Maximum Entropy Behavior Exploration for Sim2Real Zero-Shot Reinforcement Learning

Jiajun Hu[1,†], **Núria Armengol Urpí**[2,3,†], **Jin Cheng**[2], **Stelian Coros**[2]
jiajun.hu@epfl.ch, nuriaa@ethz.ch

[1]**Department of Mechanical Engineering, EPFL**
[2]**Department of Computer Science, ETH Zurich**
[3]**Max Planck Institute for Intelligent Systems**

[†] Corresponding authors

Abstract

Zero-shot reinforcement learning (RL) algorithms aim to learn a family of policies from a reward-free dataset, and recover optimal policies for any reward function directly at test time. Naturally, the quality of the pretraining dataset determines the performance of the recovered policies across tasks. However, pre-collecting a relevant, diverse dataset without prior knowledge of the downstream tasks of interest remains a challenge. In this work, we study *online* zero-shot RL for quadrupedal control on real robotic systems, building upon the Forward-Backward (FB) algorithm. We observe that undirected exploration yields low-diversity data, leading to poor downstream performance and rendering policies impractical for direct hardware deployment. Therefore, we introduce FB-MEBE, an online zero-shot RL algorithm that combines an unsupervised behavior exploration strategy with a regularization critic. FB-MEBE promotes exploration by maximizing the entropy of the achieved behavior distribution. Additionally, a regularization critic shapes the recovered policies toward more natural and physically plausible behaviors. We empirically demonstrate that FB-MEBE achieves improved performance compared to other exploration strategies in a range of simulated downstream tasks, and that it renders natural policies that can be seamlessly deployed to hardware without further finetuning. Videos and code available on our website.

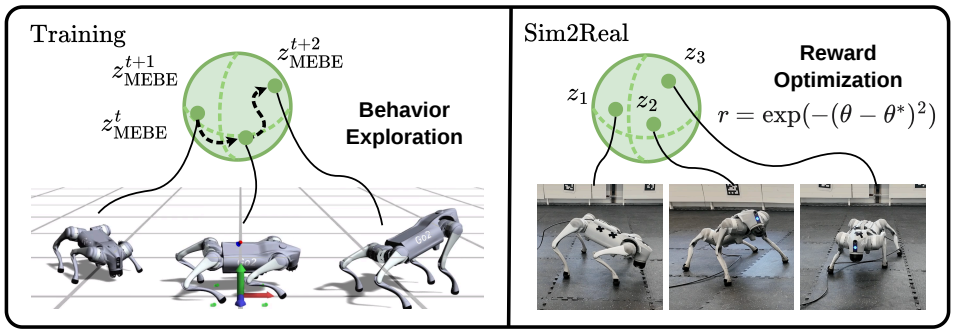

Figure 1: FB-MEBE improves behavior exploration by maximizing the behavior entropy during training. With the regularizer critic, FB-MEBE achieves zero-shot hardware deployment by optimizing reward at inference time.

1 Introduction

Zero-shot reinforcement learning (RL) (Touati et al., 2023) has emerged as a promising paradigm for developing general-purpose agents. By learning successor measures that decouple dynamics from rewards, these methods generate near-optimal policies for a big family of reward functions at test time, with minimal amount of computation and relying solely on unlabeled, precollected data. This departs from standard, fixed-reward-based RL (Sutton & Barto, 1998), which typically requires training a new policy whenever the objective changes. As these methods scale beyond complex and broader settings, they are usually referred to as Behavior Foundation Models (BFMs) (Tirinzoni et al., 2025). Such generalization capabilities are particularly appealing for robotics, where usually learning relevant policies involves tedious reward engineering. While general-purpose foundation models trained on vast amounts of (un)labeled data have emerged as a promising paradigm for robotic manipulation (Intelligence et al., 2025; Zitkovich et al., 2023; Kim et al., 2025), the application of BFMs for legged systems still remains underexplored. One of the main bottlenecks is the reliance on large, diverse, task-agnostic precollected datasets (Laskin et al., 2021), which are often unavailable at scale for legged robots. Recently, Tirinzoni et al. (2025) proposed an online zero-shot RL algorithm grounding policy learning and exploration towards an external motion capture dataset. Based on this idea, BFM-Zero (Li et al., 2025) represents an early effort to transfer such zero-shot RL methods to a humanoid robot. However, these methods still rely on supervision from external datasets. In this paper, we focus on the quadrupedal setting. We observe that naive deployment of online FB suffers from ineffective exploration and has difficulties in discovering the sparse manifold of stable locomotion. In contrast to previously explored efforts (Li et al., 2025), large motion capture datasets are not readily available for quadrupedal robots and thus cannot be used to regularize exploration.

We thus propose FB-MEBE (Maximum Entropy Behavior Exploration), an online zero-shot RL framework based on the forward-backward (FB) algorithm (Touati & Ollivier, 2021), which does not require access to external datasets.

FB-MEBE leverages an unsupervised behavior exploration strategy that guides the agent to maximize the entropy of the achieved behavior distribution, similarly to (Pitis et al., 2020). Closest to our work, (Urpí et al., 2025) framed the exploration problem in FB as a minimization of the epistemic uncertainty that the algorithm has of its representations. In our case, we adopt a frequentist approach, guiding the agent towards reaching behaviors with low visitation count. This strategy implicitly induces a curriculum on the behaviors with which to explore next, effectively pushing the frontier of achievable behaviors. Furthermore, to ensure that the manifold of solutions induced by the unsupervised strategy is aligned with the downstream task of interest, we embed an inductive bias to favor physically plausible behaviors via a behavior regularizer. This forces the recovered policies to maintain their zero-shot capabilities while being readily deployable to real hardware.

We empirically demonstrate that FB-MEBE achieves improved downstream performance compared to other exploration strategies in simulation in a range of downstream tasks, and that it renders policies with natural behaviors that can be seamlessly deployed to hardware without further finetuning. To the best of our knowledge, this is the second instantiation of the FB algorithm on real robotic systems, and the very first to do so entirely online without relying on prior external datasets. In summary, in this work we provide a maximum entropy exploration strategy for efficiently learning FB representations online that a) exhibits improved downstream performance in quadruped tasks in simulation compared to other exploration alternatives, b) renders policies with smooth, physically plausible behaviors and c) enables the direct, zero-shot deployment of recovered policies onto real hardware.

2 Related Work

Zero-shot RL Zero-shot RL spans a range of methods that train agents on unsupervised data to enable zero-shot generalization to a wide range of tasks. It traces back to successor representations

(Dayan, 1993), a direct extension to the continuous MDPs being the successor features (Barreto et al., 2017). In (Borsa et al., 2019), given a handcrafted feature map on the states, it defines the set of all reward functions that lie on the linear span of the features. Several extensions have been proposed to learn the features map (Hansen et al., 2019; Laskin et al., 2021). The forward backward algorithm, FB (Touati & Ollivier, 2021; Touati et al., 2023), applies a similar idea to a finite-rank model of successor measures, adopting a contrastive loss computing pairwise dot-products across each training batch, and learns the feature map and corresponding successor features *jointly*. FB has been subsequently extended to alternative parameterizations (Bagatella et al., 2025; Cetin et al., 2025), for zero-shot imitation (Pirotta et al., 2024), offline training from low quality data (Jeen et al., 2024), online training regularized with unlabeled expert data(Tirinzoni et al., 2025), training on environments with different dynamics (Bobrin et al., 2025), extending it to optimising general utilities beyond linear RL (Bagatella et al., 2026) online finetuning (Sikchi et al., 2025), test-time task inference (Rupf et al., 2025) or self-supervised exploration (Urpí et al., 2025; Sun et al., 2025). Other zero-shot methods are HILP (Park et al., 2024a), that learn state features with a goal reaching loss in a latent space and PSM (Agarwal et al., 2025) learning an affine decomposition of the successor measure for a discrete codebook of policies.

Exploration in unsupervised RL There has been vast work on exploration in unsupervised RL (Eysenbach et al., 2018; Burda et al., 2018; Sharma et al., 2020; Park et al., 2024b) or unsupervised goal-conditioned RL focused on discovering a wide range of goals and learning corresponding goal-reaching policies (Bagaria & Konidaris, 2019; Mendonca et al., 2021), exploring with goals that have lower density (Florensa et al., 2018; Pitis et al., 2020; Pong et al., 2020; Diaz-Bone et al., 2026). Most of these works, in contrast to zero-shot RL require separate methods for learning policies. In the zero-shot realm, VISR (Hansen et al., 2019) and its successor APS (Liu & Abbeel, 2021) learn features online, where the representation is directly optimized to satisfy a diversity criterion. FB-MEBE also learns features online but decouples the representation-learning objective from the exploration objective. Exploration in FB-MEBE is driven by setting intrinsic goals that maximize the entropy of the achieved behavior distribution. Closest to our work, Urpí et al. (2025) framed the exploration problem in online FB as a minimization of epistemic uncertainty, exploring with task embeddings that lead to the highest information gain, while Sun et al. (2025) leveraged RND (Burda et al., 2018) to promote data diversity.

3 Background

We consider a standard reward-free Markov decision process (MDP), defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \gamma, d_0)$, where $\mathcal{S}, \mathcal{A}$ are the state and action space, P is the transition kernel, $\gamma \in (0, 1)$ is the discount factor and d_0 is the initial state distribution. Given the MDP $\mathcal{M}$, a policy $\pi : \mathcal{S} \to \Delta\mathcal{A}$, mapping states to probabilities over actions, its successor measure (Dayan, 1993; Blier et al., 2021) for any initial state-action pair $(s_0, a_0) \in \mathcal{S} \times \mathcal{A}$ is the discounted cumulative distribution of visiting future states when starting from state $s_0 \in \mathcal{S}$, take action $a \in \mathcal{A}$ and following policy π thereafter. Formally, it is defined as

$$M^\pi(X|s_0, a_0) = \sum_{t\geq 0} \gamma^t P(s_{t+1} \in X|s_0, a_0, \pi) \quad \forall X \in \mathcal{S}. \tag{1}$$

Conveniently, the action-value function for policy π and any reward function $r : \mathcal{S} \to \mathbb{R}$, can be expressed as

$$Q_r^\pi(s, a) := \mathbb{E}\big[\sum_{t\geq 0} \gamma^t r(s_{t+1})|s, a, \pi\big] = \int_{s'\in\mathcal{S}} M^\pi(ds' \mid s, a)\, r(s'), \tag{2}$$

which decomposes the Q-function between a reward-agnostic measure M^π, modeling the evolution of the policy in the environment, and the reward function. There exist several algorithms for learning zero-shot policy agents (Eysenbach et al., 2021; Borsa et al., 2019; Agarwal et al., 2025), but in this

work we will focus on the Forward Backward (FB) (Touati & Ollivier, 2021; Touati et al., 2023) algorithm. See related work and Section K for extensive analysis.

Given a representation space $\mathcal{Z} \subseteq \mathbb{R}^d$ (usually the unit hypersphere of radius $\sqrt{d}$, and a family of policies $\pi_{z \in \mathcal{Z}}$, parameterized by z, the FB algorithm aims to learn a *forward* representation $F^\pi : \mathcal{S} \times \mathcal{A} \times \mathcal{Z} \to \mathcal{Z}$ and a *backward* representation $B : \mathcal{S} \to \mathcal{Z}$ such that the successor measure in Equation (1) admits the low-rank factorization

$$M^{\pi_z}(ds' \mid s, a) \approx F(s, a, z)^\top B(s')\rho(ds') \quad \pi_z(s) = \arg\max_a F(s, a, z)^\top z, \tag{3}$$

where ρ is the state distribution, $z \in \mathcal{Z}$ is the task embedding of policy π_z. The FB networks are trained to minimize a temporal-difference *contrastive* loss over transitions $\{(s, a, s')\}$, independent future states $\{s^+\}$, and reward embeddings $z \sim \nu$, where ν is a distribution over $\mathcal{Z}$ specified later:

$$\begin{aligned}\mathcal{L}_{\text{FB}}(F, B) = \mathbb{E}_{\substack{z\sim\nu,\,(s,a,s')\sim\rho,\\ s^+\sim\rho,\,a'\sim\pi_z(s')}} &\left[\left(F(s, a, z)^\top B(s^+) - \gamma \overline{F}(s', a', z)^\top \overline{B}(s^+)\right)^2\right] \\ &- 2\,\mathbb{E}_{z\sim\nu,\,(s,a,s')\sim\rho}\left[F(s, a, z)^\top B(s')\right],\end{aligned} \tag{4}$$

where $\overline{F}$ and $\overline{B}$ denote target networks and ρ is the dataset distribution. In practice, as per Touati et al. (2023) additional regularization terms are used, see Section A for details.

The $\arg\max$ in Equation (3) is approximated by jointly training an actor network with the following loss:

$$\mathcal{L}_{\text{actor}}(\pi) = -\mathbb{E}_{z\sim\nu,\, s\sim\rho,\, a\sim\pi_z(s)}\left[F(s, a, z)^\top z\right]. \tag{5}$$

After training, given a reward function r specified at test time, we can find the task embedding z_r parameterizing the optimal learnt policy for reward r, by performing a simple linear regression of r onto the span of the learnt features B. For FB, the reward embedding can be computed simply via $z_r = \mathbb{E}_{s\sim\rho}\big[B(s)r(s)\big]$ (Touati & Ollivier, 2021), where the expectation is approximated through sampling from the dataset distribution used for training (or a subset).

If Equation (3) holds, then for any reward function r in the linear span of the features, the policy π_{z_r} is guaranteed to be optimal with optimal Q-function $Q^*_r(s, a) = F(s, a, z_r)^\top z_r$ (Touati & Ollivier, 2021). In practice, the quality of the learnt representation depends on the chosen factorization d and the coverage of the data ρ used to train them. In this work, we focus on the second issue.

The optimality criteria of FB (Touati & Ollivier, 2021) shows that it is sufficient to sample latents z from a distribution ν that has support over the entire hypersphere. Previous works (Tirinzoni et al., 2025) propose biasing a fraction r of the latent samples for exploration and training to allocate more model capacity for goal-reaching tasks. Our work focuses on a novel sampling strategy for efficient exploration.[1]

4 Limitations of Regularized Undirected Exploration

Naturally, how well we can recover the optimal policies depends on the quality of data used to learn them. However, when d is finite and when the dataset has narrow data distributions, the learning objective lacks sufficient inductive bias on which policies to favor. This often leads to a collapse of the FB representations.

Consistent with empirical observations in (Tirinzoni et al., 2025), our findings demonstrate this failure mode. In Figure 2 we report the average zero-shot performance when using an *undirected* exploration strategy, i.e., exploring by uniformly sampling random reward embeddings from the

[1]Remark: Most of the existing works leveraging the FB framework (Touati et al., 2023; Cetin et al., 2025; Pirotta et al., 2024) are offline methods, where ρ is the dataset distribution collected by decoupled unsupervised exploration methods. Our work is in the *online* setting instead, where we do not have access to external datasets and we collect data transitions jointly with the learning of the FB algorithm.

hypersphere. We observe that this strategy suffers from performance degradation and entropy stagnation. Additionally, the recovered policies exhibit severe foot dragging which prevents seamless sim-to-real transfer. To mitigate these issues, we propose to ground the unsupervised policy learning by adding a regularizer. Formally, this leads to the actor loss objective:

$$\mathcal{L}_{\text{FB-Reg}}(\pi) = -\mathbb{E}_{z\sim\nu,\ s\sim\rho,\ a\sim\pi_z(s)}\Big[F(s,a,z)^\top z + \lambda_{\text{reg}}\, Q_{\text{reg}}(s,a)\Big] \tag{6}$$

where Q_{reg} is a critic trained with a behavior regularizer reward. We refer interested readers to the details at Sections F and G. While this modification regularizes the solution manifold by effectively suppressing foot dragging (see Figure 2 (right)), we find that exploration degrades further, resulting in lower entropy and performance. We hypothesize that this is because the behavior-regularized critic assigns higher Q_{reg} values when the robot remains stationary, which further biases the policy toward conservative behaviors and suppresses FB exploration. To tackle these issues, in the next section we introduce FB-MEBE, an algorithmic modification to FB exploration strategies to improve downstream task performance.

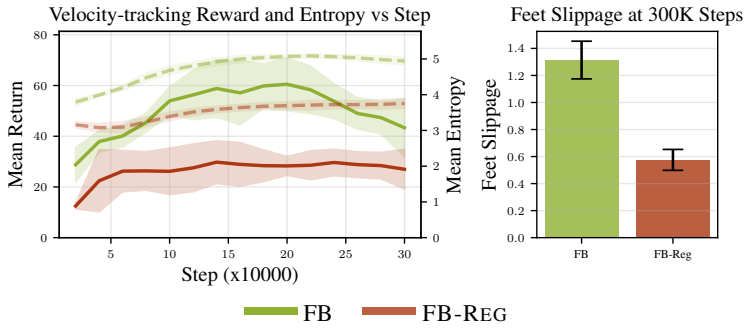

Figure 2: Limitations of undirected exploration: Standard FB algorithm with undirected exploration (FB) suffers from performance degradation and entropy stagnation (left, higher is better) and yields unnatural behaviors (right, lower is better). While introducing a behavior regularizer (FB-Reg) leads to more plausible behaviors (right), it leads to severe performance degradation and further reduces behavior diversity (left).

5 Maximum Entropy Behavior Exploration

5.1 Density-Inverse Exploration for Online FB

How should a zero-shot RL agent explore in an environment in order to accumulate diverse, relevant data, while aligned with downstream task of interest? While we lack a proper characterization of which data is relevant *a priori*, the support of the empirical training distribution ρ must cover the occupancy measures of policies optimal for downstream tasks z_r^d. Hence, an exploration strategy should select intrinsic behaviors z_r^t to bring the agent's historical achieved behavior distribution $p^t_{z_r^a}$ close to the downstream behavior distribution $p_{z_r^d}$. This can be formalized with the following distribution matching objective:

$$J_{\text{oracle}}(p^t_{z_r^a}) = D_{KL}(p_{z_r^d} || p^t_{z_r^a}) \tag{7}$$

where $p^t_{z_r^a}$ is the distribution of achieved behaviors in the replay buffer at time t and D_{KL} is the forward KL divergence (support covering). Unfortunately, in the unsupervised setting, objective in Equation (7) is intractable, because we do not have access to the desired behavior distribution $p_{z_r^d}$. In the absence of a useful inductive bias on the direction where to expand the support of $p_{z_r^a}$, similarly to Pitis et al. (2020), we expand the support on all directions maximizing the entropy of the achieved behavior distribution $H[p^t_{z_r^a}]$. This leads to the following objective:

$$J_{\text{MEBE}}(p^t_{z_r^a}) = -H[p^t_{z_r^a}], \tag{8}$$

implicitly inducing a curriculum on the behaviors with which explore next to effectively push the frontier of achievable behaviors. To optimize Equation (8), at time t we act with an exploration policy π_t^E with reward embedding z_r^E such that

$$z_r^E = \arg\max_{z_r \in \mathcal{Z}} \mathbb{E}_{z'_r \sim q(z'_r|z_r)} H\left[p^t_{z_r^a \cup z'_r}\right]. \tag{9}$$

Importantly, Equation (9) depends on $q(z'_r|z_r)$ which encodes the conditional probability of achieving behavior z'_r if we act with the policy parameterized with z_r, which we do not know. However, since the uniform distribution over a bounded support maximizes entropy, increasing $H[p^t_{z_r^a}]$ encourages an even allocation of probability mass across the behavior space.

Consequently, rather than predicting future outcomes, FB-MEBE steers the achieved behavior distribution towards uniformity by sampling exploration behaviors z_r^E that have the minimum density in the current buffer.

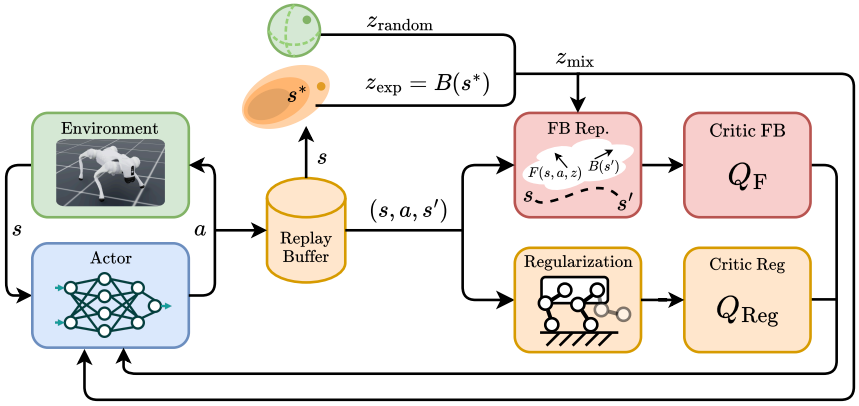

Figure 3: Maximum Entropy Behavior Forward-Backward Exploration: FB-MEBE is an online zero-shot RL algorithm that collects data by acting with behaviors z_{exp} that maximize the entropy of the achieved behavior distribution. Collected data is used to train policies with a regularized loss combining policy improvement objective based on the FB action value function (Q_{FB}) and a critic (Q_{reg}) trained on a behavior regularizer reward to induce meaningful locomotion patterns.

5.2 Practical FB-MEBE

Let $q_\psi(z_r^a)$ be the density model trained on the achieved behaviors stored in the replay buffer. To encourage exploration toward under-visited behaviors, we sample exploration behaviors according to a distribution that is inversely proportional to their estimated density, i.e.,

$$z_r^E \propto (q_\psi(z_r^a) + \epsilon)^{-\beta} \tag{10}$$

where $\epsilon > 0$ ensures numerical stability and $\beta \geq 0$ controls the strength of the inverse reweighting.

In practice, because we have direct access to states, we learn the density model on the space of achieved states $q_\psi(s)$ instead (or on a projection $\varphi(s)$[2] of it). This effectively focuses on goal-reaching behaviors, i.e., $z_r = B(s)$, and could circumvent challenges arising from representation drifts that occur due to B representations changing through time.

Various approaches can be used to estimate the state density, such as Kernel Density Estimation (KDE), Variational Autoencoders (VAE) (Kingma & Welling, 2013), k-NN particle estimators, Random Network Distillation (RND) (Burda et al., 2018) or normalizing flows (Ghugare & Eysenbach,

[2]To focus exploration on relevant parts of the state space, we learn the density on projections of it i.e., $\varphi : S \to S'$, where $S \subseteq \mathbb{R}^d, S' \subseteq \mathbb{R}^m, m < d$.

2025). We chose normalizing flows (NFs) to fit our density $q_\psi(s)$ over all the alternatives due to its efficient likelihood evaluation, compared to quadratic computation cost of non-parametric models, while offering true density estimates compared to surrogates produced by RND or VAE. Leveraging NFs for unsupervised goal sampling has also been explored in (Ghugare & Eysenbach, 2025). Note that during the inverse sampling process, the state s is sampled from the replay buffer rather than from the flow model to improve training stability. Details for the NF can be found in Table 7. Finally, we explore with the set of policies with reward embedding

$$\{z_j^E = B(s_j^E)\}_{j=1}^K, \quad s_j^E \sim (q_\psi^t(s))^{-\beta} \tag{11}$$

where K is the number of exploration policies collecting data in parallel and $q_\psi^t(s)$ is the density model fitted on data from the replay buffer at episode t.

Finally, as reported in Section 4, FB-MEBE regularizes exploration by enforcing relevant gait patterns via a behavior regularizer, as in Equation (6). We provide the pseudocode for our algorithm in Algorithm 1 and a visual schematic of FB-MEBE in Figure 3. For details, see Sections G and J.

Algorithm 1 Maximum entropy behavior exploration (FB-MEBE)

1: **Input:** F_θ and $\bar{F}_\theta$, B_ϕ and $\bar{B}_\phi$, q_ψ, Q_{reg}
2: **while** not converged **do**
3: Sample behavior z_r^E according to eq. (11).
4: Collect data $\mathcal{D}_n = \text{Rollout}(\pi(z_r^E))$.
5: Add data to buffer $\mathcal{D}_{1:n} = \mathcal{D}_{1:n-1} \cup \mathcal{D}_n$.
6: Fit F_θ, B_ϕ, policies π_z, q_ψ and Q_{reg} with $\mathcal{D}_{1:n}$.
7: **end while**

6 Experiments

Our experiments are designed to address the following questions:

i) Does FB-MEBE produce more natural behaviors than unregularized counterparts?

ii) Does FB-MEBE achieve gains in performance comparing with other exploration strategies?

iii) Does FB-MEBE enable effective transfer from simulation to real-world hardware?

Environment. All experiments are conducted in IsaacLab simulation environment (Mittal et al., 2025). We use the Unitree Go2 quadruped robot with 12 degrees of freedom. The action space is defined as $\mathcal{A} \subseteq [-1, +1]^{12}$. The simulation runs at 200 Hz, while the PD controller operates at 50 Hz. Details about state space and domain randomization can be found in Section I.

Task Metrics. We evaluate FB-MEBE on two different sets of downstream tasks: velocity-tracking tasks and base orientation tasks. Both the velocity-tracking and base-orientation settings consists of 17 downstream tasks. Performance is measured by the average episode return achieved on each task, with the episode length equals 250 steps and a maximum achievable reward of 250. Detailed downstream task reward are in appendix Section E. We additionally report the foot sliding to assess locomotion quality (details in Section F).

Baselines. We compare FB-MEBE with several baselines and ablations. FB (Touati & Ollivier, 2021) performs undirected exploration by uniformly sampling random reward embeddings from the hypersphere. FB-Reg uses the same uniform exploration strategy as FB but includes a behavior regularizer (see Section 4). This isolates the performance gains provided by our exploration strategy from the gains provided by the regularizer. FB-MEBE (β=2, our method) includes a behavior regularizer and directs exploration sampling 80% of its embeddings with our proposed MEBE strategy (Equation (11)) and 20% using uniform random embeddings. Finally, we include an ablation

FB-MEBE (β=0), which sets the reverse sampling coefficient β to zero. This can be seen as an instantiation of FB-CPR (Tirinzoni et al., 2025), with no data prior. Instead of prioritizing rare behaviors, it performs uniform exploration over the achieved behaviors. This isolates the performance gains provided by our exploration strategy from the gains provided by the biasing towards goal-reaching behaviors. For other ablations on the β parameter and on sampling strategies, see Sections C.1 and C.2, respectively.

In practice, to focus exploration on relevant parts of the state space, we conduct exploration over a lower-dimensional projections of the state rather than directly operating on the full state space. Formally we define a projection mapping $\varphi : S \to S'$, where $S \subseteq \mathbb{R}^d, S' \subseteq \mathbb{R}^m, m < d$. For velocity-tracking tasks, exploration is performed in the planar base velocity space $[v_x, v_y]$. For base-orientation tasks exploration is performed in the space of the gravity vector expressed in the robot base frame $[g_x, g_y, g_z]$, which characterizes the base orientation. While this restricts the exploration space to task-relevant settings, FB-MEBE could be easily extended to operate directly on the full state-space. However, its performance is bound by the accuracy of density estimators, which inherently degrade in high-dimensional spaces.

6.1 Main Results

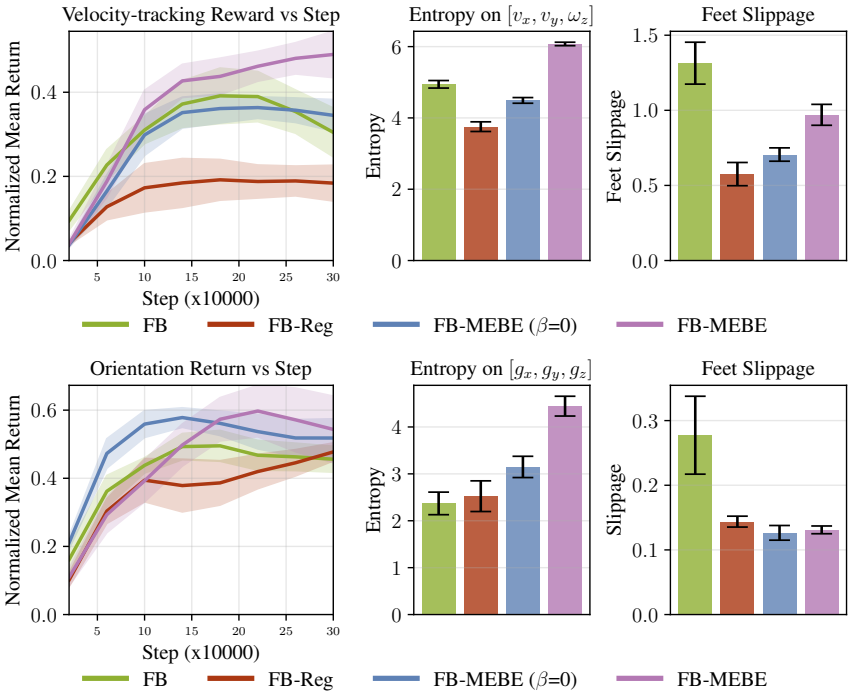

Figure 4: Left: Zero-shot performance averaged over 17 downstream velocity tracking tasks (top) and orientation tasks (bottom). We plot normalized return wrt the mean return of Fast-TD3 (Seo et al., 2025). Results are averaged across 5 random seeds, with shaded regions indicating $\pm$ 1-standard deviation. Middle: Policy entropy on $[v_x, v_y, w_z]$ (top) and $[g_x, g_y, g_z]$ (bottom) measured at 300K training steps. Right: Feet slippage measured at 300K training steps (lower is better).

Overall Performance. In Figure 4, we report results on several downstream tasks. All returns are normalized using the mean return achieved by Fast-TD3 (Seo et al., 2025). Fast-TD3 is a supervised reinforcement learning algorithm trained directly with task-specific rewards, and therefore serves as an approximate upper bound under explicit reward supervision. In contrast, all FB-based methods are trained without access to downstream task rewards and must rely on unsupervised exploration to acquire useful behaviors. FB-MEBE achieves an average performance that is the

highest over all the other baselines for velocity tracking tasks or on par with best performing baseline for orientation tasks. Most notably, it yields the highest entropy among all baselines. This indicates that FB-MEBE expands behavioral coverage during online training. Importantly, this broader exploration translates into improved downstream task performance rather than unstructured behavior diversification. This can be further seen in Figures 5 and 6 where FB-MEBE outperforms all other baselines in extreme downstream tasks. For orientation tasks, FB-MEBE shows lower sample efficiency than FB-MEBE-$\beta = 0$, showcasing that unprioritized goal-reaching exploration can be a valid strategy for some downstream tasks. In contrast, FB and FB-Reg perform the worst among all algorithms, which we attribute to uninformed exploration (FB) and the additional regularization constraints (FB-Reg) that overly restrict exploration, especially in velocity-tracking tasks, leading to limited behavioral coverage and consequently lower average performance. Regarding gait-related metrics, FB-MEBE substantially reduces the feet slide penalty compared to the original FB, indicating that the regularization successfully suppresses unnatural dragging behaviors. Although FB-MEBE exhibits slightly higher feet slippage than FB-Reg, this difference is expected. Since FB-MEBE actively explores higher locomotion velocities, increased horizontal foot velocities naturally lead to higher sliding penalties. Therefore, the observed difference reflects expanded behavioral coverage rather than degraded gait stability.

Detailed Locomotion Performance. Figure 5 reports performance across 17 velocity-tracking tasks. While all methods perform reasonably on moderate velocity commands, the baselines suffer significant degradation on high-velocity tasks. This is consistent with the replay buffer distribution (Section D), where the FB baseline shows a low coverage of high-velocity states. By contrast, FB-MEBE expands the buffer coverage into sparsely visited velocity regions, maintaining strong performance even on high-velocity tasks (see Section E for details).

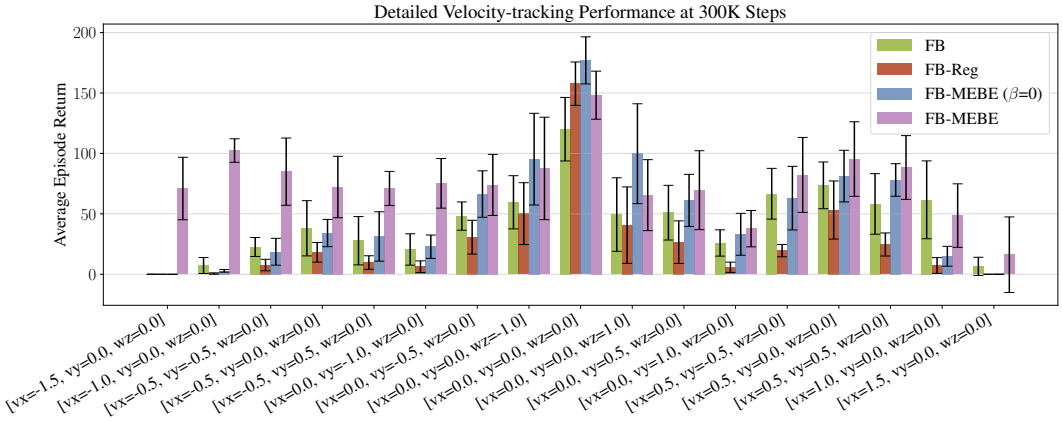

Figure 5: Zero-shot scores for locomotion tasks. Results are averaged over 5 random seeds, and error bars denote $\pm$ 1-standard deviation.

Detailed Orientation Performance. Figure 6 shows the performance across 12 base orientation tracking settings. Notably, FB-MEBE demonstrates superior performance on challenging large-angle tasks compared to baselines. This indicates enhanced exploration within the orientation behavior space. However, we observe that FB-MEBE achieves lower returns than baselines on small-angle orientation tasks, including FB-Reg. We hypothesize that this is because small-angle commands correspond to behaviors near the initial state, which are naturally well-covered by random exploration. FB-MEBE prioritizes less-visited behaviors, allocating more capacity toward exploring extreme orientations, which can reduce its performance on these more common states. FB-Reg does not exhibit this degradation because its regularization primarily constrains foot-slippage behaviors, thereby leaving orientation exploration unaffected.

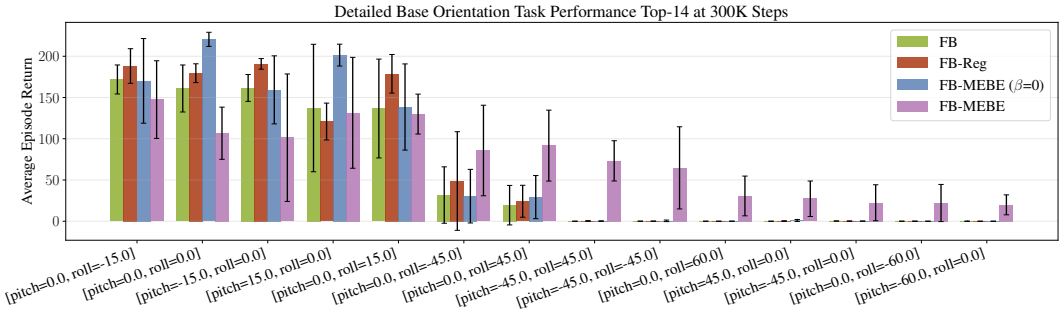

Figure 6: Zero-shot scores for orientation tasks. Results are averaged over 5 random seeds, and error bars denote $\pm$ 1-standard deviation.

Hardware Tests. We deploy FB-MEBE on a real Unitree Go2 robot and evaluate it on both locomotion and orientation control tasks. To facilitate real-world control, commands are issued through a joystick interface, and the robot is guided by a composite reward function (details in Section E) that simultaneously regulates velocity, base orientation, and base height. As illustrated in Figure 7, FB-MEBE enables the robot to track commanded translational velocities, adjust pitch and roll orientations, and regulate its height. The learned policy also supports compositional commands, such as walking forward while maintaining a pitch angle of 15°, or tracking yaw angular velocity while keeping the base height at $0.32\,\mathrm{m}$. Importantly, the resulting behaviors remain stable and natural, without exhibiting excessive action rates or noticeable foot slippage that could destabilize the robot.

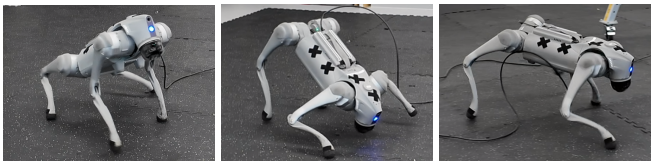

Figure 7: FB-MEBE allows for zero-shot sim2real transfer to the Unitree Go2 quadrupedal robot.

7 Conclusion

We propose FB-MEBE, a practical online zero-shot RL algorithm that enables efficient exploration for learning quadrupedal control. FB-MEBE addresses the undirected exploration of naive online FB algorithm, by using an exploration strategy that guides data collection policies towards maximizing the entropy of the achieved behavior distribution. To constrain exploration, FB-MEBE introduces a behavior regularizer, shaping the recovered policies towards more natural and physically plausible patterns. Through this novel yet simple approach, FB-MEBE yields policies that can be seamlessly deployed to hardware without further finetuning. To the best of our knowledge, this is the first instantiation of a fully online zero-shot RL algorithm on real robotics systems without relying on external datasets as motion priors.

While FB-MEBE represents a principled attempt toward efficient online zero-shot RL, some limitations remain. The recovered policies will struggle with downstream tasks requiring behaviors that fall far outside the collected dataset. Additionally, the exploration strategy is still loosely constrained, which can occasionally lead the agent to explore extreme state configurations that are irrelevant for downstream task of interest. A deeper theoretical understanding of the FB algorithm's components could provide better intuition regarding whether FB implicitly prioritizes some behaviors over others. Finally, while we leveraged a useful behavior regularizer, developing more theoretically sound ways to characterize a specific family of reward functions of interest could lead to safer, more directed exploration in future work.

Broader Impact Statement

Our work focuses on zero-shot RL algorithms for quadrupedal control. While we recognize that autonomous robotic systems have the potential to be deployed for harmful applications, our contributions are fundamental in nature and do not directly target any specific negative use cases.

Acknowledgments

The authors would like to thank Marco Bagatella and Thomas Rupf for their help in reviewing this manuscript, and the Max Planck ETH Center for Learning Systems for supporting Núria Armengol.

References

Siddhant Agarwal, Harshit Sikchi, Peter Stone, and Amy Zhang. Proto successor measure: Representing the behavior space of an rl agent. In *International Conference on Machine Learning*, pp. 566–586. PMLR, 2025.

Akhil Bagaria and George Konidaris. Option discovery using deep skill chaining. In *International Conference on Learning Representations*, 2019.

Marco Bagatella, Matteo Pirotta, Ahmed Touati, Alessandro Lazaric, and Andrea Tirinzoni. Td-jepa: Latent-predictive representations for zero-shot reinforcement learning. *arXiv preprint arXiv:2510.00739*, 2025.

Marco Bagatella, Thomas Rupf, Georg Martius, and Andreas Krause. Soft forward-backward representations for zero-shot reinforcement learning with general utilities. *arXiv preprint arXiv:2602.06769*, 2026.

André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado P van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. *Advances in neural information processing systems*, 30, 2017.

Léonard Blier, Corentin Tallec, and Yann Ollivier. Learning successor states and goal-dependent values: A mathematical viewpoint. *arXiv preprint arXiv:2101.07123*, 2021.

Maksim Bobrin, Ilya Zisman, Alexander Nikulin, Vladislav Kurenkov, and Dmitry Dylov. Zero-shot adaptation of behavioral foundation models to unseen dynamics. *arXiv preprint arXiv:2505.13150*, 2025.

Diana Borsa, Andre Barreto, John Quan, Daniel J Mankowitz, Hado van Hasselt, Remi Munos, David Silver, and Tom Schaul. Universal successor features approximators. In *International Conference on Learning Representations*, 2019.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Edoardo Cetin, Ahmed Touati, and Yann Ollivier. Finer behavioral foundation models via auto-regressive features and advantage weighting. In *Reinforcement Learning Conference*, 2025.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural computation*, 5(4):613–624, 1993.

Leander Diaz-Bone, Marco Bagatella, Jonas Hübotter, and Andreas Krause. Discover: Automated curricula for sparse-reward reinforcement learning. *Advances in Neural Information Processing Systems*, 38:21863–21896, 2026.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. C-learning: Learning to achieve goals via recursive classification. In *International Conference on Learning Representations*, 2021.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, pp. 1515–1528. PMLR, 2018.

Raj Ghugare and Benjamin Eysenbach. Normalizing flows are capable models for rl. *arXiv preprint arXiv:2505.23527*, 2025.

Steven Hansen, Will Dabney, Andre Barreto, David Warde-Farley, Tom Van de Wiele, and Volodymyr Mnih. Fast task inference with variational intrinsic successor features. In *International Conference on Learning Representations*, 2019.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: a vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

Scott Jeen, Tom Bewley, and Jonathan M Cullen. Zero-shot reinforcement learning from low quality data. *Advances in Neural Information Processing Systems*, 37:16894–16942, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, et al. Openvla: An open-source vision-language-action model. In *Conference on Robot Learning*, pp. 2679–2713. PMLR, 2025.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

Yitang Li, Zhengyi Luo, Tonghe Zhang, Cunxi Dai, Anssi Kanervisto, Andrea Tirinzoni, Haoyang Weng, Kris Kitani, Mateusz Guzek, Ahmed Touati, et al. Bfm-zero: A promptable behavioral foundation model for humanoid control using unsupervised reinforcement learning. *arXiv preprint arXiv:2511.04131*, 2025.

Hao Liu and Pieter Abbeel. Aps: Active pretraining with successor features. In *International Conference on Machine Learning*, pp. 6736–6747. PMLR, 2021.

Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and achieving goals via world models. *Advances in Neural Information Processing Systems*, 34:24379–24391, 2021.

Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Munoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, et al. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025.

Seohong Park, Tobias Kreiman, and Sergey Levine. Foundation policies with hilbert representations. In *International Conference on Machine Learning*, pp. 39737–39761. PMLR, 2024a.

Seohong Park, Oleh Rybkin, and Sergey Levine. METRA: Scalable Unsupervised RL with Metric-Aware Abstraction, March 2024b. arXiv:2310.08887.

Matteo Pirotta, Andrea Tirinzoni, Ahmed Touati, Alessandro Lazaric, and Yann Ollivier. Fast imitation via behavior foundation models. In *International Conference on Learning Representations*, volume 2024, pp. 12685–12724, 2024.

Silviu Pitis, Harris Chan, Stephen Zhao, Bradly Stadie, and Jimmy Ba. Maximum entropy gain exploration for long horizon multi-goal reinforcement learning. In *International Conference on Machine Learning*, pp. 7750–7761. PMLR, 2020.

Vitchyr Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. In *International Conference on Machine Learning*, pp. 7783–7792. PMLR, 2020.

Thomas Rupf, Marco Bagatella, Marin Vlastelica, and Andreas Krause. Optimistic task inference for behavior foundation models. *arXiv preprint arXiv:2510.20264*, 2025.

Younggyo Seo, Carmelo Sferrazza, Haoran Geng, Michal Nauman, Zhao-Heng Yin, and Pieter Abbeel. Fasttd3: Simple, fast, and capable reinforcement learning for humanoid control. *arXiv preprint arXiv:2505.22642*, 2025.

Archit Sharma, Michael Ahn, Sergey Levine, Vikash Kumar, Karol Hausman, and Shixiang Gu. Emergent real-world robotic skills via unsupervised off-policy reinforcement learning. *CoRR*, abs/2004.12974, 2020.

Harshit Sikchi, Andrea Tirinzoni, Ahmed Touati, Yingchen Xu, Anssi Kanervisto, Scott Niekum, Amy Zhang, Alessandro Lazaric, and Matteo Pirotta. Fast adaptation with behavioral foundation models. In *Reinforcement Learning Conference*, 2025.

Jingbo Sun, Songjun Tu, Qichao Zhang, Haoran Li, Xin Liu, Yaran Chen, Ke Chen, and Dongbin Zhao. Unsupervised zero-shot reinforcement learning via dual-value forward-backward representation. In *International Conference on Learning Representations*, volume 2025, pp. 25604–25633, 2025.

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

Andrea Tirinzoni, Ahmed Touati, Jesse Farebrother, Mateusz Guzek, Anssi Kanervisto, Yingchen Xu, Alessandro Lazaric, and Matteo Pirotta. Zero-shot whole-body humanoid control via behavioral foundation models. *arXiv preprint arXiv:2504.11054*, 2025.

Ahmed Touati and Yann Ollivier. Learning one representation to optimize all rewards. *Advances in Neural Information Processing Systems*, 34:13–23, 2021.

Ahmed Touati, Jérémy Rapin, and Yann Ollivier. Does zero-shot reinforcement learning exist? In *The Eleventh International Conference on Learning Representations*, 2023.

Núria Armengol Urpí, Marin Vlastelica, Georg Martius, and Stelian Coros. Epistemically-guided forward-backward exploration. In *Reinforcement Learning Conference 2025*, 2025.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Algorithm Details

Following prior work on Forward-Backward (FB) representations (Touati et al., 2023), we include two additional regularization losses to stabilize FB training. First, we adopt an orthonormality regularization on the backward representation $\Sigma_B = \mathbb{E}\left[B(s)B(s)^\top\right]$, which encourages the covariance of $B(s)$ to approach the identity matrix. This regularization prevents the backward representation from collapsing and improves the conditioning of the learned embedding space. Second, we introduce a temporal-difference regularization loss on the forward map that encourages $F(s,a,z)^\top z$ to approximate the action-value function corresponding to the reward induced by $B(s)^\top \Sigma_B^{-1} z$. This loss encourages the forward representation to capture directions in the latent space that are relevant for policy optimization. See (Touati et al., 2023) (Appendix B) for more details.

B Practical Exploration

To improve exploration efficiency in locomotion exploration, we discard states where the robot pitch or roll exceeds $21.5°$ (note that in orientation exploration we keep these states, since we want robot to reach extreme behavior). These states typically correspond to failure events such as falling, which can produce large velocities unrelated to meaningful locomotion behaviors. Filtering such degenerate transitions helps focus exploration on valid locomotion regimes.

C Ablation Tests

C.1 Ablation Study on β.

Figure 8 presents the ablation results with different inverse-density coefficients β. When $\beta = 0$, corresponding to uniform latent sampling without density reweighting, both entropy and task reward remain relatively low. As β increases, entropy and reward consistently improve.

These results confirm that inverse-density reweighting effectively increases the entropy of the achieved behavior distribution and mitigates replay buffer occupancy collapse during online training.

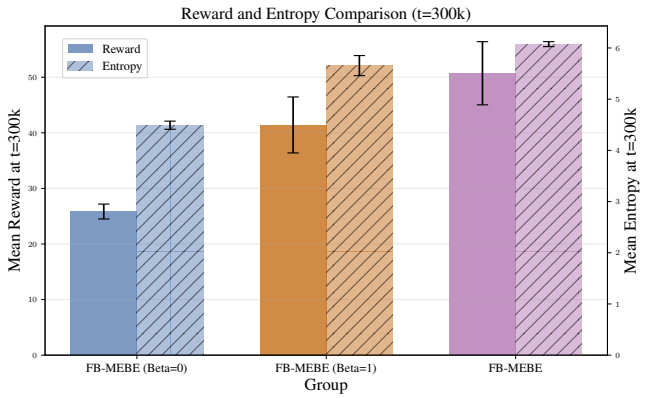

Figure 8: Comparison between different betas on FB-MEBE on locomotion tasks

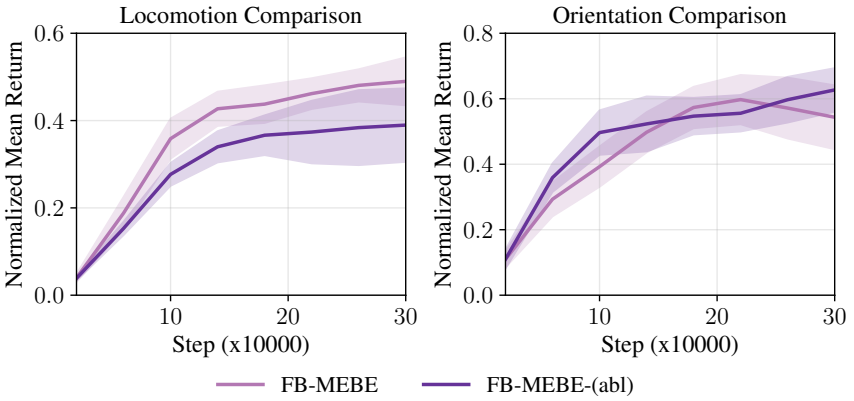

Figure 9: Comparison on velocity-tracking tasks between FB-MEBE and an ablation FB-MEBE-abl leveraging different sampling strategies for training the FB algorithm

C.2 Ablation study on sampling strategy

We evaluate a version of FB-MEBE (FB-MEBE-abl), that performs an ablation on the sampling strategy used for *training* the FB representations. FB-MEBE uses the *same* sampling strategy for reward embeddings used for data collection ($0.8z_{exp}$ + $0.2z_{random}$) and the reward embedding used for training FB representations. Here, we compare with an alternative version which only uses our strategy for data collection, and uses $(0.8z_{\text{goal reaching}} + 0.2z_{\text{random}}$ for training. The later is analogous to the exploration strategy FB-MEBE($\beta = 0$) uses. While FB-MEBE puts more emphasis on learning FB representations and policies for low visitation behaviors and might hence learn better policies useful for exploration,FB-MEBE-abl samples exploration behaviors uniformly from the replay buffer. While these two approaches are different in nature, in practice we do not find significant differences as can be seen in Figure 9. Extensive evaluation on different mixing ratios can also bring more insights on better sampling alternatives.

D Replay buffer data distribution

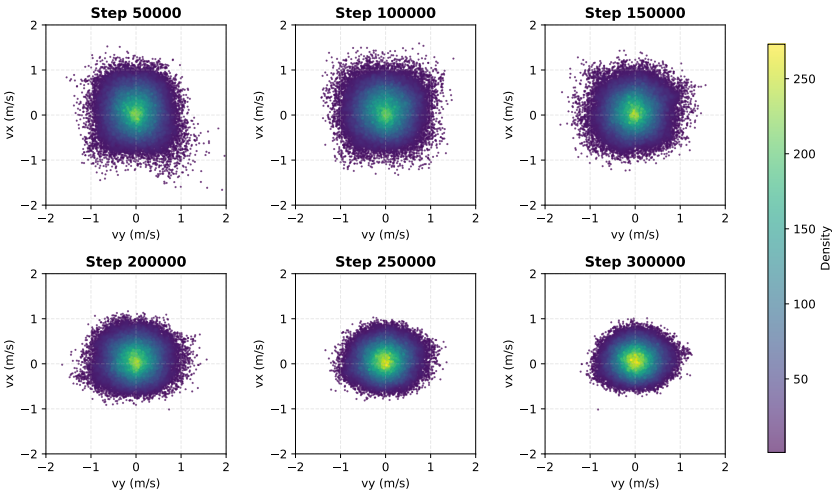

Figure 10: Velocity $[v_x, v_y]$ distribution of FB replay buffer at different training step. The color represents the density of data points.

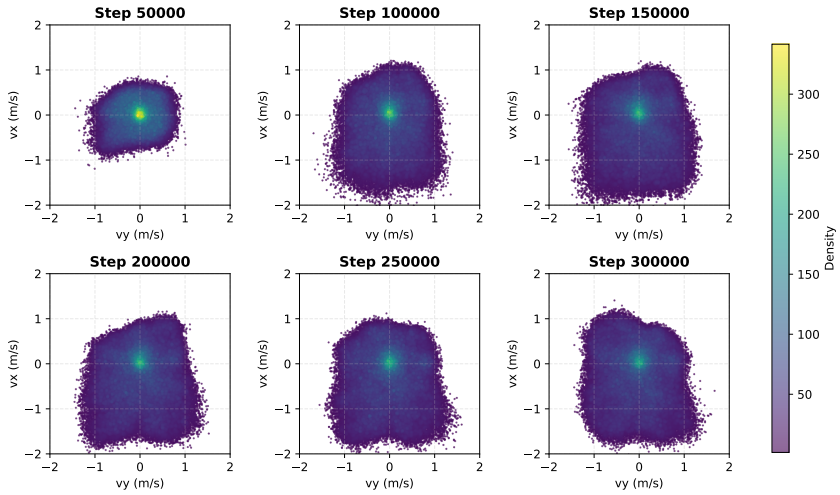

Figure 11: Velocity $[v_x, v_y]$ distribution of FB-MEBE replay buffer at different training step. The color represents the density of data points.

E Task Reward Function

Let $\mathbf{v} \in \mathbb{R}^3$ and $\boldsymbol{\omega}_z \in \mathbb{R}$ denote the current base linear velocity and yaw angular velocity, respectively, $h \in \mathbb{R}$ denote the current base height and let $\mathbf{g} \in \mathbb{R}^3$ denote the projected gravity vector. The corresponding command targets are denoted by $\mathbf{v}^*$, ω_z^*, h^*, and $\mathbf{g}^*$.

We define the tracking errors as

$$e_v = \|\mathbf{v} - \mathbf{v}^*\|_2, \tag{12}$$
$$e_\omega = |\omega_z - \omega_z^*|, \tag{13}$$
$$e_g = \|\mathbf{g} - \mathbf{g}^*\|_2 \tag{14}$$
$$e_h = \|h - h^*\|_2. \tag{15}$$

The individual reward terms are computed using Gaussian-shaped functions:

$$r_v = \exp\left(-\left(\frac{e_v}{0.3}\right)^2\right), \tag{16}$$
$$r_\omega = \exp\left(-\left(\frac{e_\omega}{0.2}\right)^2\right), \tag{17}$$
$$r_g = \exp\left(-\left(\frac{e_g}{0.1}\right)^2\right), \tag{18}$$
$$r_h = \exp\left(-\left(\frac{e_h}{0.05}\right)^2\right). \tag{19}$$

The locomotion and orientation task reward is defined as follows:

$$r_{\text{locomotion}} = r_v \cdot r_\omega \cdot r_g \tag{20}$$
$$r_{\text{orientation}} = r_g \cdot r_h \tag{21}$$

The reward for sim-to-real transfer is defined by multiplication of all terms:

$$r_{\text{sim2real}} = r_v \cdot r_\omega \cdot r_g \cdot r_h \tag{22}$$

F Regularization rewards

We provide the mathematical definitions of the regularization reward. Let N_j denote the number of actuated joints and N_f the number of feet. At time step t, let $\ddot{\mathbf{q}}_t \in \mathbb{R}^{N_j}$ be the joint acceleration vector, $\mathbf{a}_t \in \mathbb{R}^{N_j}$ be the action vector, and $\mathbf{a}_{t-1}$ be the previous action. For each foot $i \in \{1, \dots, N_f\}$, let $h_{t,i}$ be its vertical position (world frame), and $\mathbf{v}_{i,t}^{xy} \in \mathbb{R}^2$ be its horizontal linear velocity (world frame).

Joint acceleration penalty. The joint-acceleration penalty is defined as the squared ℓ_2 norm of the joint acceleration vector:

$$r_{\text{joint_acc}} = \|\ddot{\mathbf{q}}_t\|_2^2 = \sum_{k=1}^{N_j} \left(\ddot{q}_{t,k}\right)^2 . \tag{23}$$

Action rate penalty. The action-rate penalty discourages abrupt changes between consecutive actions:

$$r_{\text{action_rate}} = \|\mathbf{a}_t - \mathbf{a}_{t-1}\|_2^2 = \sum_{k=1}^{N_j} \left(a_{t,k} - a_{t-1,k}\right)^2 . \tag{24}$$

Feet slippage. The feet-slide penalty penalizes horizontal foot motion during ground contact. A binary contact indicator $c_{t,i}$ is defined as

$$c_{t,i} = \begin{cases} 1, & \text{if } h_{t,i} > h_{\text{feet_contact}}, \\ 0, & \text{otherwise.} \end{cases} \tag{25}$$

$$r_{\text{feet_slide}} = \sum_{i=1}^{N_f} c_{t,i} \, \|\mathbf{v}_{t,i}^{xy}\|_2. \tag{26}$$

Total typical regularization reward. In our experiments, the regularization reward used to train the critic is:

$$r_{\text{reg}} = -2.5 \times 10^{-7} \, r_{\text{joint_acc}} - 0.1 \, r_{\text{action_rate}} - 0.1 \, r_{\text{feet_slide}} \tag{27}$$

G Regularized Exploration.

We use critic $Q_{\text{reg}}(s, a)$ to estimate the long-term return of the regularization reward r_{reg} defined in Section F. Given tuples $(s, a, r_{\text{reg}}, s')$ sampled from the replay buffer, we train the regularization critic by minimizing a standard temporal-difference (TD) loss:

$$\mathcal{L}_{\text{critic}}(Q_{\text{reg}}) = \mathbb{E}_{(s,a,s')\sim\rho, a'\sim\pi_z(s')} \Big[\big(Q_{\text{reg}}(s,a) - \big(r_{\text{reg}} + \gamma \, \bar{Q}_{\text{reg}}(s', a') \big) \big)^2 \Big] \tag{28}$$

Therefore, to pretrain FB agent and learn good gait at the same time, we have the following actor loss:

$$\mathcal{L}_{\text{actor}}(\pi) = -\mathbb{E}_{z\sim\nu, \, s\sim\rho, \, a\sim\pi_z(s)} \Big[Q_{\text{FB}}(s, a, z) + \lambda_{\text{reg}} \, Q_{\text{reg}}(s, a) \Big] \tag{29}$$

H Simulation Setup

Joint Control We employ joint position control as the low-level control interface. The joint configuration of the robot in the standing pose is defined as the nominal joint position q_{nom}. The policy outputs joint position offsets a, which are scaled by a factor s and added to the nominal configuration to obtain the target joint positions

$$q_{target} = q_{nom} + s \cdot a. \tag{30}$$

The target joint positions are tracked using a PD controller

$$\tau = K_p(q_{target} - q) + K_d(\dot{q}_{target} - \dot{q}), \tag{31}$$

where q and $\dot{q}$ denote the current joint positions and velocities.

Table 1: Joint Control Parameters

Parameter	**Value**
Action Scale	0.5
Proportional Gain K_p	25.0
Derivative Gain K_d	0.5
Control Frequency [Hz]	50Hz

Table 2: Domain Randomization

Parameter	**Range**
Friction Coefficient	$\mathcal{U}([0.5,\ 1.5])$
Base COM Offset [m]	$\mathcal{U}([-0.05,\ 0.05])$
Base Mass perturbation [kg]	$\mathcal{U}([-2.0,\ 2.0])$
Link Mass Offset [m]	$\mathcal{U}([-0.01,\ 0.01])$
Link Mass perturbation [kg]	$\mathcal{U}([-0.2,\ 0.2])$
Default Joint Pos [rad]	$\mathcal{U}([-0.3,\ 0.3])$

Table 3: Additive Observation Noise

Observation	**Range**
v [m/s]	$\mathcal{U}([-0.1,\ 0.1])$
ω [rad/s]	$\mathcal{U}([-0.2,\ 0.2])$
grav	$\mathcal{U}([-0.05,\ 0.05])$
q [rad]	$\mathcal{U}([-0.01,\ 0.01])$
$\dot{q}$ [rad/s]	$\mathcal{U}([-0.15,\ 0.15])$

Domain Randomization and Observation Noise

I Observation Space

Network	Obs Dimension Breakdown
Critic (F)	$[\mathbf{v}_3, \boldsymbol{\omega}_3, \mathbf{g}_3, h_{\text{base},1}, \mathbf{q}_{12}, \dot{\mathbf{q}}_{12}]$
B	$[\mathbf{v}_3, \boldsymbol{\omega}_3, \mathbf{g}_3, h_{\text{base},1}]$
Actor	$[\mathbf{v}_3, \boldsymbol{\omega}_3, \mathbf{g}_3, \mathbf{q}_{12}, \dot{\mathbf{q}}_{12}, \mathbf{a}_{12}]$
Critic (Reg)	$[\mathbf{v}_3, \boldsymbol{\omega}_3, \mathbf{g}_3, h_{\text{base},1}, \mathbf{q}_{12}, \dot{\mathbf{q}}_{12}, \mathbf{a}_{12}, h_{\text{feet},4}, F_{\text{feet},4}]$

Table 4: Inputs for different networks

The subscripts represent the dimension of this variable. $\mathbf{v}$ is the base linear velocity, $\boldsymbol{\omega}$ is the base angular velocity, $\mathbf{g}$ is the projected gravity vector, h_{base} is the base height, $\mathbf{q}$ and $\dot{\mathbf{q}}$ are joint positions and joint velocities, respectively, $\mathbf{a}$ is the last action, h_{foot} denotes foot heights (4 feet), and F_{feet} denotes foot contact forces (4 feet). *.

Note on B mapping: We note that the B representation acts on a lower-dimensional projection of the full state space instead. When dealing with high dimensionality environments, learning future probabilities for all states is difficult and generally requires large d to accommodate for all possible rewards. In general, we are often interested in rewards that depend not on the full state but on a projection of it. If this is known in advance, the representation B can be trained on that part of the state only, with same theoretical guarantees (Appendix, Theorem 4 (Touati & Ollivier, 2021)). Hence, we leverage an environment-dependent feature map $\vartheta : S \to G$, and learn $B(g)$ instead of $B(s)$, where $g = \vartheta(s)$. Importantly, rewards can be arbitrary functions of g. This was also suggested in (Touati & Ollivier, 2021).

J Hyperparameters

Hyperparameter	**Value**
Number of training steps	150k
Number of parallel environments	2048
Number of rollout steps between each agent update	10
Number of gradient steps per agent update	10
Policy delay (TD3 actor update interval)	2
Number of initial steps with random actions	1000
Number of samples for task inference	100k
Replay buffer size	4M
Batch size	4096
Discount factor	0.98
z update step interval	100
z dimension d	50
z distribution ν	
-uniform on unit sphere	20%
-goals from online buffer	80%
Coefficient for orthonormality loss	100
Coefficient for Fz-regularization loss	0.1
Regularization coefficient λ_{reg}	20

Table 5: Training hyperparameters

Hyperparameter	**Critic** (F)	**B**	**Critic** (**Reg**)	**Actor**
Learning Rate	$5 \cdot 10^{-5}$	$5 \cdot 10^{-5}$	$5 \cdot 10^{-5}$	$5 \cdot 10^{-5}$
Soft update coefficient	0.01	0.01	0.005	0.01
Input Variables	(s, a, z)	(s)	(s, a)	(s, z)
Output Dim	d	d	1	12
Embedding Hidden Units	1024	–	–	1024
Feed Forward Hidden Layers	2	2	4	2
Feed Forward Hidden Units	1024	512	1024	1024
Activations	ReLU	ReLU	ReLU	ReLU
Number of Parallel Networks	2	1	2	1

Table 6: Network hyperparameters

Table 7: Hyperparameters of the Normalizing Flow.

Hyperparameter	Value
Flow architecture	RealNVP-style normalizing flow
Number of coupling layers	10
Hidden dimension	64
Activation function	ReLU
Scale output activation	Tanh
Masking strategy	Alternating checkerboard / channel mask
Base distribution	Standard Gaussian $\mathcal{N}(0, I)$
Learning rate	$1 \cdot 10^{-3}$
Batch size	256
Training epochs	30
Goal buffer size	10k

Table 8: Hyperparameters of reverse sampling.

Hyperparameter	Value
normalizing flow update frequency	1000 steps
sample buffer size	100k
ϵ (for stability)	0.1
β (reverse sampling strength)	2.0

K Connection to Universal Successor Features

Given a feature map $\phi : \mathcal{S} \to \mathbb{R}^d$ that embeds states into d-dimensional features, the successor features (Barreto et al., 2017) for policy π are defined as $\Phi^\pi(s,a) = \mathbb{E}\big[\sum_{t\geq 0} \gamma^t \phi(s_{t+1}) | s, a, \pi\big]$. Then for any reward function that can be expressed as a linear combination of the features, i.e., $r(s) = z^\top \phi(s)$, where $z \in \mathbb{R}^d$ are the weights, the Q-function can be expressed as: $Q_r^\pi(s,a) = z^\top \Phi^\pi(s,a)$. Universal successor features (USF) (Borsa et al., 2019), goes one step further and allows not only zero-shot policy evaluation, but a generic framework for zero-shot policy optimization. This is achieved by learning a family of policies $\{\pi_z\}$ parameterized by a latent vector $z \in \mathbb{R}^d$, such that for each z, the policy π_z is trained to be optimal with respect to reward function $r_z := z^\top \phi$. Formally,

$$\mathbb{E}\Big[\sum_{t\geq 0} \gamma^t \phi(s_{t+1}) | s, a, \pi\Big], \quad \pi_z(s) = \arg\max_a \Phi(s,a,z)^\top z. \tag{32}$$

There exist several instantiations of USF (Agarwal et al., 2025; Park et al., 2024a) (see (Touati et al., 2023) for an extensive list), however, they requires a separate methods for learning features ϕ and one for learning its successor features, usually via a TD-learning approach. In contrast, the forward backward representation (Touati & Ollivier, 2021) provides an alternative framework for zero-shot RL that jointly learns the features and its successor features, via a contrastive learning approach.

FB representations are closely related to successor features. In fact, $F(s,a,z)$ is the successor feature of policy π_z of the basic features $\phi(s) = \mathrm{Cov}_\rho(B)^{-1} B(s)$ (Touati et al., 2023), where $\mathrm{Cov}_\rho B = \mathbb{E}_{\tilde{s}\sim\rho}\big[B(\tilde{s})B(\tilde{s})^\top\big]$.

After training, given a reward function r specified at test time, we can find the task embedding z_r parameterizing the optimal learnt policy for reward r, by performing a simple linear regression of r onto the span of the learnt features B. Namely we find the weights z_r^* (coordinates of the feature basis) that minimize $z_r^* = \arg\min_z \mathbb{E}_{s\sim\rho}\big[\big((r(s) - z_r^\top \phi(s)\big)^2\big] = \mathrm{Cov}_\rho(\phi)^{-1}\mathbb{E}_{\tilde{s}\sim\rho}\big[\phi(s)r(s)\big]$. For FB, the reward embedding can be computed simply via $z_r = \mathbb{E}_{s\sim\rho}\big[B(s)r(s)\big]$ (Touati & Ollivier,

2021), where the expectation is approximated through sampling from the dataset distribution used for training (or a subset).