



(f) Push (g) Push Wall (h) Shelf Place (i) Stick Pull (j) Window Close

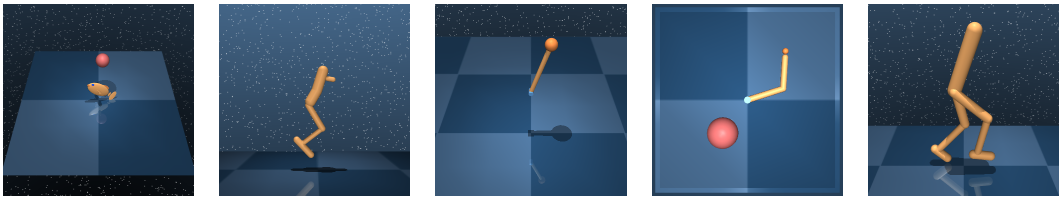
Figure 5: Continual World benchmark tasks.

B.2 DM Control

DM Control [17; 18] is a widely used continuous control benchmark in reinforcement learning. Built on the MuJoCo physics engine [48] and the dm control software stack, it provides a suite of locomotion tasks spanning diverse morphologies and difficulty levels. Representative domains include walker for bipedal walking and running, cheetah for quadruped locomotion, hopper for single-leg hopping, cartpole for balancing and swing-up, finger for dexterous manipulation, and humanoid for high-dimensional whole-body control. Each domain defines multiple tasks of increasing difficulty, such as walker-stand, walker-walk, and walker-run. Observation spaces are continuous and domain-specific in dimensionality; action spaces are continuous and low-dimensional. In the continual RL setting, DMC tasks are organized into sequences in which the agent must acquire distinct locomotion skills successively, testing its ability to retain motor control policies across task switches. We include DMC to evaluate HERD on locomotion tasks, complementing the manipulation benchmarks Continual World and Continual Bench. For cross-domain continual learning, we construct the



(a) Acrobot Swingup (b) Cartpole Swingup (c) Cheetah Run (d) Cup Catch (e) Finger Spin



(f) Fish Swim (g) Hopper Hop (h) Pendulum Swingup (i) Reacher Easy (j) Walker Walk

Figure 6: DM Control benchmark tasks.

DMC_Cross benchmark comprising 10 tasks from 10 distinct DMControl domains: walker-walk, cheetah-run, hopper-hop, reacher-easy, pendulum-swingup, cartpole-swingup, finger-spin, cup-catch, fish-swim, acrobot-swingup.

B.3 Continual Bench

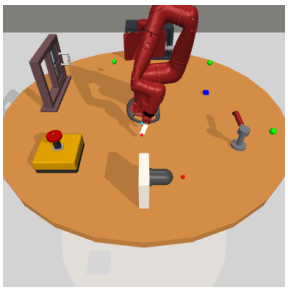


Figure 7: Continual Bench environment.

Continual Bench [15] is a continual reinforcement learning benchmark designed for the unified-dynamics setting. Unlike Continual World, where tasks are temporally concatenated and may suffer from inconsistent object semantics or conflicting dynamics, Continual Bench places multiple manipulation tasks spatially within a single simulated tabletop environment. This design ensures that all tasks share the same underlying physics, state space, and action space, allowing the benchmark to evaluate continual learning under a unified world model. Built on the MuJoCo physics engine [48] and task primitives from Meta-World [47], Continual Bench uses a Sawyer robotic arm and includes six tasks of varying difficulty: pick-place, button-press, door-open, peg-unplug, window-close, and faucet-close. The tasks are arranged around the same table with large spatial separation, inducing distributional shifts across task switches while preserving meaningful overlap for transfer. All objects are present in the scene concurrently, and task changes are specified only through changes in the reward function.

C Hyperparameters and Experimental Details

The hyperparameters used across our experiments are summarized in Table 4. Our algorithm is implemented based on the TD-MPC2[12] architecture. Detailed configurations for the network, optimization, MPC planning, and the proposed URP mechanism are listed in the table.

Table 4: Hyperparameter settings for our method.

Hyperparameter	**Value**
Architecture & Optimization	
Batch size	256
Learning rate	3×10^{-4}
Encoder learning rate scale	0.3
Latent dimension	512
MLP dimension	512
Encoder dimension	256
Number of Q-networks	5
Target network update rate (τ)	0.01
Discount factor (γ)	0.95
Gradient clip norm	20
Planning (MPC)	
Planning iterations	6
Sample size (N)	512
Number of elites (K)	64
Min/Max horizon	$3/10$
Min/Max std	$0.05/2.0$
Temperature	0.5
URP Mechanism	
Number of reward heads (M)	5
Base bonus coefficient (β_0)	0.5
Bonus warmup steps	5000
Bonus decay rate	0.9995
Task switch grace period	30000
Bonus cap ratio	0.2

D Compute Resources

All experiments were conducted on NVIDIA RTX 5090 GPUs (32 GB). The aggregate compute for all experiments reported in this paper, including hyperparameter tuning, ablation studies, and baseline reproductions, totals approximately 1400 GPU-hours.

E More Ablation Studies

E.1 Ablation study of adaptive horizon

Table 5 compares the adaptive planning horizon (AH) with multiple fixed-horizon variants. Overall, AH consistently outperforms all fixed horizons in the table: it achieves the highest average performance, the lowest STT, and maintains a competitive level of forgetting. In contrast, fixed horizons exhibit a clear trade-off between performance (P) and STT. For example, the conservative H=1 strategy ensures acceptable average performance but adapts too slowly. H=3 neither leverages long-horizon prediction to escape local optima nor avoids introducing accumulated errors, resulting in a marked degradation in average performance. As the horizon increases from 5 to 7, both P and STT improve in tandem, which is associated with the greater foresight afforded by longer rollouts. The H=10 case further reveals that an excessively long fixed horizon, while capable of accelerating convergence to some extent, can amplify prediction errors during early task switches. By adopting a more conservative planning length at the onset of a task switch and extending the effective horizon once predictions stabilize, AH alleviates the difficulty that fixed-horizon strategies face in simultaneously balancing P and STT.

Table 5: Ablation study of adaptive horizon (AH) on the CW10 benchmark, comparing AH against fixed planning horizons.

Method	P (↑)	F (↓)	STT (↓)
Adaptive Horizon	$\mathbf{0.82_{\pm 0.01}}$	$\mathbf{-0.02_{\pm 0.02}}$	$\mathbf{0.30_{\pm 0.02}}$
Horizon=1	$0.74_{\pm 0.04}$	$-0.07_{\pm 0.04}$	$0.53_{\pm 0.03}$
Horizon=3	$0.68_{\pm 0.02}$	$-0.02_{\pm 0.02}$	$0.52_{\pm 0.01}$
Horizon=5	$0.74_{\pm 0.03}$	$-0.06_{\pm 0.06}$	$0.42_{\pm 0.02}$
Horizon=7	$0.80_{\pm 0.02}$	$-0.01_{\pm 0.05}$	$0.34_{\pm 0.03}$
Horizon=10	$0.77_{\pm 0.03}$	$-0.05_{\pm 0.06}$	$0.30_{\pm 0.02}$

E.2 Ablation study of URP

Table 6 examines URP from two perspectives: ensemble size and the individual contribution of its two routes. Increasing the number of reward heads from 3 to 7 steadily improves P while reducing F, indicating that richer ensemble disagreement yields more accurate uncertainty estimates and thus more effective calibration. Considering the trade-off between performance gain and computational overhead, we adopt 5 heads for all experiments in this paper. The two routes serve distinct and complementary roles. Route A (Optimistic Planning) primarily accelerates adaptation, achieving considerably lower STT than Route B, but provides limited forgetting mitigation due to its focus on exploration rather than model correction. Route B (Reward Model Calibration) excels at knowledge retention, producing the strongest negative F among all variants, yet its high STT reveals that training-time calibration alone cannot accelerate the planner's early-phase behavior. Combining both routes recovers the advantages of each while compensating for their individual weaknesses, yielding the best overall balance of adaptation speed, performance, and forgetting mitigation.

Reward ensemble. The reward-head variants in Table 6 vary the number of reward predictors in URP. This study disables Adaptive Horizon and Elite Replay to isolate the reward ensemble. Increasing the ensemble from three to five heads raises P from 0.88 to 0.90 and lowers STT from 0.30 to 0.28, while F changes slightly from -0.04 to -0.03. Seven heads further increase P to 0.94 and improve F to -0.11, but their mean STT remains 0.28. The results therefore suggest that additional heads can benefit task performance and retention, while the adaptation gain is already attained with five heads in this comparison. We retain five heads as the default because they provide comparable adaptation speed with lower computational overhead than seven. This choice reflects a performance–cost trade-off rather than the configuration with the highest P or most negative F.

Table 6: Ablation study of URP on the CW10 benchmark. "URP k Heads" denotes the number of reward ensemble heads used. Route A refers to Optimistic Planning, while Route B refers to Reward Model Calibration. "URP 5 Heads" and "Route A + B" denote the same experiment.

Experiment	P (↑)	F (↓)	STT (↓)
URP 3 Heads	$0.88_{\pm 0.02}$	$-0.04_{\pm 0.01}$	$0.30_{\pm 0.02}$
URP 5 Heads	$0.90_{\pm 0.02}$	$-0.03_{\pm 0.02}$	$0.28_{\pm 0.01}$
URP 7 Heads	$0.94_{\pm 0.03}$	$-0.11_{\pm 0.03}$	$0.28_{\pm 0.03}$
Route A	$0.70_{\pm 0.02}$	$0.03_{\pm 0.02}$	$0.35_{\pm 0.02}$
Route B	$0.73_{\pm 0.05}$	$-0.18_{\pm 0.03}$	$0.50_{\pm 0.03}$
Route A + B	$0.90_{\pm 0.02}$	$-0.03_{\pm 0.02}$	$0.28_{\pm 0.01}$

E.3 Ablation study of Elite Replay

Based on HERD, we replace Elite Replay with alternative replay strategies to conduct an ablation study, as shown in Table 7. All variants retain the AH and URP modules, so that the comparison isolates the effect of the replay strategy. Elite Replay achieves the best overall performance across all three metrics, consistently outperforming Uniform Replay, GDM, and PER. In contrast, removing replay leads to substantial performance degradation and severe forgetting. These results indicate that quality-stratified episode sampling provides more effective replay signals than random, gradient-based, or priority-based sampling, improving retention without sacrificing adaptation speed.

Table 7: Ablation study of different replay schemas on the CW10 benchmark. Uniform denotes random replay, GDM selects replay samples based on gradient diversity, and PER prioritizes samples according to their TD error.

Method	P (↑)	F (↓)	STT (↓)
Elite (Ours)	$\mathbf{0.95_{\pm 0.02}}$	$\mathbf{-0.06_{\pm 0.01}}$	$\mathbf{0.26_{\pm 0.03}}$
Uniform	$0.93_{\pm 0.01}$	$-0.02_{\pm 0.01}$	$0.28_{\pm 0.03}$
No Replay	$0.15_{\pm 0.04}$	$0.95_{\pm 0.03}$	$0.29_{\pm 0.02}$
GDM	$0.91_{\pm 0.01}$	$-0.01_{\pm 0.03}$	$0.32_{\pm 0.01}$
PER	$0.92_{\pm 0.02}$	$-0.03_{\pm 0.03}$	$0.33_{\pm 0.02}$

F Task-Level Retention under Sequential Training

Figure 8 demonstrates that HERD is the only method whose per-task success rate distribution closely aligns with the single-task oracle across all three task orders. This indicates that the improvement in plasticity for new tasks does not come at the expense of stability for previous tasks. This balance stems from the ability of URP and Adaptive Horizon to generate high-quality rollouts, thereby preventing the shared world model from degrading during the adaptation process; meanwhile, Elite Replay explicitly reinforces knowledge of previous tasks during training.

In contrast, the various baseline methods exhibit failure modes inherent to their underlying paradigms. Regularization methods such as EWC can maintain partial coverage under Order 1, but exhibit increasing asymmetry under other task orders, as parameter constraints cannot prevent the cumulative drift of shared representations. Parameter isolation methods such as PackNet yield the most polarized performance distributions: they approach perfect scores on specific isolated tasks but drop to near zero on others, reflecting the fundamental tension between avoiding interference and sharing knowledge. CTD, which relies on naive replay without quality control, manages to retain fragmented performance; however, as the task order changes, its performance polygon shrinks and fragments, suggesting that uniform replay is insufficient for robust knowledge retention.

This cross-order comparison demonstrates that the balanced performance distribution of HERD is not a spurious artifact of a specific task order. Because its components adapt to the current model state rather than explicit task identities, HERD maintains a robust stability-plasticity balance across diverse task sequences.

G Comprehensive Analysis on Three Benchmarks

Figure 9 visualizes the evolutionary performance of various continual-learning methods across three CW10 task orders, as well as on the Continual Bench (CB) and DeepMind (DM) Control suites. Each panel displays a task-by-task success matrix: the rows correspond to the incremental training stages, while the columns denote the evaluation performance on individual tasks. In this visualization scheme, the main diagonal reflects a method's plasticity (its capacity to rapidly adapt to a newly introduced task), whereas the lower-triangular region captures its stability (its ability to retain or even improve upon previously acquired knowledge).

- **Performance and Order Robustness on CW10:** On the CW10 benchmark, HERD exhibits consistently high success rates along the diagonal, which indicates rapid and highly effective adaptation to new tasks. Furthermore, it maintains exceptionally strong performance in the off-diagonal regions associated with previously encountered tasks. Visually, this translates into dense, prominently dark patterns uniformly distributed throughout the lower-triangular sections of the matrices. This pattern provides compelling evidence of limited forgetting and positive backward transfer as training progresses. Mechanistically, this sustained balance can be attributed to the synergistic components of HERD: URP and Adaptive Horizon generate high-quality rollouts that prevent the shared world model from degrading, while Elite Replay explicitly reinforces critical prior knowledge. Notably, these dark lower-triangular patterns persist across all three CW10 task orders, underscoring the algorithm's robust stability regardless of task sequence variations. In stark contrast, the baseline methods largely fail to sustain this equilibrium. For instance, regularization approaches like EWC typically exhibit progressively fading lower-triangular regions, indicative of cumulative representation drift and catastrophic forgetting. Parameter isolation methods such as PackNet display highly polarized heatmaps, succeeding on specific isolated tasks while dropping to near zero on others, which reflects the fundamental tension between avoiding task interference and enabling constructive knowledge sharing. Furthermore, CTD, which relies on naive replay without quality control, manages to retain only sporadic performance that visibly shrinks as the training sequence elongates.
- **Generalization to CB and DM Control:** The dominance of HERD extends seamlessly to both the Continual Bench (CB) and DM Control benchmarks. Across these evaluations, the heatmaps for HERD consistently display robust, dark lower-triangular regions, implying stable, long-term knowledge accumulation and profound resilience against task interference. This visual evidence confirms that HERD maintains competitive plasticity for acquiring advanced new skills while rigorously preserving previously learned competence. Baseline algorithms often suffer severe performance degradation across these evaluations, especially when forced to navigate the highly heterogeneous task structures and complex environmental dynamics introduced by DM Control. In contrast, the underlying components of HERD adapt flexibly to the current model state rather than relying on explicit task identities. Consequently, its behavior generalizes effectively, establishing a state-of-the-art balance between plasticity and stability across diverse continual reinforcement learning scenarios.

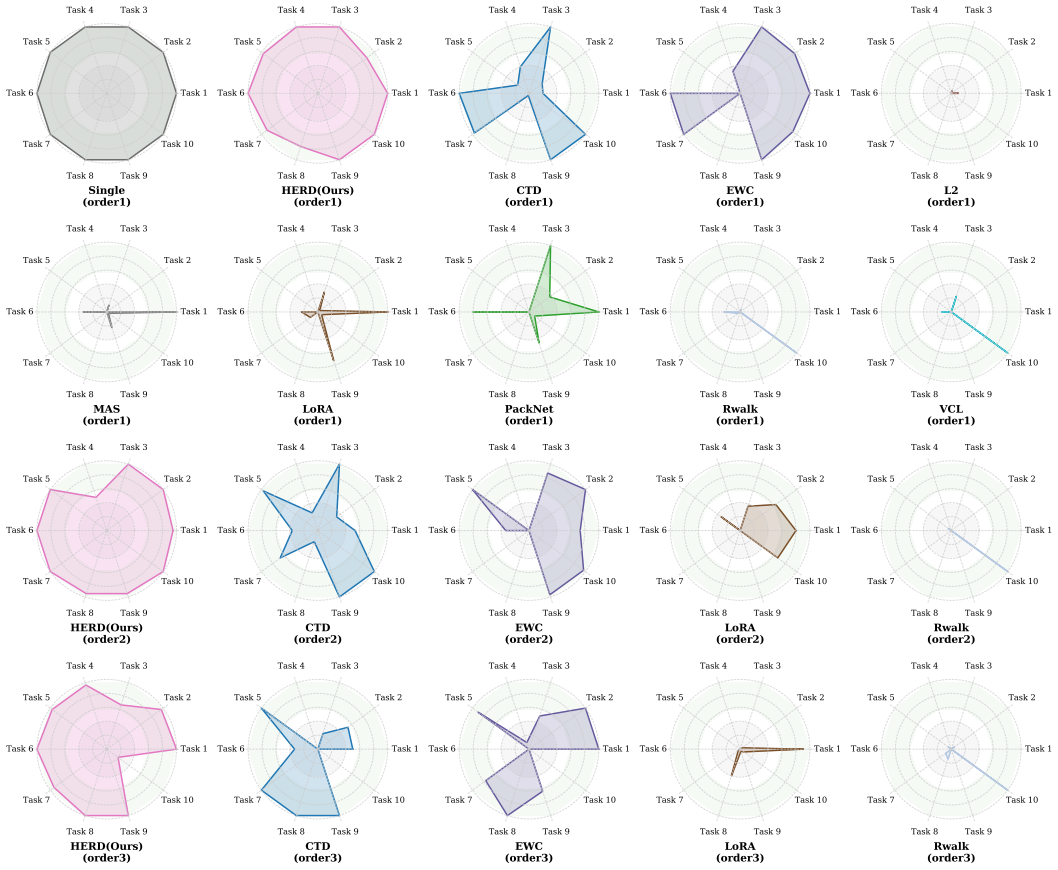


Figure 8: Final success rates on each CW10 task after sequential training under three task orderings. Each radar plot shows the per-task performance of a single method; the gray dashed circle indicates the single-task upper bound.

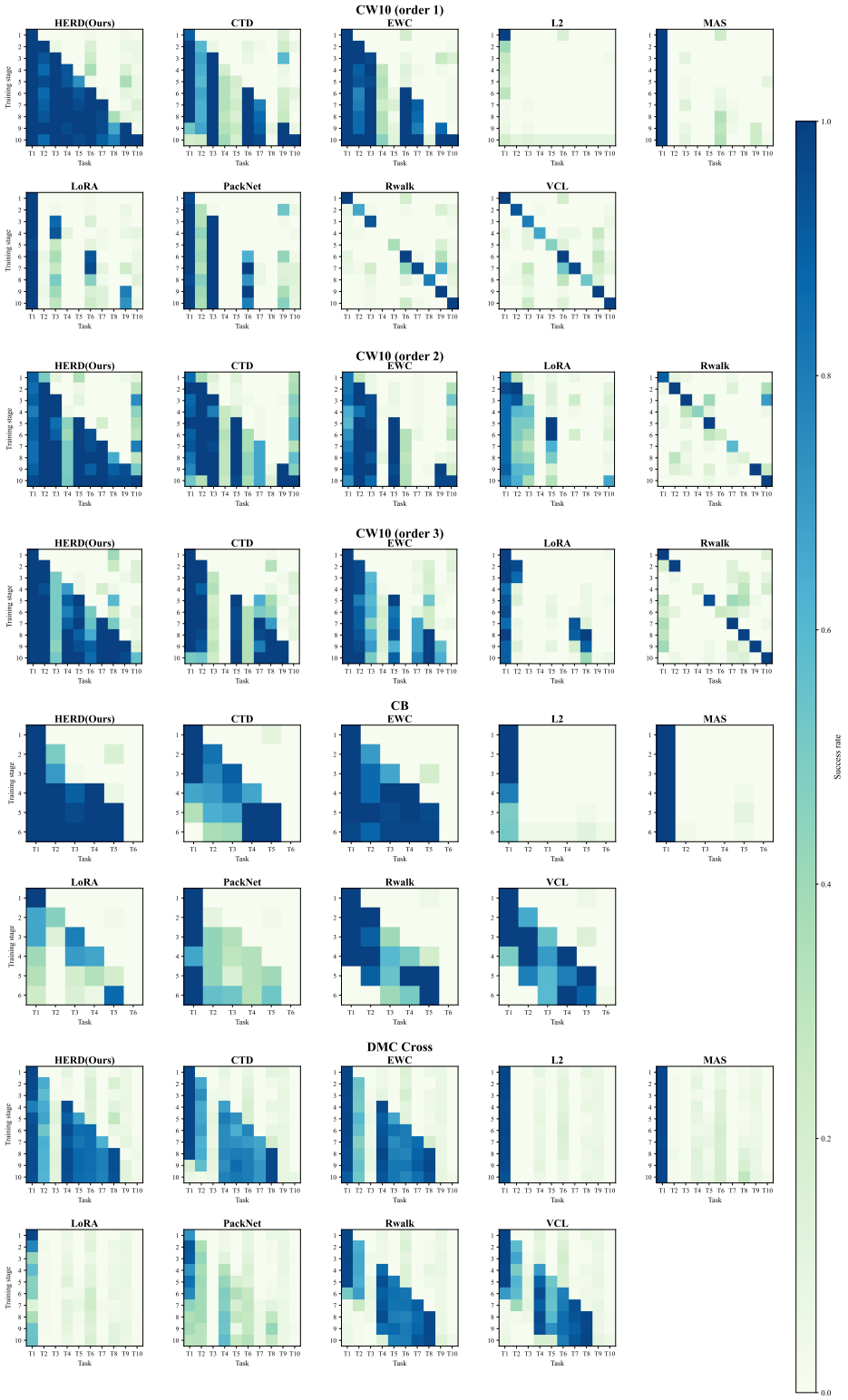


Figure 9: Heatmaps of per-task success after each training stage for nine methods on CW10 (orders 1–3), CB, and DMC Cross; cell (i, j) is performance on task j after training through stage i, averaged over 3 seeds.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit. Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper. The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found. IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims and contributions are explicitly outlined in the introduction and validated through comprehensive empirical evaluations.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our approach, including potential assumptions and computational overhead, in the concluding limitation section.

   Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [No]

   Justification: This paper is primarily an empirical study focused on algorithmic improvements and does not claim any formal theoretical results.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Detailed explanations of the algorithm, architecture, and evaluation protocols are provided to facilitate faithful reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: The source code and environment setups will be made publicly open-source upon the acceptance of the paper.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Comprehensive experimental settings, including environment details and hyperparameter choices, are clearly documented in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All empirical results report the mean performance along with error bars computed across multiple independent random seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Information regarding the specific hardware configurations used and the estimated computational time for the experiments is detailed in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research methodology and execution fully comply with the NeurIPS Code of Ethics guidelines.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: As foundational algorithmic research in reinforcement learning, this work does not have direct, foreseeable negative societal impacts.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release any high-risk models, generators, or datasets that necessitate specific safeguards.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We properly credit and cite all existing benchmarks, environments, and codebases used, strictly following their respective terms of use.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: This paper relies exclusively on existing benchmark environments and does not introduce or release any new data assets.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The conducted research relies entirely on simulated environments and involves no human subjects or crowdsourcing.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Since the study does not involve human subjects, IRB approval is not applicable to this work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: We did not utilize any Large Language Models as part of the core methodology or scientific development of this work.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.