ADAPTIVE BEHAVIORAL FOUNDATION MODELS: CONTINUAL TRAINING FROM REWARD-FREE EXPERIENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Behavioral foundation models (BFMs) learn general-purpose policies and successor features from reward-free offline data, enabling agents to retrieve a policy for a given reward function without additional learning or planning. However, this policy can fail when test-time dynamics differ from those seen during pretraining. Memory-based approaches address this mismatch through in-context learning without updating model parameters. We study reward-free continual training for adaptation to unseen dynamics within a limited online interaction budget. We introduce AdaBFM, which frames pretraining as learning an adaptive prior distribution over test-time dynamics. At test time, it adapts this prior through continual fine-tuning on reward-free online experience. The zero-shot policy is conditioned on causal memory and a belief over successor ensemble members, allowing it to adapt as evidence accumulates, while successor-belief information gain guides continual training. We evaluate AdaBFM across an aggregate of 23 navigation, locomotion, and manipulation tasks under unseen dynamics. AdaBFM achieves the highest mean performance in 9/23 tasks in the zero-shot setting and 16/23 after continual training, improving by an average of 28% on harder tasks[1].

1 INTRODUCTION

Inspired by the success of self-supervised pretraining in language and vision (Devlin et al., 2019; Caron et al., 2021; Brown et al., 2020; Assran et al., 2023), unsupervised reinforcement learning (RL) seeks to learn transferable representations and behaviors from reward-free interaction data (Barreto et al., 2017; Borsa et al., 2019; Hansen et al., 2019; Eysenbach et al., 2018; Liu & Abbeel, 2021; Park et al., 2024b). In robotics, pretraining on large datasets such as DROID (Khazatsky et al., 2024) provides a similar starting point for learning reusable behaviors. Behavioral foundation models (BFMs) extend this paradigm to zero-shot RL, inferring a target skill and retrieving its corresponding policy without further training given a reward function or a reward-labeled dataset (Touati & Ollivier, 2021; Pirotta et al., 2024; Park et al., 2024a; Agarwal et al., 2025). This provides a natural framework for separating task specification from dynamics adaptation.

However, keeping pretrained representations *actionable* requires adapting behavior when deployment dynamics differ from those represented in offline training data. Changes in layout, terrain, or actuator response can alter the consequences of an action even when the desired task remains unchanged, motivating adaptation to target-domain dynamics (Xie et al., 2026). Reusing offline experience under such shifts requires updating how the agent acts under test-time dynamics. We consider tasks specified once through a small reward-labeled dataset, analogous to a user providing demonstrations of the desired outcome, and study rapid adaptation to unseen dynamics through reward-free online interaction.

While BFMs enable zero-shot policy inference, current approaches remain static and cannot handle shifts in test-time dynamics. The same state-action pair can lead to different future occupancies across environments, making predictions ambiguous when offline datasets mixes dynamics. Memory-based BFMs address this through in-context learning (ICL) (Jeen et al., 2025; Bobrin et al., 2025), but two key challenges remain. First, ICL requires broad data coverage, and extending context length

[1]We provide qualitative videos and code at anonymous.4open.science/w/AdaBFM/ and anonymous.4open.science/r/AdaBFM/, respectively.

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

alone does not guarantee effective utilization of additional evidence, as observed in long-context language models (Liu et al., 2023). Second, conditioning on collected history does not inherently determine which actions would distinguish between counterfactual explanations of that history. These challenges motivate organizing memory, exploration, and continual training jointly. Here, memory summarizes observed transitions, exploration collects evidence that distinguishes dynamics, and continual training uses this evidence to finetune pretrained successor and policy representations beyond zero-shot BFMs.

How can we efficiently adapt pretrained BFMs to unseen dynamics?

We introduce AdaBFM, which frames pretraining as learning an adaptive prior over test-time dynamics. It combines causal memory with a belief over a successor-feature ensemble, enabling adaptation through both in-context inference and continual training. Successor-belief information gain guides online exploration, and the collected reward-free transitions finetune AdaBFM.

We evaluate AdaBFM on navigation, locomotion, and dexterous manipulation tasks under unseen target dynamics, leveraging off-dynamics benchmarks from ODRL (Lyu et al., 2024) after offline pretraining on nominal dynamics. Across 23 aggregate comparisons under unseen dynamics, AdaBFM achieves the highest mean score in 9 tasks zero-shot and 16 after continual training. Across the four benchmark summaries (Fig. 3), reward-free continual training increases AdaBFM's aggregate score by 28%, outperforming state-of-the-art memory based baselines MetaBFM (Grigsby et al., 2026) (20%) and GRU-TD-JEPA (Jeen et al., 2025; Bagatella et al., 2025) (10%).

2 Related work

Adaptation in BFMs and RL. Successor features (Barreto et al., 2017; Borsa et al., 2019) enable transfer across tasks in RL by decoupling environment dynamics from reward specifications. Building on this line of work, BFMs learn a family of latent-conditioned policies through unsupervised pretraining, enabling specific behaviors to be retrieved at test time given a reward function (Touati & Ollivier, 2021; Sikchi et al., 2025). A common protocol consists of offline pretraining, task inference from reward-labeled data, and zero-shot deployment. Table 1 summarizes extensions targeting either novel rewards or unseen environment dynamics. Adaptation to novel tasks primarily uses online rewards to infer or optimize test-time tasks through latent-space (skills) search or parameter updates (Sikchi et al., 2025; Kim et al., 2025; Olivieri et al., 2026; Rupf et al., 2025). We instead study reward-free continual training under unseen dynamics.

Adaptation of BFMs to novel dynamics has largely been studied through in-context inference within zero-shot policies. For instance, Belief-FB (Bobrin et al., 2025) adopts a non-causal formulation, conditioning its successor model on episodic dynamics context. Memory-based BFMs (Jeen et al., 2025) instead infer context causally as the interaction history unfolds. While both approaches summarize observed transitions, they do not explicitly maintain counterfactual explanations, actively gather evidence to resolve representation ambiguity, or update pretrained parameters. Beyond in-context learning, model-based RL methods such as AdaJEPA (Wang et al., 2026) perform continual training of world-model parameters alongside test-time replanning to adapt to novel dynamics. AdaBFM bridges this gap by framing pretraining as learning an adaptive prior distribution over

Method	Unseen env	Online feedback	Adaptation mechanism and target
LoLA (Sikchi et al., 2025)	Reward	States + rewards	RL in latent z-space
ReLA (Sikchi et al., 2025)	Reward	States + rewards	Residual RL updates critic and actor
U2O (Kim et al., 2025)	Reward	States + rewards	RL updates actor and critic with mixed replay
OpTI-BFM (Rupf et al., 2025)	Reward	States + rewards	Optimistic search with a posterior over z
Continual BFM (Olivieri et al., 2026)	Reward	States + rewards	CT updates BFM parameters with coverage-preserving replay
MetaBFM (Grigsby et al., 2026)	Reward	Transitions + rewards	Memory updates c_t; optimistic search updates the posterior over z
Belief-FB (Bobrin et al., 2025)	Dynamics	Offline test trajectories	Non-causal in-context inference of episodic context c
Memory-based BFMs (Jeen et al., 2025)	Dynamics	Reward-free transitions	Causal in-context updates to history state h_t
MetaBFM (Grigsby et al., 2026)	Dynamics	Reward-free transitions	In-context updates to c_t with k-shot exploration
AdaBFM (ours)	Dynamics	Reward-free transitions	Memory and CT; updates context, successor ensemble, and actor

Table 1: **Overview of test-time adaptation methods for BFMs.** Unseen env distinguishes changes in reward or dynamics. CT denotes continual training; mixed replay combines offline and online data.

test-time dynamics and using continual training to update its successor and policy representations directly to resolve ambiguity within a few short episodes (see Fig. 2(c)).

Ensembles provide practical proxies for epistemic uncertainty across various RL paradigms. For instance, FBEE (Urpí et al., 2025) leverages ensemble disagreement to guide exploration during online pretraining of BFMs. APE-V (Ghosh et al., 2022) uses Q-function ensembles to train Bayes-adaptive policies for offline RL, while NeuBay (Ni et al., 2026), an offline MBRL approach, employs world-model ensembles for adaptation through background planning and in-context learning. Similarly, AdaBFM uses ensembles to model an adaptive prior over dynamics, performing reward-free online continual training to rapidly adapt to unseen dynamics by selecting behaviors that maximize successor-belief information gain (Table 2).

3 Problem Setup and Preliminaries

Successor measures and features. We consider a reward-free Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, p_0, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces, $P(\mathrm{d}s'|s,a)$ is the transition kernel, p_0 is the initial-state distribution, and $\gamma \in [0,1)$ is the discount factor. A Markov policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ induces trajectories by sampling $s_0 \sim p_0$, $a_t \sim \pi(\cdot|s_t)$, and $s_{t+1} \sim P(\cdot|s_t, a_t)$.

For a policy π, its successor measure (Dayan, 1993; Blier et al., 2021) $M^\pi(\cdot \mid s,a)$ is the unnormalized discounted future-state occupancy measure: for any measurable set $\mathcal{X} \subseteq \mathcal{S}$, $M^\pi(\mathcal{X} \mid s,a) = \sum_{t\geq 0} \gamma^t \Pr(s_{t+1} \in \mathcal{X} \mid s_0 = s, a_0 = a, \pi)$. Given a reward function $R : \mathcal{S} \to \mathbb{R}$, the action-value decomposes as $Q_R^\pi(s,a) = \int_{\mathcal{S}} R(s^+) M^\pi(\mathrm{d}s^+ \mid s,a)$, separating dynamics induced by the policy from the reward. Successor features (Barreto et al., 2017; Borsa et al., 2019) summarize this measure through a feature map $\psi : \mathcal{S} \to \mathbb{R}^d$ by accumulating features rather than individual state visitations: $F_\psi^\pi(s,a) = \mathbb{E}_\pi[\sum_{t\geq 0} \gamma^t \psi(s_{t+1}) | s_0 = s, a_0 = a]$. For rewards in the feature span, $R_z(s) = \psi(s)^\top z$, where $z \in \mathbb{R}^d$ is the task coordinate, the value is $Q_{R_z}^\pi(s,a) = F_\psi^\pi(s,a)^\top z$. Successor features inherit the Bellman equation (Barreto et al., 2017):

$$F_\psi^\pi(s,a) = \mathbb{E}_{s' \sim P(\cdot|s,a), a' \sim \pi(\cdot|s')} \left[\psi(s') + \gamma F_\psi^\pi(s', a') \right]. \tag{1}$$

Behavior Foundation Model (BFM). BFMs (Touati & Ollivier, 2021; Park et al., 2024a; Agarwal et al., 2025) follow pretraining-then-adaptation paradigm. In the pretraining stage, BFM learns a family of latent-conditioned policies $\{\pi_z(a|s) \mid z \in \mathcal{Z} \subseteq \mathbb{R}^d\}$ from a *reward-free* offline dataset $\mathcal{D} = \{(s,a,s')\}$, with each z indexing the reward function $R_z(s) = \psi(s)^\top z$. At test time, the task coordinate z^* is specified by reward-labeled samples $\mathcal{D}_{\mathrm{rwd}} = \{(s,r)\}$, by optimizing $z^* = \arg\min_z \mathbb{E}_{(s,r)\sim\mathcal{D}_{\mathrm{rwd}}}[(r - \psi(s)^\top z)^2]$. In the classical form of BFMs, known as *zero-shot* RL (Touati & Ollivier, 2021), the agent then directly deploys π_{z^*} without any adaptation. In this paper, we consider **test-time adaptation** (TTA) problem: the agent need to adapt to unseen dynamics through various adaptation forms, such as in-context learning and parameter updates.

TD-JEPA. TD-JEPA (Bagatella et al., 2025), a recent BFM pretraining method which we build upon, learns a state encoder ϕ, a task encoder ψ, and a policy-conditioned predictor $T_\phi(\phi(s), a, z)$ to approximate $F_\psi^{\pi_z}(s,a)$ where z specifies the policy π_z. TD-JEPA learns this predictor off-policy by minimizing the successor-feature Bellman error:

$$\mathcal{L}_{\mathrm{TDJ}}(\phi, T_\phi, \psi) = \mathbb{E}_{\substack{(s,a,s')\sim\mathcal{D},\ z\sim p(z) \\ a'\sim\pi_z(\cdot|s')}} \left[\left\| T_\phi(\phi(s), a, z) - \overline{\psi(s')} - \gamma \overline{T_\phi(\phi(s'), a', z)} \right\|_2^2 \right]. \tag{2}$$

where $\overline{\psi(\cdot)}$ is the target network. The reciprocal loss $\mathcal{L}_{\mathrm{TDJ}}(\psi, T_\psi, \phi)$ is obtained by exchanging the roles of ϕ and ψ. The scalar $Q_{\mathrm{SF}}(s,a,z) = T_\phi(\phi(s), a, z)^\top z$ serves as the critic, and the actor minimizes $\mathcal{L}_{\pi_z} = -\mathbb{E}_{s\sim\mathcal{D}, z\sim p(z), a\sim\pi_z(\cdot|s)}[Q_{\mathrm{SF}}(s,a,z)]$, encouraging each π_z to optimize R_z.

Bayesian RL for BFM. The pretraining data may combine offline datasets collected from multiple environments by unknown behavior policies. Rather than identifying a single transition model from this heterogeneous dataset, the Bayesian principle treats the true deployment MDP $\mathcal{M}^*$ as a random variable and maintains a dataset posterior $\Pr_{\mathcal{D}}(\mathcal{M})$ over plausible models (Strens, 2000). After observing the test-time reward-free history $H_t = (s_0, a_0, s_1, \ldots, s_t)$, this posterior is updated as $\Pr(\mathcal{M}|\mathcal{D}, H_t) \propto \Pr(H_t|\mathcal{M}) \Pr_{\mathcal{D}}(\mathcal{M})$. Consequently, instead of optimizing return under a single estimated model, the Bayesian objective averages over the current posterior:

$$\max_\pi J_{\mathrm{Bayes}}(\pi \mid H_t) := \max_\pi \mathbb{E}_{\mathcal{M}\sim\Pr(\cdot|\mathcal{D},H_t)}[J(\pi, \mathcal{M} \mid H_t)], \tag{3}$$

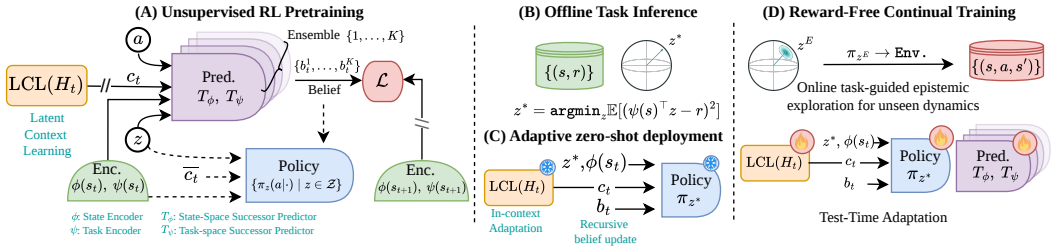


Figure 1: **Method overview. (A)** AdaBFM is pretrained on reward-free offline data to learn a causal context encoder, a successor-feature ensemble, and latent-conditioned policies. **(B)** Given a reward-labeled dataset, the pretrained BFM infers the downstream task coordinate z^* and retrieves the corresponding policy. **(C)** To handle unseen dynamics during zero-shot deployment, AdaBFM uses in-context inference and recursive belief updates without parameter updates. **(D)** During continual test-time training, successor-belief information gain selects task-compatible exploratory behaviors for data collection. The resulting transitions update the context encoder, successor model, and policy, while the task encoder and task coordinate remain fixed.

where $J(\pi, \mathcal{M} \mid H_t)$ is the expected remaining discounted return under the specified downstream reward, and π may adapt to future observations. Since $\mathcal{M}^*$ is hidden and the history provides information about it, deployment forms an epistemic partially observable Markov decision process (POMDP), or equivalently a Bayes-adaptive Markov decision process (BAMDP), in which actions affect both task return and what the agent learns about the current dynamics.

4 AdaBFM

We propose AdaBFM, a behavioral foundation model that learns an adaptive dynamics prior using causal memory and successor beliefs, enabling reward-free continual training in unseen environments. Figure 1 illustrates its components and adaptation stages. Algorithms 1 and 2 summarize the offline pretraining and reward-free online adaptation procedures, respectively.

4.1 Memory-based BFM

To adapt to varying environment dynamics, AdaBFM compresses the interaction history into a learned, timestep-level representation. Following self-supervised representation-learning approaches that infer predictive context from unlabeled trajectories (Oord et al., 2018), AdaBFM adopts latent-context-learning (LCL) (Grigsby et al., 2026). Given interaction history H_t (potentially spanning multiple episodes), LCL infers a causal context c_t summarizing the active dynamics. We condition both the policy and successor predictor on this context, $\pi_\theta(a \mid s, c_t, z)$ and $T_\phi(\phi(s), c_t, a, z)$, allowing their predictions to evolve as interaction evidence accumulates. LCL and the BFM are trained jointly, but gradients from the policy and successor-feature objectives are stopped at c_t ($\overline{c_t}$); consequently, the LCL encoder is updated solely through its self-supervised dynamics objective (Appendix D).

4.2 Approximate posterior estimation

While memory-based BFMs can infer active dynamics from past transitions, a single deterministic context estimate cannot explicitly retain counterfactual explanations or quantify remaining uncertainty. Consequently, the agent risks acting on an averaged or prematurely selected context, leaving it trapped in local sub-optima (Figures 2(b) and 11).

Motivated by the Bayesian objective eq. 3, a model-based agent would maintain a posterior over dynamics and update it via transition likelihoods. However, explicit likelihoods are unavailable in model-free BFMs, which predict successor features rather than low-level state transitions. Following Ghosh et al. (2022), we use an ensemble to approximate the induced posterior over action values, employing Bellman consistency as a surrogate likelihood. We extend their scalar value ensemble and TD residuals to vector-valued successor predictors. A normalized belief b_t over K bootstrapped predictors approximates the induced posterior $\Pr(T_\phi \mid \mathcal{D}, H_t)$ over successor predictors for π_z, yielding the Bayesian value approximation:

$$Q^{\pi_z}_{\text{Bayes},z}(H_t, a) \approx \mathbb{E}_{T_\phi \sim \Pr(\cdot|\mathcal{D},H_t)} \left[T_\phi(\phi_t, c_t, a, z)\right]^\top z \approx \sum_{k=1}^{K} b_{t,k} T_{\phi,k}(\phi_t, c_t, a, z)^\top z. \quad (4)$$

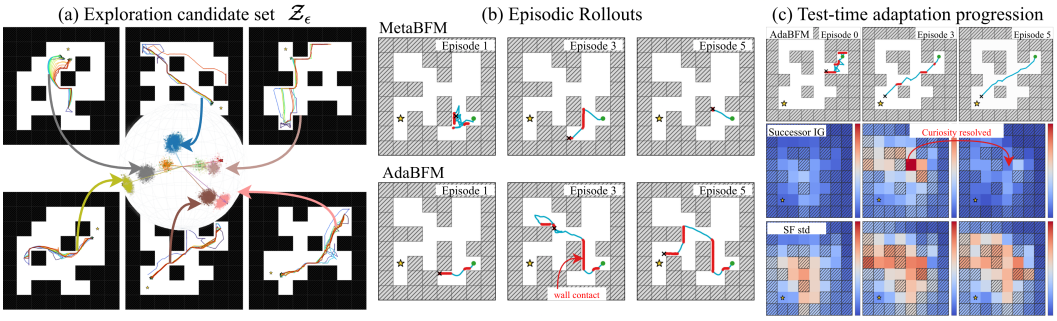


Figure 2: **(a)** Online replay-buffer trajectories. **(b)** Episodic rollouts across test-time adaptation (TTA) iterations on the same unseen maze. MetaBFM remains trapped in a local region, whereas AdaBFM probes task-relevant walls. **(c)** Test-time adaptation progression. Curiosity decreases once the local region is explored and the model is updated; successor-measure disagreement remains dynamic and results in task-irrelevant exploration.

To update b_t, we evaluate each ensemble member's Bellman residual in task-feature space:

$$\delta_{k,t} = T_{\phi,k}(\phi_t, c_t, a_t, z_t) - \left[\overline{\psi(s_{t+1})} + \gamma(1-d_t)\overline{T_{\phi,k}(\phi_{t+1}, c_{t+1}, a'_{t+1}, z_t)}\right], \quad D_{k,t} = \delta_{k,t}^\top C_\psi \delta_{k,t}, \tag{5}$$

where z_t is the behavior used to collect the transition, $d_t \in \{0,1\}$ the episode termination, and a'_{t+1} is the target-policy action. The matrix $C_\psi \approx \mathbb{E}_{x\sim\rho_{\text{ref}}}[\psi(x)\psi(x)^\top]$ weights residual directions according to their contribution to scalar feature readouts under a reference state distribution. Lower $D_{k,t}$ indicates greater compatibility with the observed transition. We recursively update the belief weights via

$$b_{t+1,k} = \frac{b_{t,k}\exp[-\eta D_{k,t}/(2\sigma^2)]}{\sum_{j=1}^K b_{t,j}\exp[-\eta D_{j,t}/(2\sigma^2)]}, \tag{6}$$

where η/σ^2 controls the update sharpness. Because this residual operates on task features rather than scalar environment rewards, belief updates require only reward-free transitions.

Following Ghosh et al. (2022), we train the belief-conditioned policy $\pi_\theta(a \mid s_t, c_t, b_t, z)$ to maximize the belief-weighted critic Q_b by minimizing the loss $\mathcal{L}_\pi = -\mathbb{E}[Q_b(\phi_t, c_t, a, z)]$, where $a \sim \pi_\theta(\cdot \mid s_t, c_t, b_t, z)$. During zero-shot deployment, network parameters remain fixed, while dynamic updates to c_t and b_t allow the policy to adapt as environment evidence accumulates.

4.3 Reward-free continual training

At deployment, the downstream task coordinate z^* is given, but the environment dynamics remain unknown. Fine-tuning pretrained models using standard online RL algorithms(i.e., TD3 (Fujimoto et al., 2018)) typically requires reward feedback and risks catastrophic unlearning under tight interaction budgets (Zhou et al., 2025; Sikchi et al., 2025) (Appendix C).

The BFM factorization provides a reward-free alternative. The fixed coordinate z^* specifies the task, while the successor predictors encode policy-dependent future occupancies and serve as a world-model proxy. We therefore freeze z^* and the task encoder, updating only the context encoder, successor ensemble, and policy using reward-free transitions. To gather informative data without deviating from the pretrained behavior manifold, we explore the latent policy space and filter candidate behaviors based on their predicted value under z^*.

Unconstrained exploration driven solely by successor disagreement can direct the agent toward uncertain, task-irrelevant regions (Fig. 2(c)). To prevent this, we sample N candidate exploration coordinates $\{z_i^E\}_{i=1}^N$. For each candidate z_i^E, we draw an action $a_i \sim \pi_\theta(\cdot|s_t, c_t, b_t, z_i^E)$ and evaluate its predicted value under the target task coordinate z^*:

$$\widetilde{Q}_i = \sum_{k=1}^K b_{t,k} T_{\phi,k}(\phi_t, c_t, a_i, z_i^E)^\top z^*, \qquad \mathcal{Z}_\epsilon = \left\{ z_i^E \;\middle|\; \widetilde{Q}_i \geq \max_{j\in\{1,\dots,N\}} \widetilde{Q}_j - \epsilon \right\}. \tag{7}$$

Here, $\widetilde{Q}_i$ estimates the task value of the exploratory candidate z_i^E. The set $\mathcal{Z}_\epsilon$ filters this pool, retaining only the behaviors whose predicted values lie within ϵ of the highest-value candidate. We then rank this task-compatible subset by its expected reduction in successor-belief entropy.

Algorithm 1: Reward-free pretraining.

Require: Dataset $\mathcal{D}$, ensemble size K, context len W

Initialize networks and target networks; $b_0 \leftarrow \mathbf{1}/K$
while not converged **do**
  $H_t, (s_t, a_t, s_{t+1}) \sim \mathcal{D}, z \sim p(z)$
  $(c_t) \leftarrow \mathrm{LCL}(H_t)$
  $\mathcal{L}_{\mathrm{LCL}} \leftarrow \mathcal{L}_{\mathrm{LDM}} + \mathcal{L}_{\mathrm{IDM}} + \mathcal{L}_{\mathrm{FDM}}$ (eq. 10-eq. 12)
  Compute $\delta_{k,t}$ and update b_{t+1} (eq. 5-eq. 6)
  Compute $\mathcal{L}_{\mathrm{TDJ}}$ (Sec. 3)
  $\mathcal{L}_\pi \leftarrow -\mathbb{E}_{a\sim\pi_\theta}[Q_b(\phi_t, c_t, a, z)] + \lambda_{\mathrm{BC}}\mathcal{L}_{\text{Flow-BC}}$ (eq. 4 & 15)
  $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{LCL}} + \mathcal{L}_{\mathrm{TDJ}} + \mathcal{L}_\pi$
  backward($\mathcal{L}$); update target networks

Algorithm 2: Test-time training (reward-free).

Require: Policy π_θ, task coordinate z^*, budget B

Freeze z^* (Fig. 1(B)); $b_0 \leftarrow \mathbf{1}/K$; $\mathcal{D}_{\mathrm{on}} \leftarrow \{\}$
for $t = 0, \ldots, B-1$ **do**
  if resampling exploration behavior z^E **then**
    $z_t^E \leftarrow$ eq. 9
  $a_t \sim \pi_\theta(\cdot \mid s_t, c_t, b_t, z_t^E)$
  Observe (s_{t+1}, d_t); $H_{t+1} \leftarrow (H_t, a_t, s_{t+1})$
  $(c_{t+1}) \leftarrow \mathrm{LCL}(H_{t+1})$
  Update b_{t+1} using eq. 6
  $\mathcal{D}_{\mathrm{on}} \leftarrow \mathcal{D}_{\mathrm{on}} \cup \{(s_t, a_t, s_{t+1}, d_t)\}$
  if $|\mathcal{D}_{\mathrm{on}}| \geq W$ **then**
    Sample $H_t, (s_t, a_t, s_{t+1}, d_t) \sim \mathcal{D}_{\mathrm{on}}$
    Compute Algorithm 1 losses with (ψ, z^*) fixed
    backward($\mathcal{L}$); update target networks

Let $\mu_{k,i} = T_{\phi,k}(\phi_t, c_t, a_i, z_i^E)$ denote ensemble member k's successor prediction for candidate i. Treating this prediction as a hypothetical outcome, we apply the compatibility rule from eq. 6 to compute the counterfactual belief for member j:

$$b_{t+1,j}^{(k,i)} \propto b_{t,j} \exp\left[-\frac{\eta}{2\sigma^2}(\mu_{j,i} - \mu_{k,i})^\top C_\psi (\mu_{j,i} - \mu_{k,i})\right], \tag{8}$$

normalized across j. We then select the exploration coordinate from $\mathcal{Z}_\epsilon$ that maximizes the expected information gain over the successor belief:

$$\mathrm{IG}_{\mathrm{succ}}(z_i^E; b_t) = \mathcal{H}(b_t) - \sum_{k=1}^{K} b_{t,k} \mathcal{H}\left(b_{t+1}^{(k,i)}\right), \qquad z_t^E = \arg\max_{z_i^E \in \mathcal{Z}_\epsilon} \mathrm{IG}_{\mathrm{succ}}(z_i^E; b_t), \tag{9}$$

where $\mathcal{H}$ denotes entropy. This objective identifies the behavior expected to yield the greatest one-step reduction in successor uncertainty while remaining task-compatible. The agent then executes z_t^E to collect a reward-free transition for continual training.

5 Experimental Results

Our aim to study how continual training improves the adaptation of BFMs to unseen dynamics. We conduct extensive experiments across multiple baselines and ablations to evaluate the proposed design. Specifically, we ask: *(i)* does adaptive-prior pretraining improve zero-shot transfer (Sec 5.1); *(ii)* can continual training recover performance under unseen dynamics and, if so, which explore-exploit strategy most effectively improves upon zero-shot performance during TTA (Sec 5.2)?

Experimental setup. We evaluate offline-to-online adaptation: agents are pretrained on offline data (nominal dynamics), the pretraining buffer is unavailable at deployment, and they receive a limited online interaction budget. We use the following benchmark environments:

- *Navigation.* PointMaze (`PM-randN`, where N is the number of training layouts) contains procedurally generated maze layouts. We pretrain on offline data collected by a random policy over 1, 3, 10, or 30 training layouts and evaluate on 20 held-out layouts with *easy*, *medium*, and *hard* configurations, yielding 60 test configurations. This setting tests adaptation from low-quality, high-coverage offline data. We additionally consider D4RL (Fu et al., 2020) medium and large `AntMaze` layouts, evaluating five tasks on six unseen layouts per maze size from ODRL (Lyu et al., 2024), yielding another 60 test configurations. This setting tests adaptation with more limited offline coverage and diversity.
- *Locomotion.* We pretrain `HalfCheetah` and `Ant` on D4RL medium-quality data collected under nominal dynamics (Fu et al., 2020) and evaluate under morphology (3) and friction (3) shifts per environment (Lyu et al., 2024), totaling 12 test configurations.
- *Dexterous manipulation.* We pretrain Adroit Pen (Rajeswaran et al., 2017) on offline data and evaluate under broken-joint (3) and shrunk-finger (3) shifts (Lyu et al., 2024), totaling six test configurations. These changes may require new behaviors, motivating comparison of reward-free and reward-based continual training. Additional details are in Appendix A.

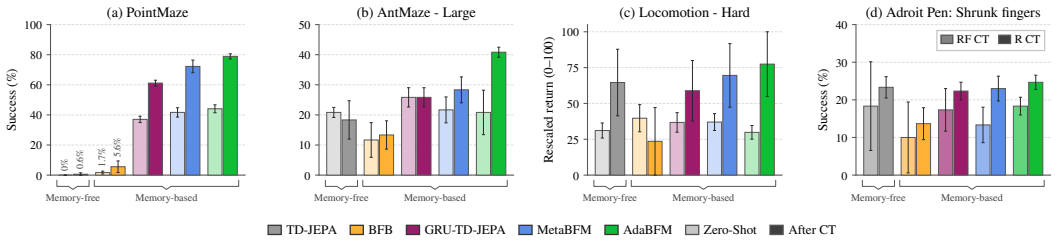


Figure 3: **Adaptation to unseen dynamics.** Performance of zero-shot and continually trained policies across baselines, averaged over **(a)** all `PM-rand30` tasks, **(b)** `AntMaze-Large` tasks, **(c)** four hard locomotion domains, and **(d)** three shrunk-finger variants of `Adroit Pen`. Locomotion returns are shown on a normalized scale. Continual training (CT) is reward-free unless indicated otherwise, with RF CT and R CT denoting reward-free and reward-based continual training in `Adroit Pen`.

Baselines. We compare against TD-JEPA (Bagatella et al., 2025), a state-of-the-art *memoryless* offline BFM. From this, we adapt GRU-TD-JEPA, a recurrent history-conditioned variant (Jeen et al., 2025), where the recurrent memory is updated directly through the BFM successor objective. We also evaluate BFB (Bobrin et al., 2025), which infers an episode-level (fixed) non-causal context[2], and MetaBFM (Grigsby et al., 2026), which uses LCL for timestep-level context inference decoupled from the BFM losses. AdaBFM extends this step-level inference by inferring recursive belief. Unless stated otherwise, *adaptation* denotes reward-free online continual training with gradient updates, while *zero-shot adaptation* relies solely on in-context or recursive-belief inference (see App. A.4).

Evaluation and TTA setup. After reward-free pretraining, we infer the downstream task coordinate z^* from offline reward-labeled samples (Fig. 1(B)). For maze tasks, we generate these samples by relabeling pretraining states with the target goal. Methods are first evaluated zero-shot in Sec 5.1, then trained continually on reward-free online transitions within a fixed interaction budget in Sec 5.2. Results are averaged over 10 evaluation seeds across 4 pretraining seeds (Appendix A.5).

5.1 Does adaptive pretraining improve zero-shot transfer?

Fig. 3**(a–c)** compares zero-shot performance across navigation and hard locomotion shifts, while Table 3**(b)** presents the corresponding results for Adroit Pen. We highlight key observations below.

Memoryless vs Memory-based. Memoryless BFMs struggle on navigation tasks, particularly when pretrained across diverse dynamics, which can be attributed to latent representation interference as also observed in (Bobrin et al., 2025). Memory-based methods achieve substantially higher average zero-shot success in this setting. However, this advantage is less consistent in locomotion, where subsequent actions can often compensate for previous errors, making trajectories easily recoverable. In contrast, in navigation tasks where an agent may become stuck in corridor (Fig. 2B).

Successor belief. Beyond MetaBFM's context encoding, AdaBFM recursively reweights successor hypotheses using observed transition consistency during zero-shot deployment. This allows its policy adapt to unseen dynamics without parameter updates and contribute to its stronger navigation performance. The ensemble-size ablation further examines this contribution (Table 9).

Memory trained through successor learning. Causal LCL's self-supervised dynamics objectives capture how actions influence future states, providing useful information for adapting to changed navigation layouts and locomotion dynamics. In contrast, GRU-TD-JEPA achieves the strongest zero-shot performance on `Adroit Pen`, outperforming AdaBFM and MetaBFM, whose contexts are trained separately from the successor objective. Manipulation benefits more from memory optimized directly for successor prediction. Such memory supports in-context adaptation and self-correction under actuator noise, consistent with findings in (Tarasov et al., 2025).

Need for continual training. Zero-shot performance remains limited on difficult navigation tasks and severe morphology and friction shifts, motivating online updates to adapt predictions and control to unseen dynamics (Appendices B.1 and B.2). We next study online exploration strategies and evaluate reward-free continual training.

[2]The official BFB implementation assumes access to test-layout trajectories. We remove this assumption, so our scores are not directly comparable to the original work.

(a) Exploration Strategy

Rule	AUC ↑	Success ↑ (Δ)
Task only	30.21±1.94	44.17±3.54 (+29.17)
Random feasible	29.92±4.48	47.50±3.54 (+32.50)
SF disagreement	29.17±4.60	46.67±2.36 (+31.67)
LCL error	30.17±2.00	45.00±2.36 (+30.00)
Successor IG	**40.50**±4.83	**56.70**±2.90 (+41.70)

(b) Parameter group finetuned

Scope	AUC ↑	Success ↑
z_π only	13.69±4.03	32.22±5.09
+ LCL only	7.06±2.25	33.89±3.85
+ Actor/critic only	29.61±1.75	48.33±1.67
Full adaptation (all)	**40.50**±4.83	**56.70**±2.90

Table 2: **AdaBFM ablations on PM-rand hard task.** (a) Exploring for epistemic uncertainty and (b) parameters updated online.

5.2 Does continual training improve adaptation?

Before evaluating test-time adaptation (TTA), we first establish an effective online exploration strategy, as exploration choices strongly dictate performance under tight interaction budgets.

Exploration strategies for sample-efficient adaptation. To evaluate these components in AdaBFM, we fix the pretrained weights, interaction budget, and fine-tuning parameter groups while varying only the online collection rule. We compare five distinct strategies. *(i) Task-only* executes the downstream policy by freezing z^*. *(ii) Random-feasible* samples evidence goals uniformly within the feasible set $z_t^E \sim \mathrm{Unif}(\mathcal{Z}_\epsilon)$ defined in eq. 7. *(iii) Successor disagreement* selects the candidate with the highest ensemble dispersion in feature space. *(iv) LCL prediction error* (Grigsby et al., 2026) selects the candidate with the highest predicted dynamics error $\widehat{\mathcal{L}}_{\mathrm{LCL}}(z_i^E)$. *(v) Successor-belief information gain (belief IG)* (AdaBFM) selects the candidate with the largest expected belief-entropy reduction eq. 9. Except for task-only collection, all strategies rank candidates within the task-compatible set $\mathcal{Z}_\epsilon$.[3] Tables 2**(a)** and 10 report results on exploration strategy on `PM-rand` and locomotion.

In maze environments, *belief IG* outperforms other exploration strategies by 10%. Because agents easily become locally trapped in mazes, *belief IG* probes regions where ensemble members disagree and resolves environmental ambiguity through continual training (Fig. 2**(c)**). In contrast, naive disagreement acts as a purely dynamic intrinsic motivation, distracting the agent from the task. In locomotion, where task diversity is more constrained and primarily focused on forward motion, simpler strategies such as *task-only* collection also perform well.

Regularization in continual training. Latent task inference z_r depends on the reward dataset (Sec 3). We conduct a constrained search within the skill space to collect online experience that reduces belief entropy, as moving off this task manifold causes unlearning (Sikchi et al., 2025) (see Appendix C).

Fine-tuning parameter groups. Using successor-belief information gain for exploration, we evaluate which parameter groups to fine-tune during online adaptation. All configurations perform an exploratory search over candidate latents $z^E \in \mathcal{Z}_\epsilon$ (maximizing expected belief-entropy reduction) and feed z^E to the policy $\pi(a \mid s, z^E)$ for environment rollouts. We compare four parameter group configurations: *(i)* z^E *search only* optimizes only the exploratory behavior latent z^E while keeping all network weights frozen (Sikchi et al., 2025). *(ii)* $+$ *LCL only* fine-tunes the LCL context encoder alongside z^E search. *(iii)* $+$ *Actor/critic only* fine-tunes the actor and successor critic alongside z^E search while leaving the context encoder frozen. *(iv) Full adaptation* fine-tunes all parameter groups jointly (z^E, LCL context encoder, actor, and successor critic). We observe that adapting *all* actor-critic parameters yields the largest improvement (Table 2**(b)**). Updating successor features refines both the world representation and the policy. Consequently, all subsequent continual training experiments combine successor-belief IG with full adaptation over short reward-free episodes.

Continual training results. Across 23 aggregate tasks (5 navigation, 12 locomotion, and 6 manipulation), AdaBFM achieves the highest mean score in 16 following continual training. While continual training improves all baselines across the four benchmark summaries (Fig. 3), reward-free adaptation increases AdaBFM's aggregate score by 28% relative to zero-shot performance, compared to 20% for MetaBFM and 10% for GRU-TD-JEPA.

Detailed domain-specific breakdowns support these trends. Table 8 reports PointMaze continual training results and zero-shot gains, while Table 13 presents evaluations on AntMaze.

[3] Restricting candidates to $\mathcal{Z}_\epsilon$ utilizes the known task coordinate without requiring online reward observations, avoiding hand-crafted exploration schedules.

(a) PointMaze (offline training layouts)

Method	1	10	30
TD-JEPA	2.4±0.8 (+2.4)	1.1±0.6 (+1.1)	0.6±1.0 (+0.6)
GRU-TD-JEPA	51.7±4.4 (+30.0)	56.7±1.7 (+25.7)	61.1±1.9 (+24.1)
MetaBFM	**61.1**±5.1 **(+48.7)**	72.8±3.5 (+32.6)	72.2±4.2 (+30.5)
AdaBFM	52.2±6.7 (+41.7)	**76.7**±5.0 **(+33.7)**	**78.9**±1.7 **(+34.8)**

(b) Adroit Pen (broken joints)

Method	ZS	RF CT	R CT
TD-JEPA	12.5±3.5	12.5±10.6	32.0±1.4
GRU-TD-JEPA	**32.5**±10.6	**27.5**±16.3	29.0±7.1
BFB	5.0±1.4	2.5±3.5	23.5±3.5
MetaBFM	7.0±8.5	8.0±8.5	30.0±4.2
AdaBFM	6.5±6.4	22.5±17.7	**37.0**±8.5

Table 3: **Continual training under unseen dynamics.** Success (%) on PointMaze (60 configurations) and Adroit Pen (easy and medium broken joints, 2 configurations). Parentheses show gains over zero-shot in percentage points. ZS, RF CT, and R CT denote zero-shot, reward-free CT, and reward-based CT.

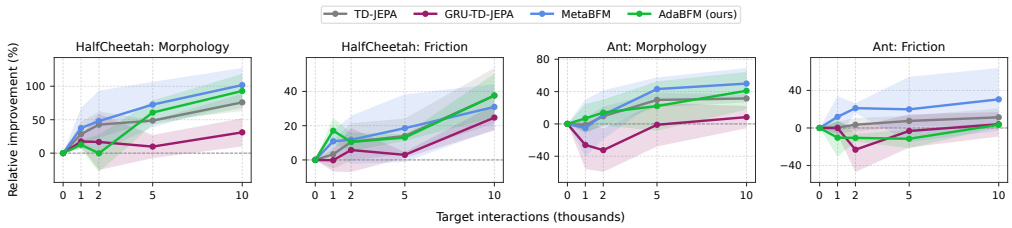


Figure 4: **Training progression of locomotion tasks under dynamics shifts.** Test-time adaptation progression evaluated over reward-free online transitions (see Tables 11 and 12, as well as Fig. 10 for full table).

Locomotion's continual training progression is shown in Fig. 4 alongside Tables 11 and 12, and Table 14 summarizes performance on Adroit Pen.

Continual training on hard tasks. Hard locomotion restricts joint motion or body size, while hard PointMaze requires longer paths. On these challenging shifts, reward-free continual training improves AdaBFM's score by 36% over zero-shot performance, compared to 26% for MetaBFM and 18% for GRU-TD-JEPA. Visualizing successor representations in PointMaze (Figures 11 and 12) reveals that AdaBFM maintains temporal consistency. In contrast, MetaBFM remains overly optimistic about path lengths, while TD-JEPA representations collapse entirely.

Continual training on Adroit. Severe kinematic shifts invalidate offline-inferred skills, requiring novel online behaviors. While memory-based approaches like GRU-TD-JEPA leverage successor learning for self-correction, AdaBFM uses binary online rewards to perform optimistic task inference (Rupf et al., 2025). This refines the task estimate alongside the world model, resolving complete policy failures on `Adroit` (Table 14, Appendix C).

6 Discussion

Reframing BFMs as adaptive Bayesian priors over environment dynamics, AdaBFM extends their role beyond zero-shot execution. Using ensembles of successor predictors as world-model proxies bridges representation learning and active exploration, allowing agents to continuously refine their world model during interaction without relying on test-time planning.

Conclusion. We present AdaBFM, a framework for continual training of behavior foundation models from reward-free online experience. By modeling BFMs as history-dependent adaptive priors through a causal context encoder, a belief-weighted successor feature ensemble, and task-compatible successor-belief information gain, AdaBFM addresses unseen environment dynamics. Across navigation and locomotion tasks, AdaBFM achieves superior zero-shot transfer and test-time adaptation, demonstrating that active, belief-guided exploration significantly accelerates online fine-tuning under tight interaction budgets.

Limitations and future work. Our primary formulation isolates adaptation to novel dynamics by assuming a known downstream task inferred from offline reward data. When the offline dataset provides insufficient coverage of the downstream goal region, this assumption can limit performance. Additionally, successor ensembles offer only a finite proxy for the true dynamics posterior. Developing parameter-efficient posterior approximations and extending belief-guided adaptation to high-dimensional visual domains present promising avenues for future research.

AI USE STATEMENT

In this work, we used generative AI tools for tasks with required disclosure, specifically for debugging or editing baseline code implementations, to enable GPU/ batch parallization for speed. We have not used generative AI tools for other tasks with required disclosure, such as generating synthetic datasets, developing conceptual frameworks, proposing hypotheses, designing experiments, or interpreting results. Tasks such as formulating mathematical claims and assisting in writing proofs are not applicable to this work. Additionally, we used generative AI tools for tasks with recommended disclosure, specifically for language polishing and formatting results tables, videos and plots. We have not used generative AI tools for literature surveys, brainstorming, or generating research ideas. We have reviewed all AI-assisted work, and all LLM-assisted code optimizations were verified and tested for correctness. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, we include our codebase in the supplementary material. We also present computational cost analyses in Appendix D.4, dataset descriptions in Appendix A, and implementation details in Appendix D.

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.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629. IEEE, 2023.

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.

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.

Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. A tutorial on meta-reinforcement learning. *Foundations and Trends in Machine Learning*, 18(2-3):224–384, 2025.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF international conference on computer vision (ICCV)*, pp. 9630–9640. IEEE, 2021.

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

Pim De Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. *Advances in neural information processing systems*, 32, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019. URL https://api.semanticscholar.org/CorpusID:52967399.

Ron Dorfman, Idan Shenfeld, and Aviv Tamar. Offline meta reinforcement learning–identifiability challenges and effective data collection strategies. *Advances in Neural Information Processing Systems*, 34:4607–4618, 2021.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. Pmlr, 2018.

Dibya Ghosh, Anurag Ajay, Pulkit Agrawal, and Sergey Levine. Offline rl policies should be trained to be adaptive. In *International Conference on Machine Learning*, pp. 7513–7530. PMLR, 2022.

Dibya Ghosh, Chethan Anand Bhateja, and Sergey Levine. Reinforcement learning from passive data via latent intentions. In *International Conference on Machine Learning*, pp. 11321–11339. PMLR, 2023.

Jake Grigsby, Siddhant Agarwal, Yu Lei, Leonidas Varveropoulos, and Yuke Zhu. Meta-reinforcement learning with zero-shot reinforcement learning. In *Third Reinforcement Learning Beyond Rewards Workshop at the Reinforcement Learning Conference*, 2026. URL https://openreview.net/forum?id=XyGJJ4FPoX.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

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.

Pranaya Jajoo, Harshit Sikchi, Siddhant Agarwal, Amy Zhang, Scott Niekum, and Martha White. Regularized latent dynamics prediction is a strong baseline for behavioral foundation models. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jdL6WB5jHZ.

Scott Jeen, Tom Bewley, and Jonathan Cullen. Zero-shot reinforcement learning under partial observability. In *Reinforcement Learning Conference*, 2025. URL https://openreview.net/forum?id=2JuMpSfJ4D.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Junsu Kim, Seohong Park, and Sergey Levine. Unsupervised-to-online reinforcement learning, 2025. URL https://openreview.net/forum?id=YGhV8wQv3C.

Daniel Lawson, Adriana Hugessen, Charlotte Cloutier, Glen Berseth, and Khimya Khetarpal. Self-predictive representations for combinatorial generalization in behavioral cloning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=FkeURAdA0h.

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

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. arxiv 2023. *arXiv preprint arXiv:2307.03172*, 2023.

Jiafei Lyu, Kang Xu, Jiacheng Xu, Mengbei Yan, Jing-Wen Yang, Zongzhang Zhang, Chenjia Bai, Zongqing Lu, and Xiu Li. ODRL: A benchmark for off-dynamics reinforcement learning. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024. URL https://openreview.net/forum?id=ap4x1kArGy.

Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent model-free RL can be a strong baseline for many POMDPs. In *International Conference on Machine Learning*, pp. 16691–16723. PMLR, 2022.

Tianwei Ni, Esther Derman, Vineet Jain, Vincent Taboga, Siamak Ravanbakhsh, and Pierre-Luc Bacon. Long-horizon model-based offline reinforcement learning without explicit conservatism. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=NHu6haXTM9.

Pierriccardo Olivieri, Siddhant Agarwal, and Amy Zhang. Continual reinforcement learning via behavioral foundation models. In *Continual Reinforcement Learning Workshop at RLC 2026*, 2026. URL https://openreview.net/forum?id=cv8kqddPwR.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

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.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025a.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=KVf2SFL1pi.

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.

Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

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

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.

Malcolm JA Strens. A bayesian framework for reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pp. 943–950, 2000.

Denis Tarasov, Alexander Nikulin, Ilya Zisman, Albina Klepach, Andrei Polubarov, Nikita Lyubaykin, Alexander Derevyagin, Igor Kiselev, and Vladislav Kurenkov. Yes, q-learning helps offline in-context rl. *arXiv preprint arXiv:2502.17666*, 2025.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, 2012.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

Ying Wang, Oumayma Bounou, Yann LeCun, and Mengye Ren. Adajepa: An adaptive latent world model. *arXiv preprint arXiv:2606.32026*, 2026.

Chuan Wen, Jierui Lin, Trevor Darrell, Dinesh Jayaraman, and Yang Gao. Fighting copycat agents in behavioral cloning from observation histories. *Advances in Neural Information Processing Systems*, 33:2564–2575, 2020.

Yifan Wu, George Tucker, and Ofir Nachum. The laplacian in rl: Learning representations with efficient approximations. *arXiv preprint arXiv:1810.04586*, 2018.

Angchen Xie, Nikhil Sobanbabu, Ishayu Shikhare, Alan Wang, Max Simchowitz, and Guanya Shi. Fada: Few-shot domain adaptation via dynamics alignment for humanoid control. *arXiv preprint arXiv:2606.28476*, 2026.

Zhiyuan Zhou, Andy Peng, Qiyang Li, Sergey Levine, and Aviral Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data. In *International Conference on Learning Representations*, volume 2025, pp. 32343–32368, 2025.

Luisa M Zintgraf, Leo Feng, Cong Lu, Maximilian Igl, Kristian Hartikainen, Katja Hofmann, and Shimon Whiteson. Exploration in approximate hyper-state space for meta reinforcement learning. In *International Conference on Machine Learning*, pp. 12991–13001. PMLR, 2021.

APPENDIX

A ADDITIONAL EXPERIMENTAL DETAILS AND SETUP

We summarize the environments, and test-time dynamics shifts in Table 4. Across our evaluation suite, dynamics shifts span structural, physical, and morphological perturbations. Also we provide qualitative videos and code for our implementation here [4].

A.1 NAVIGATION

PointMaze. Figures 5 and 7 illustrate `PointMaze`, a continuous-control MuJoCo environment (Todorov et al., 2012) with two-dimensional state and action spaces. Pretraining utilizes reward-free trajectories collected by a random policy across a set of training layouts. At evaluation, agents are deployed on unseen layouts (Figure 6). The downstream goal is known, with task difficulties (easy, medium, and hard) defined by start-goal pairs of progressively longer shortest-path distances. We aggregate results across 20 held-out layouts, four training seeds, and five evaluation seeds per task. Figure 2(a) visualizes representative online exploratory trajectories.

Environment statistics. Mean shortest-path lengths on train versus test layouts are 6.30 vs. 5.88 (easy), 6.24 vs. 6.02 (medium), and 9.37 vs. 9.28 (hard). Consequently, held-out tasks are slightly shorter on average, which partially explains their marginally higher success rates compared to the train split (Table 6).

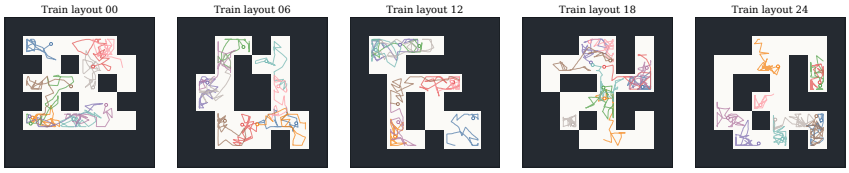


Figure 5: **Reward-free offline data.** Ten representative trajectories collected by a random policy across the `PointMaze` training layouts. The dataset provides broad state coverage but is of low quality (no successful trajectories).

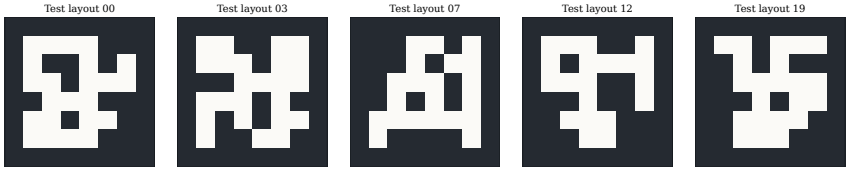


Figure 6: **Unseen layouts.** A subset of the unseen `PointMaze` layouts used for evaluation.

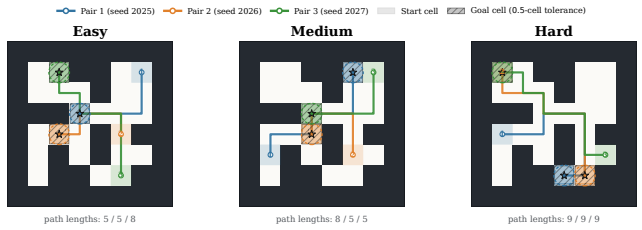


Figure 7: **Evaluation tasks.** Easy, medium, and hard start-goal configurations on an unseen layout, defined by increasing shortest-path distance to the goal. We sample three distinct start-goal pairs for each layout and difficulty level.

[4] We provide qualitative videos and code at anonymous.4open.science/w/AdaBFM/ and anonymous.4open.science/r/AdaBFM/, respectively.

AntMaze. For `AM-med`, we pretrain on the *medium*-quality D4RL dataset (Fu et al., 2020) under the nominal layouts {*medium, large*}. At test time, we evaluate across the modified environmental layouts specified by Lyu et al. (2024). While prior work restricts evaluation to a single diagonal start-goal pair, we introduce a more rigorous evaluating by testing across five distinct tasks per layout. Because the offline data provides limited state coverage from just one pretraining environment, solving these additional tasks requires robust continual training and sample-efficient exploration to navigate the novel dynamics and reach the downstream goal regions (see results in Table 13).

A.2 Locomotion

We pretrain all baselines on the *medium-quality* D4RL datasets (Fu et al., 2020) under nominal dynamics. Following the dynamics-shift protocol of Lyu et al. (2024), we evaluate on `HalfCheetah` and `Ant` across six environments spanning morphological and friction changes. Because these perturbations alter the physical transition dynamics while preserving the underlying objective (i.e., forward locomotion), the task remains identical across offline pretraining and online deployment. This provides an ideal setting to evaluate reward-free adaptation using a fixed task (i.e., skill or behavior). We summarize the final adaptation performance across all shifts in Tables 11 and 12, and visualize the corresponding continual training progression in Figure 10.

A.3 Dextrous Manipulation

We evaluate dextrous manipulation using the `Pen` task from the `Adroit` suite (Rajeswaran et al., 2017). We pretrain on a uniform mixture of Human, Cloned, and Expert offline datasets (comprising a total of $\sim$1M transitions) collected exclusively under nominal dynamics. To study dynamics adaptation, we evaluate these models across six shifted domains featuring broken joints and shrunk fingers at varying severities (Lyu et al., 2024). Because these morphological shifts fundamentally alter the physical interaction frequently causing the agent to drop the pen recovering performance requires discovering entirely novel manipulation behaviors. Consequently, reward-free continual training alone is insufficient. In this setting, we instead alternate between optimistic task inference (optimizing the task coordinate z_r using online rewards) and continual representation updates using exclusively the online-collected experience buffer.

A.4 Baselines

We compare memoryless and memory-based BFMs under both zero-shot transfer and reward-free TTA. This comparison isolates whether interaction history improves adaptation to unseen dynamics, and whether sample-efficient data collection provides additional gains under the same online budget. All the methods receive the same reward-free pretraining data, and the online interaction budget.

Category	Environment	Obs.	Act.	Horizon	Evaluated variations
LocoNavigate	PointMaze	2	2	250	`PM-rand` with 30 training and 20 unseen layouts. Easy, medium, and hard tasks.
	AntMaze	29	8	1,000	`AM-med` medium and large layouts. One nominal and six modified layouts. Five tasks per layout.
Locomotion	Ant	111	8	1,000	Nominal dynamics (med-quality). Shortened legs at three severities. Friction at $0.5\times$, $2\times$, and $5\times$.
	HalfCheetah	17	6	1,000	Nominal dynamics (med-quality). Thigh morphology at three severities. Friction at $0.5\times$, $2\times$, and $5\times$.
Dextrous Manipulation	Adroit Pen	45	24	100/200	Nominal environment with 100 steps. Disabled joints and shortened fingers at three severities with 200 steps.
OGBench	AntMaze Medium	29	8	1,000	Navigate and Stitch. Five tasks per suite.
	AntMaze Large	29	8	1,000	Navigate and Stitch. Five tasks per suite.
	SceneManipulator	40	5	750	Five nominal tasks.

Table 4: **Overview of environments.** We outline the observation spaces, action spaces, and episode lengths across all domains, detailing the specific layout variations, morphological changes, and friction shifts used for zero-shot and test-time adaptation.

TD-JEPA. TD-JEPA (Bagatella et al., 2025) is a zero-shot BFM on which our variants are built. We use the authors' official implementation[5] as a memoryless baseline. Its policy and successor predictor condition on the current state and task coordinate, but not on interaction history.

Belief-FB. BFB (Bobrin et al., 2025)[6] is a zero-shot BFM that extends Forward-Backward (FB) representations (Touati et al., 2023) by using a non-causal Transformer (Vaswani et al., 2017) to infer an episode-level dynamics context and condition its successor predictions on this context. This context encoder is pretrained exclusively via a reward-free next-state prediction mean squared error (MSE) loss, after which its parameters are frozen. Furthermore, while the original protocol assumes access to offline trajectories from the test environment, we remove this assumption. Our adaptation infers its context entirely from online transitions collected by its own deployment policy.

GRU-TD-JEPA. We adapt the recurrent BFM formulation of (Jeen et al., 2025)[7] to TD-JEPA. While (Jeen et al., 2025) maintain separate memory modules for the actor and successor predictor, our adaptation utilizes a single, shared GRU to summarize the growing interaction history. This shared encoder conditions both the successor predictor and the policy. Crucially, we apply a stop-gradient operator to the shared features to prevent actor gradients from updating the memory module. Consequently, the recurrent representation is trained exclusively via the TD-JEPA representation loss. Unlike LCL, this baseline employs no separate self-supervised context-learning objective. We choose this design to test whether temporal-difference successor-feature learning alone can induce useful in-context memory, complementing MetaBFM's separately trained self-supervised context encoder and motivated by evidence that reinforcement-learning objectives can improve offline in-context learning (Tarasov et al., 2025).

MetaBFM. MetaBFM (Grigsby et al., 2026)[8] conditions a BFM with latent context learning (LCL, Section 4.1). Its causal context lets the policy adjust its behavior using observed dynamics history. LCL prediction error can also serve as an intrinsic curiosity signal for exploration. However, under a limited interaction budget, it may prioritize exploring regions unrelated to the downstream task. Moreover, LCL does not explicitly represent uncertainty over successor measures or assess how actions from the current state affect the reachability of task-relevant future states.

AdaBFM. Building on TD-JEPA and LCL, AdaBFM incorporates a bootstrapped successor ensemble to capture alternative successor predictions. Bellman-consistency scores reweight this ensemble into a surrogate successor belief, which conditions the policy to enable adaptive behavior. To resolve dynamics uncertainty, AdaBFM selects task-compatible behaviors that maximize successor-belief information gain, using the resulting reward-free transitions to continually update its succesor predictor and policy.

A.5 Test-time Adaptation (TTA)

Table 5 summarizes the TTA protocol. Collected reward-free transitions are stored in an online-only replay buffer, ensuring the pretraining offline dataset is never revisited. During TTA, AdaBFM updates its context encoder (Section D.1), successor ensemble, and policy, while keeping the task encoder ψ and task coordinate z^* frozen for navigation and locomotion domains. Online rewards are utilized exclusively for the dextrous manipulation domain, though we also provide results for a fully reward-free TTA variant. Exploration during deployment follows the random-feasible strategy in eq. 7 which allows task oriented exploration, without this constraint we obserce the updates result in unlerning of representation C, leveraging the collected transitions for parameter updates under unseen dynamics.

[5] https://github.com/facebookresearch/td_jepa.git

[6] https://github.com/maxsbob/BeliefConditionedFB.git. The original codebase lacks continuous-control environments, we adapted it for our benchmarks.

[7] https://github.com/enjeeneer/bfms-with-memory.git

[8] Official code base unavailable

TTA setting	Navigation	Locomotion	Dexterous manipulation
Target-interaction budget B	2,000 per held-out layout	10,000 per dynamics shift	25,000 per dynamics shift
Episode horizon	250	1,000	100
Update frequency	one per transition	one per five transitions	one per five transitions
Total gradient updates	2,000	2,000	5,000
Online reward used	no	no	yes / no
Task encoder and z^*	frozen	frozen	frozen

Table 5: **Test-time adaptation protocol.** Interaction budgets, update frequencies, and frozen task components. Dexterous manipulation includes both reward-free and reward-based adaptation.

		Number of offline training layouts							
		Train layouts used in pretraining				*Unseen layouts (20)*			
Task	Method	1	3	10	30	1	3	10	30
Easy	TD-JEPA	11.1±19.2	0.0±0.0	0.0±0.0	1.1±0.0	0.0±0.0	0.0±0.0	0.0±0.0	1.1±1.0
	GRU-TD-JEPA	**77.8±19.2**	48.1±17.0	41.1±5.1	52.2±6.7	**26.1±1.9**	36.7±8.8	46.1±2.5	52.8±1.9
	MetaBFM	33.3±33.3	**59.3±6.4**	38.9±15.8	53.0±3.9	16.1±1.9	42.2±5.9	55.0±8.7	53.3±5.8
	AdaBFM	22.2±38.5	33.3±19.2	**52.2±15.0**	**55.2±6.1**	13.9±1.0	**43.3±1.7**	**59.4±9.2**	**60.0±5.8**
Medium	TD-JEPA	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.6±1.0	1.7±1.7	1.1±1.0	0.6±1.0
	GRU-TD-JEPA	**88.9±19.2**	**66.7±11.1**	40.0±11.5	47.0±3.4	**35.6±1.0**	26.7±7.3	39.4±5.9	47.8±7.7
	MetaBFM	55.6±38.5	44.4±29.4	41.1±12.6	53.0±1.7	19.4±6.7	**36.1±4.2**	50.6±3.5	**57.8±4.2**
	AdaBFM	22.2±19.2	48.1±6.4	**51.1±8.4**	**53.3±3.3**	16.1±2.5	35.0±5.0	**53.3±4.4**	57.2±1.0
Hard	TD-JEPA	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
	GRU-TD-JEPA	**22.2±38.5**	**11.1±11.1**	6.7±6.7	6.7±1.1	**3.3±0.0**	5.0±3.3	7.2±2.5	10.6±2.5
	MetaBFM	0.0±0.0	**11.1±11.1**	5.6±3.8	**12.2±4.0**	1.7±2.9	**7.8±4.2**	15.0±7.6	13.9±2.5
	AdaBFM	11.1±19.2	7.4±12.8	**10.0±5.8**	7.4±1.3	1.7±0.0	7.2±1.0	**16.1±5.4**	**15.0±6.0**
Avg.	TD-JEPA	3.7±6.4	0.0±0.0	0.0±0.0	0.4±0.0	0.2±0.3	0.6±0.6	0.4±0.3	0.6±0.0
	GRU-TD-JEPA	**63.0±23.1**	**42.0±11.9**	29.3±2.8	35.3±3.0	**21.7±1.0**	22.8±6.4	30.9±2.0	37.0±2.1
	MetaBFM	29.6±23.1	38.3±11.9	28.5±7.8	**39.4±2.5**	12.4±2.6	**28.7±3.1**	40.2±2.0	41.7±3.1
	AdaBFM	18.5±17.0	29.6±11.1	**37.8±8.0**	38.6±2.5	10.6±0.6	28.5±2.2	**43.0±5.3**	**44.1±2.6**

Table 6: **Zero-shot transfer across pretraining diversity in `PM-rand`.** Success (%) on training layouts versus 20 unseen layouts with three task difficulties each (60 test configurations). For each diversity level, the train split evaluates only the specific 1, 3, 10, or 30 layouts used during pretraining; thus, train-split results are not directly comparable across columns.

B Additional Experimental Results

B.1 Additional Results on PointMaze Tasks

Task-feasiblity. We ablate the task-value slack parameter ϵ in eq. 7 on the *hard* class of tasks across unseen layouts after pretraining AdaBFM on 30 training layouts. As shown in Table 7, $\epsilon = 0.5$ yields the best performance, which we adopt across all experiments. Candidate zero is always set to the target task coordinate ($z_0^E = z^*$), providing a task-directed, exploitative fallback. However, under poor task coverage (e.g., in completely unseen goal regions), this behavior alone may fail to discover successful paths. This limitation stems from the offline dataset: because online rewards are unavailable during pretraining, the task coordinate may not accurately represent novel target regions (see Section 6 for further discussion). Nevertheless, we present preliminary results in Section C on leveraging online rewards for adaptation under unseen dynamics.

Task-value slack ϵ	Success AUC (%) ↑
0.25	60.7 ± 12.7
0.5	$\mathbf{66.0 \pm 9.2}$
1	51.3 ± 7.9
∞	44.3 ± 10.1

Table 7: **Task-compatibility slack on unseen `PM-rand` layouts.** AUC% for the hard `PointMaze` task, measured across a 2,000-transition test-time adaptation budget.

Zero-shot performance on PointMaze tasks. Table 6 reports zero-shot success across varying pretraining diversity. In low-diversity regimes (e.g., single-layout pretraining), GRU-TD-JEPA outperforms other baselines, achieving an average success rate of 21.7%, compared to 12.4% for MetaBFM and 10.6% for AdaBFM. In this setting, a recurrent model can efficiently capture the temporal regularities of the single observed layout without needing to disentangle competing dynamics hypotheses. Conversely, context-conditioned MetaBFM and the AdaBFM ensemble receive insufficient layout

Task	Method	Number of offline training layouts 1	3	10	30
Easy	TD-JEPA	4.4±1.0 (+4.4)	3.3±1.7 (+3.3)	1.1±1.0 (+1.1)	0.0±0.0 (+0.0)
	GRU-TD-JEPA	70.0±15.0 (+43.9)	78.3±7.6 (+41.7)	83.3±2.9 (+37.2)	75.0±5.0 (+22.2)
	MetaBFM	**88.3±7.6 (+72.2)**	78.3±7.6 (+36.1)	88.3±7.6 (+33.3)	86.7±2.9 (+33.4)
	AdaBFM	75.0±8.7 (+61.1)	**91.7±2.9 (+48.3)**	**95.0±5.0 (+35.6)**	**95.0±5.0 (+35.0)**
Medium	TD-JEPA	2.8±1.9 (+2.8)	4.4±3.5 (+4.4)	2.2±1.9 (+2.2)	1.7±2.9 (+1.7)
	GRU-TD-JEPA	61.7±5.8 (+26.1)	58.3±14.4 (+31.7)	58.3±7.6 (+18.9)	66.7±7.6 (+18.9)
	MetaBFM	**65.0±8.7 (+45.6)**	80.0±0.0 (+43.9)	81.7±7.6 (+31.1)	78.3±7.6 (+20.5)
	AdaBFM	58.3±7.6 (+42.2)	**81.7±2.9 (+46.7)**	**85.0±5.0 (+31.7)**	**85.0±5.0 (+27.8)**
Hard	TD-JEPA	0.0±0.0 (+0.0)	0.0±0.0 (+0.0)	0.0±0.0 (+0.0)	0.0±0.0 (+0.0)
	GRU-TD-JEPA	23.3±2.9 (+20.0)	26.7±2.9 (+21.7)	28.3±2.9 (+21.1)	41.7±2.9 (+31.1)
	MetaBFM	**30.0±10.0 (+28.3)**	**50.0±5.0 (+42.2)**	48.3±5.8 (+33.3)	51.7±10.4 (+37.8)
	AdaBFM	23.3±5.8 (+21.7)	45.0±17.3 (+37.8)	**50.0±5.0 (+33.9)**	**56.7±2.9 (+41.7)**
Avg.	TD-JEPA	2.4±0.8 (+2.4)	2.6±1.4 (+2.6)	1.1±0.6 (+1.1)	0.6±1.0 (+0.6)
	GRU-TD-JEPA	51.7±4.4 (+30.0)	54.4±6.7 (+31.7)	56.7±1.7 (+25.7)	61.1±1.9 (+24.1)
	MetaBFM	**61.1±5.1 (+48.7)**	69.4±3.8 (+40.7)	72.8±3.5 (+32.6)	72.2±4.2 (+30.5)
	AdaBFM	52.2±6.7 (+41.7)	**72.8±7.7 (+44.3)**	**76.7±5.0 (+33.7)**	**78.9±1.7 (+34.8)**

Table 8: **TTA across pretraining diversity in `PM-rand`.** Test success after TTA (%); parentheses report the absolute gain compared to zero-shot success in Table 6.

variance during pretraining to construct informative latent contexts or distinct successor hypotheses, rendering ensemble disagreement largely uninformative.

This trend reverses as pretraining diversity expands. Exposure to varied layouts provides sufficient variation for the context encoder to identify layout-dependent dynamics and for the AdaBFM ensemble to form complementary successor predictions. Consequently, AdaBFM yields stronger performance, particularly in harder tasks where the successor-belief information gain helps update the successor and policy parameters.

Post-TTA performance. Table 8 shows that when pretrained on only a single layout, MetaBFM achieves the strongest performance following TTA. Although simpler recurrent models offer a stronger zero-shot prior, MetaBFM's compact latent representation allows rapid downstream adaptation to local constraints such as wall collisions under short online budgets. AdaBFM also improves substantially, though limited pretraining diversity prevents its ensemble from forming sufficiently distinct successor predictions. With three or more pretraining layouts, these ensemble predictions become far more informative, allowing AdaBFM to achieve the highest average post-TTA performance.

Ensemble Size. We evaluate the impact of ensemble size across tasks in `PM-rand30`. Generally, larger ensemble sizes improve performance, although an ensemble of $K = 10$ already achieves reasonable performance across both easy and hard task suites. As discussed in Section 6, a key limitation of such posterior estimation methods is that they remain approximate. Complete hyperparameter choices are detailed in Table 17.

Method	Easy	Medium	Hard	Avg.
AdaBFM	95.0 ± 5.0	85.0 ± 5.0	56.7 ± 2.9	78.9 ± 1.7
AdaBFM (−LCL)	16.67 ± 10.41	6.67 ± 5.77	0.00 ± 0.00	7.78 ± 5.36

Table 9: **Effect of LCL on `PM-rand30`.** Evaluation of AdaBFM with and without latent context conditioning across PointMaze tasks.

Effect of Latent Context Learning. As shown in Table 9, ablating LCL ($\text{AdaBFM}_{-\text{LCL}}$) severely affects agent performance, dropping average success from $\sim 70\%$ to $\sim 10\%$. In particular, the memoryless variant fails completely on hard tasks. This is a result of interference, as also observed in (Bobrin et al., 2025). Standard successor representations cannot distinguish distinct dynamics on their own and require a explicit input of context or history representation.

B.1.1 How does pretraining diversity affect transfer?

While Antmaze, locomotion, and Adroit use single-environment data, scaling layout diversity in `PM-rand` functions as offline meta-training (Tables 6 & 8).

Pretraining on diverse dynamics often induces severe representation interference in BFMs (Bobrin et al., 2025; Grigsby et al., 2026). In maze environments, memoryless variants such as TD-JEPA and

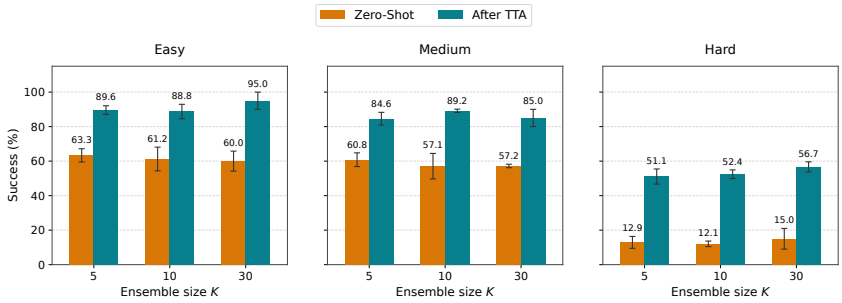


Figure 8: **Ensemble size ablation in `PM-rand30`.** Effect of ensemble size K on zero-shot and post-TTA performance.

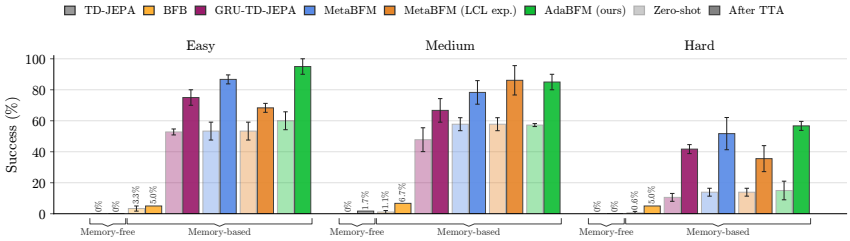


Figure 9: **Zero-shot transfer and TTA on unseen maze layouts.** Mean success before and after TTA; darker bars denote post-adaptation performance.

AdaBFM without LCL suffer performance collapse under diverse datasets (Table 9). Furthermore, layout scaling in `PM-rand` reveals two distinct adaptation regimes. Under a single pretraining layout, the ensemble observes insufficient dynamics variation to learn distinguishable successor hypotheses. Consequently, under a single layout, GRU-TD-JEPA achieves the strongest zero-shot transfer (Table 6), while MetaBFM adapts best online. With three or more pretraining layouts, AdaBFM leads across both settings; scaling from 1 to 30 layouts yields relative performance gains of 34% in zero-shot transfer and 27% in TTA.

B.2 Additional Results on Locomotion Tasks

Overall, Tables 12 and 11 show that continual training of the BFM representation generally improves adaptation to unseen dynamics. The corresponding learning curves in Figure 4 exhibit a broadly upward trend as additional target interactions are collected, although the benefit vary across models and dynamics.

Exploration and online continual training. Compared with PointMaze, locomotion is more locally recoverable. The effect of an exploratory action is typically short-lived and can be corrected by subsequent actions. In PointMaze, by contrast, an early decision may send the agent into the wrong corridor, having a long-term effect on goal reachability. Consequently, *random-feasible* exploration is already a strong locomotion baseline (Table 10), as it collects diverse dynamics-relevant transitions while largely preserving forward motion behavior.

TTA results. AdaBFM's strong zero-shot performance on unseen dynamics highlights the benefit of its belief-conditioned policy parameterization (Tables 12 and 11). The adaptive policy can select behavior appropriate to an unseen shift before any gradient updates are performed. During TTA, informative target transitions sharpen this belief and refine the underlying representation, yielding particularly strong results for Ant morphology changes and HalfCheetah friction changes. The memoryless TD-JEPA baseline also improves substantially and remains competitive, showing that continual representation learning alone can be effective, while AdaBFM gains further robustness by explicitly representing and resolving dynamics uncertainty.

Rule	Morph. easy	Morph. med.	Morph. hard	Fric. 0.5×	Fric. 2×	Fric. 5×
HalfCheetah (thigh morphology)						
Zero-shot	1228.3±134.0	1101.3±54.3	807.3±98.4	5626.3±356.5	5815.6±302.4	874.9±321.1
Task only	2726.3±319.1	**2347.2±52.5**	1340.3±199.0	6172.9±343.1	6759.6±213.4	4360.9±1931.8
Random feasible	2657.9±349.2	1879.5±142.4	**1340.6±100.2**	6257.1±170.5	**7026.5±153.5**	4478.5±2316.3
SF disagreement	2561.6±429.6	1785.7±386.0	1111.0±226.8	6132.8±441.2	6821.7±271.3	4551.2±1920.0
LCL error	2586.3±162.4	1915.5±302.9	1197.3±213.0	6267.6±357.6	6992.1±404.1	3700.7±1405.7
Successor IG	**2836.9±636.4**	2099.7±71.3	1327.4±311.5	**6530.9±178.6**	6955.7±328.1	**4739.5±1620.7**
Ant (all-leg morphology)						
Zero-shot	3424.5±101.7	940.0±77.8	-533.5±130.1	3944.5±320.6	**1401.8±253.9**	73.0±7.0
Task only	4357.0±353.2	**1523.9±386.4**	-65.9±716.8	4992.0±426.0	286.7±450.9	**563.5±155.9**
Random feasible	3964.5±341.2	1152.9±785.1	**76.6±412.8**	4658.2±384.1	127.9±2063.2	493.2±121.4
SF disagreement	2949.6±2101.8	708.7±213.1	15.0±287.4	**4916.0±332.5**	398.6±544.5	239.6±205.7
LCL error	4036.1±724.1	1445.7±794.5	-786.2±674.1	4429.2±624.0	100.1±529.2	367.9±242.6
Successor IG	**4463.3±84.6**	1073.6±973.0	-575.8±1078.8	4419.8±708.4	540.2±358.7	453.2±69.2

Table 10: **AdaBFM exploration ablation on locomotion.** Zero-shot and post-TTA returns under morphology and friction shifts.

	TD-JEPA	BFB	GRU-TD-JEPA	MetaBFM	AdaBFM
Nominal ZS	**5802.4±21.3**	3860.1±346.9	5651.6±182.8	5788.9±14.1	5781.9±328.0

(a) Thigh morphology

Shift	Method	ZS	TTA	Δ
Easy	TD-JEPA	985.5±73.8	1918.5±330.9	+933.1±264.8
	BFB	893.3±151.0	400.8±991.6	-492.6±1127.6
	GRU-TD-JEPA	985.2±92.1	1406.1±514.4	+420.9±436.8
	MetaBFM	1187.8±132.0	**2739.3±100.5**	**+1551.6±232.5**
	AdaBFM	**1228.3±134.0**	2539.9±149.7	+1311.6±155.5
Medium	TD-JEPA	959.8±138.9	1551.5±169.9	+591.7±87.6
	BFB	806.6±158.7	836.8±420.6	+30.2±505.4
	GRU-TD-JEPA	900.9±76.4	1164.1±483.3	+263.2±406.9
	MetaBFM	1088.7±46.4	**2174.3±306.0**	**+1085.6±351.6**
	AdaBFM	**1101.3±54.3**	2076.7±468.1	+975.3±429.3
Hard	TD-JEPA	595.9±75.7	979.4±398.5	+383.5±428.2
	BFB	598.7±95.5	273.2±431.8	-325.5±515.0
	GRU-TD-JEPA	645.0±27.5	736.1±583.1	+91.0±608.8
	MetaBFM	773.3±42.9	1212.3±181.4	+439.1±222.8
	AdaBFM	**807.3±98.4**	**1427.4±362.4**	**+620.0±326.3**

(b) Friction

Shift	Method	ZS	TTA	Δ
0.5×	TD-JEPA	**5715.2±25.9**	6013.2±115.0	+298.0±135.4
	BFB	4123.2±223.6	1539.1±2214.6	-2584.1±2022.1
	GRU-TD-JEPA	5528.2±185.1	6034.6±201.4	+506.4±378.6
	MetaBFM	5636.7±37.9	6084.7±84.9	+448.0±121.3
	AdaBFM	5626.3±356.5	**6250.6±425.8**	**+624.4±303.9**
2×	TD-JEPA	5782.2±84.4	6691.5±309.4	+909.2±344.0
	BFB	4195.6±211.2	2661.1±1662.2	-1534.5±1511.5
	GRU-TD-JEPA	5675.4±191.3	6369.1±168.9	+693.7±333.0
	MetaBFM	5773.9±24.1	6558.1±155.1	+784.2±136.2
	AdaBFM	**5815.6±302.4**	**6872.9±161.9**	**+1057.3±395.4**
5×	TD-JEPA	1209.6±353.4	4448.7±1386.8	+3239.1±1674.6
	BFB	**1729.8±920.6**	1314.1±1231.6	-415.7±684.5
	GRU-TD-JEPA	1669.3±517.1	3654.3±1019.4	+1985.0±504.5
	MetaBFM	1592.7±344.4	4362.9±1615.7	+2770.2±1901.9
	AdaBFM	874.9±321.1	**4725.5±1620.7**	**+3850.6±1921.8**

Table 11: **HalfCheetah test-time adaptation under dynamics shifts.** Zero-shot (ZS) and post-TTA returns; Δ is the paired gain.

B.3 Additional results on AntMaze

Table 13 presents the complete evaluation where baselines were pretrained on a single nominal dataset from D4RL (Fu et al., 2020) and evaluated across six different mazes of corresponding sizes, covering five distinct tasks per maze. Each task result is reported as an average over 10 evaluation seeds, with the standard deviation taken across training seeds. While the trajectories are of medium quality, they lack diversity and coverage, resulting in trends similar to those observed in single-layout PointMaze pretraining. While GRU-TD-JEPA achieves the strongest zero-shot performance on large mazes, AdaBFM achieves the highest post-TTA success on both maze sizes.

B.4 Additional results on Adroit

We pretrain on a mixture of human, cloned, and expert offline datasets collected exclusively on the nominal environment from D4RL (Fu et al., 2020). Table 14 presents the zero-shot and continual training results on the Adroit Pen tasks. Qualitatively, we observe that the dynamics shifts proposed in ODRL (Lyu et al., 2024) present severe zero-shot challenges, often causing the agent to drop the pen, resulting in an unrecoverable failure. This occurs because dynamics shifts such as broken joints or a shrunk finger require fundamentally different control behaviors to hold and manipulate the pen. While reward-free continual training recovers a portion of performance, we also evaluate an optimistic task update mechanism (OpTI-BFM (Rupf et al., 2025)), which operates under continual episodic updates with binary or sparse rewards without requiring privileged resets. OpTI-BFM uses observed rewards

	TD-JEPA	BFB	GRU-TD-JEPA	MetaBFM	AdaBFM
Nominal ZS	**4396.3±38.0**	1817.4±823.7	3405.7±932.5	3716.8±165.2	3607.5±415.1

(a) All-leg morphology

Shift	Method	ZS	TTA	Δ
Easy	TD-JEPA	3697.7±129.3	4517.6±197.2	+819.9±68.0
	BFB	2752.3±340.4	129.0±1731.2	-2623.3±1553.9
	GRU-TD-JEPA	**3939.7±206.3**	4217.4±264.4	+277.7±64.9
	MetaBFM	3462.3±66.6	4397.9±246.6	+935.6±238.4
	AdaBFM	3424.5±101.7	**4578.5±23.3**	**+1153.9±107.5**
Medium	TD-JEPA	845.4±66.5	1203.5±230.0	+358.0±248.9
	BFB	737.4±225.4	-186.9±758.9	-924.3±763.1
	GRU-TD-JEPA	**951.8±88.8**	925.7±480.0	-26.0±554.9
	MetaBFM	832.6±127.3	**1360.1±712.2**	**+527.6±698.8**
	AdaBFM	940.0±77.8	983.0±470.4	+42.9±403.6
Hard	TD-JEPA	-464.7±172.3	-555.9±690.9	-91.2±533.4
	BFB	**164.1±62.3**	-934.6±959.7	-1098.7±1000.0
	GRU-TD-JEPA	-344.6±248.8	-201.6±466.8	+143.0±553.6
	MetaBFM	-362.5±282.7	121.8±454.9	+484.3±519.7
	AdaBFM	-533.5±130.1	**190.7±441.6**	**+724.2±503.4**

(b) Friction

Shift	Method	ZS	TTA	Δ
0.5×	TD-JEPA	4748.0±22.8	5166.2±17.2	+418.3±22.4
	BFB	2630.4±368.8	-121.0±824.5	-2751.4±1184.7
	GRU-TD-JEPA	**4769.9±138.2**	5115.0±416.8	+345.1±295.2
	MetaBFM	3895.9±264.2	5107.5±566.1	+1211.5±312.6
	AdaBFM	3944.5±320.6	**5476.8±62.9**	**+1532.2±345.5**
2×	TD-JEPA	936.9±407.9	766.9±894.3	-170.0±751.0
	BFB	265.1±483.4	-281.8±302.8	-546.9±372.8
	GRU-TD-JEPA	753.2±40.8	280.1±230.1	-473.1±209.2
	MetaBFM	1095.9±237.8	**982.8±2030.7**	**-113.0±1816.2**
	AdaBFM	**1401.8±253.9**	65.0±220.6	-1336.8±81.3
5×	TD-JEPA	30.2±13.0	407.9±234.2	**+377.7±223.9**
	BFB	-119.0±27.4	-162.0±127.9	-43.0±100.8
	GRU-TD-JEPA	59.9±2.7	422.5±390.1	+362.6±392.3
	MetaBFM	57.6±16.3	162.4±338.1	+104.8±329.5
	AdaBFM	**73.0±7.0**	**436.2±208.2**	+363.1±201.4

Table 12: **Ant test-time adaptation under dynamics shifts.** Zero-shot (ZS) and post-TTA returns; Δ is the paired gain.

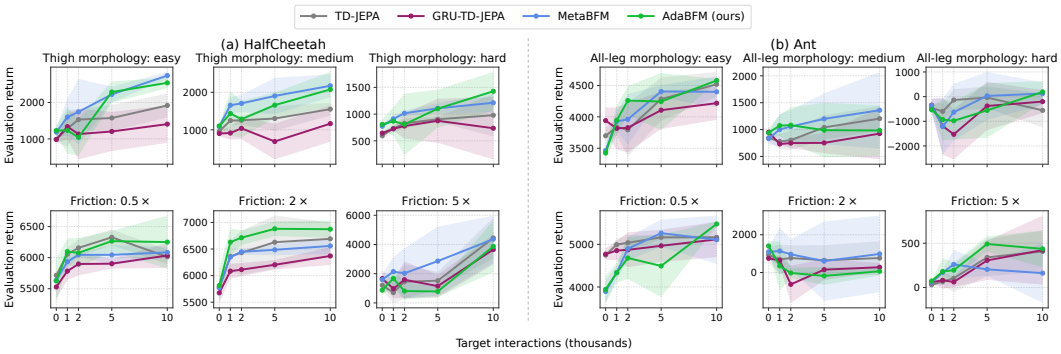


Figure 10: **Online adaptation of locomotion tasks under dynamics shifts.** Test-time training progression using reward-free online transitions; no online rewards are used (see Tables 11 and 12).

to update a reward posterior and search the pretrained behavior space optimistically for promising candidate behaviors. Alternating this search with continual training iteratively improves both the policy and successor predictions using transitions collected under the shifted dynamics. Table 14 presents the full results for this task.

B.5 Visualization of Representations

Similar to TD-JEPA (Lawson et al., 2026; Bagatella et al., 2025), given a state-action pair (s, a) and a goal g, we visualize the cosine similarity between the successor prediction $T_\phi(\phi(s), a, z_g)$ and the goal feature $\psi(g)$. Figure 11 shows these similarities on an unseen `PointMaze` layout after TTA, together with final policy rollouts. Figure 12 shows the same quantity along shared offline trajectories in a training layout. Higher similarity indicates stronger alignment with the goal feature; high similarity far from the goal may suggest optimistic predictions of goal reachability.

B.6 Pretraining Results on OGBench

To benchmark AdaBFM's baseline pretraining performance, we evaluate offline pretraining and zero-shot task inference on OGBench (Park et al., 2025a) under nominal dynamics (i.e., without online adaptation or explicit dynamics shifts). We benchmark AdaBFM across 10 training seeds against the zero-shot BFM and representation learning baselines established in TD-JEPA (Bagatella et al., 2025): Laplacian (Wu et al., 2018), ICVF* (Ghosh et al., 2023), HILP (Park et al., 2024a), FB (Touati &

Maze family	Method	Nominal: tasks solved (%) Zero-shot	Nominal: tasks solved (%) After TTA	Unseen: tasks solved (%) Zero-shot	Unseen: tasks solved (%) After TTA
Medium	TD-JEPA	**85.00±10.00**	**80.00±23.00**	35.83±5.70	39.17±5.00
	GRU-TD-JEPA	45.00±30.00	75.00±19.20	25.83±6.87	42.50±8.33
	BFB	15.00±19.20	35.00±34.20	30.00±4.70	26.67±13.07
	MetaBFM	60.00±16.40	70.00±20.00	32.50±5.70	42.50±7.40
	AdaBFM	75.00±19.20	75.00±30.00	**35.83±4.20**	**46.67±3.83**
Large	TD-JEPA	**25.00±19.20**	10.00±11.60	20.83±1.67	18.33±6.37
	GRU-TD-JEPA	20.00±16.40	20.00±16.40	**25.83±3.20**	25.83±3.20
	BFB	15.00±19.20	15.00±19.20	11.67±5.77	13.33±4.70
	MetaBFM	15.00±10.00	20.00±16.40	21.67±4.30	28.33±4.30
	AdaBFM	10.00±11.60	**35.00±10.00**	20.83±7.40	**40.83±1.67**

Table 13: **AntMaze adaptation performance (% tasks solved).** Comparison of zero-shot and post-TTA success rates across varying maze scales.

Nominal

Protocol	TD-JEPA	BFB	GRU-TD-JEPA	MetaBFM	AdaBFM
Offline evaluation	20.0±0.0	20.0±14.1	**60.0±0.0**	8.0±8.5	16.0±19.8
Reward-free CT	45.0±21.2	10.0±0.0	**53.0±15.6**	8.0±11.3	45.0±21.2
Reward-based CT	**88.0±11.3**	50.0±0.0	61.0±9.9	77.0±1.4	79.0±9.9

(a) Broken joints

Severity	Method	Offline	RF CT	R CT
Easy	TD-JEPA	15.0±7.1	15.0±7.1	49.0±7.1
	BFB	8.0±2.8	5.0±7.1	37.0±9.9
	GRU-TD-JEPA	**40.0±0.0**	**37.0±18.4**	46.0±11.3
	MetaBFM	11.0±12.7	8.0±8.5	50.0±8.5
	AdaBFM	9.0±9.9	35.0±21.2	**52.0±8.5**
Medium	TD-JEPA	10.0±0.0	10.0±14.1	15.0±4.2
	BFB	2.0±0.0	0.0±0.0	10.0±2.8
	GRU-TD-JEPA	**25.0±21.2**	**18.0±14.1**	12.0±2.8
	MetaBFM	3.0±4.2	8.0±8.5	10.0±0.0
	AdaBFM	4.0±2.8	10.0±14.1	**22.0±8.5**
Hard	TD-JEPA	5.0±7.1	5.0±7.1	10.0±0.0
	BFB	2.0±0.0	5.0±7.1	8.0±2.8
	GRU-TD-JEPA	**20.0±0.0**	**14.0±5.7**	**11.0±4.2**
	MetaBFM	2.0±0.0	2.0±0.0	8.0±5.7
	AdaBFM	4.0±5.7	0.0±0.0	5.0±7.1
Average	TD-JEPA	10.0±4.7	10.0±4.7	24.7±0.9
	BFB	4.0±0.9	3.3±0.0	18.3±3.3
	GRU-TD-JEPA	**28.3±7.1**	**23.0±12.7**	23.0±6.1
	MetaBFM	5.3±5.7	6.0±5.7	22.7±0.9
	AdaBFM	5.7±6.1	15.0±11.8	**26.3±3.3**

(b) Shrunk fingers

Severity	Method	Offline	RF CT	R CT
Easy	TD-JEPA	5.0±1.4	**10.0±0.0**	10.0±5.7
	BFB	4.0±0.0	0.0±0.0	7.0±9.9
	GRU-TD-JEPA	**15.0±7.1**	4.0±5.7	8.0±2.8
	MetaBFM	3.0±1.4	0.0±0.0	**12.0±5.7**
	AdaBFM	2.0±2.8	0.0±0.0	5.0±4.2
Medium	TD-JEPA	16.0±11.3	20.0±14.1	**41.0±4.2**
	BFB	6.0±2.8	15.0±21.2	29.0±1.4
	GRU-TD-JEPA	**40.0±0.0**	**32.0±0.0**	40.0±0.0
	MetaBFM	4.0±5.7	20.0±14.1	34.0±2.8
	AdaBFM	14.0±19.8	25.0±7.1	40.0±2.8
Hard	TD-JEPA	2.0±2.8	25.0±21.2	19.0±9.9
	BFB	6.0±5.7	15.0±7.1	5.0±1.4
	GRU-TD-JEPA	**10.0±0.0**	16.0±11.3	19.0±4.2
	MetaBFM	2.0±2.8	20.0±28.3	23.0±1.4
	AdaBFM	9.0±12.7	**30.0±14.1**	**29.0±1.4**
Average	TD-JEPA	7.7±5.2	18.3±11.8	23.3±2.8
	BFB	5.3±2.8	10.0±9.4	13.7±4.2
	GRU-TD-JEPA	**21.7±2.4**	17.3±5.7	22.3±2.4
	MetaBFM	3.0±3.3	13.3±4.7	23.0±3.3
	AdaBFM	8.3±11.8	**18.3±2.4**	**24.7±1.9**

Table 14: **Results on Adroit Pen across different dynamics shift.** Offline evaluation is based on the zero-shot results. RF CT and R CT denote reward-free and reward-based continual training.

Ollivier, 2021), RLDP (Jajoo et al., 2026), BYOL* (Grill et al., 2020), and BYOL-$\gamma^\star$ (Lawson et al., 2026).[9]

As reported in Table 15, both variants of AdaBFM achieve performance competitive with top-performing BFM methods, generally outperforming vanilla TD-JEPA. Notably, the memory-free variant ($\mathrm{AdaBFM}_{-\mathrm{LCL}}$) outperforms the full history-conditioned AdaBFM on most tasks. This is expected, as OGBench environments are fully observable and strictly Markovian, rendering history-dependent models unnecessary. In such fully observable settings, employing a non-Markovian (history-conditioned) architecture can actively degrade test-time generalization (De Haan et al., 2019; Wen et al., 2020). This phenomenon typically arises from causal confusion where history-dependent policies exploit temporal correlations, such as repeating past actions as well as an increased vulnerability to distribution shifts induced by the higher-dimensional input space. This performance gap is particularly noticable in the *Stitch* environments, where poorer dataset coverage exacerbates history misattribution, causing a 10-15% drop relative to the memoryless variant ($\mathrm{AdaBFM}_{-\mathrm{LCL}}$).

[9]Baseline performance numbers and evaluation protocols are extended directly from Table 1 of Bagatella et al. (2025).

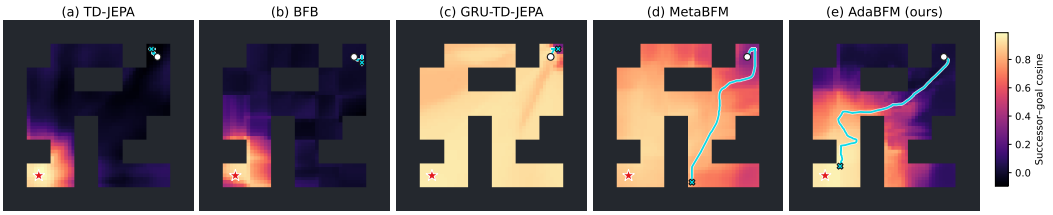


Figure 11: **Post-TTA successor geometry on an unseen maze.** Colors show the cosine similarity $\cos\big(T_\phi(\phi(s), a, z_g), \psi(g)\big)$ between the adapted successor prediction and the goal feature. Brighter values indicate states predicted to be more aligned with the goal, revealing shortest-path structure. The white circle and red star denote the start and goal.

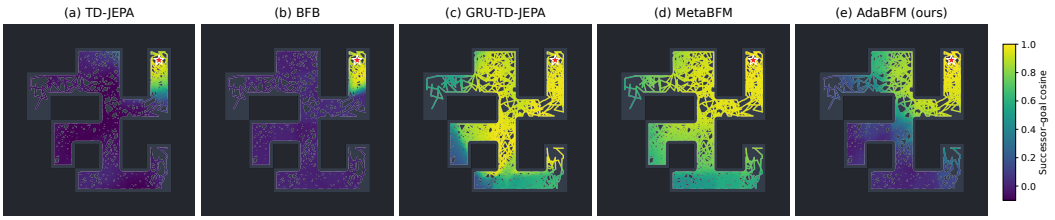


Figure 12: **Successor geometry on train layout.** Colors show offline trajectories colored according to the cosine similarity $\cos\big(T_\phi(\phi(s), a, z_g), \psi(g)\big)$ between the adapted successor prediction and the goal feature. Brighter values indicate states predicted to be more aligned with the goal, revealing shortest-path structure. The white circle and red star denote the start and goal.

Overall, we observe that AdaBFM maintains strong zero-shot foundation capabilities alongside its test-time adaptation mechanisms.

B.7 Comparison with Meta-RL

Memory-based meta-RL uses interaction histories to maintain implicit beliefs about the current environment and guide adaptation (Duan et al., 2016; Wang et al., 2016; Ni et al., 2022; Beck et al., 2025). MetaBFM (Grigsby et al., 2026) extends this perspective to BFMs via memory-based pretraining and intrinsic exploration driven by latent-context prediction errors, balancing exploration and exploitation at test time across novel settings, through in-context learning.

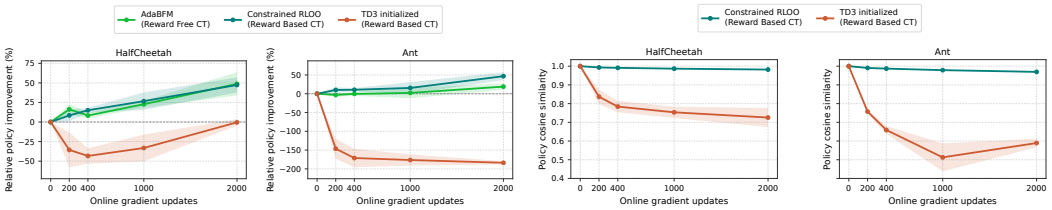
Related Bayesian meta-RL approaches explicitly model exploration under task uncertainty. HyperX (Zintgraf et al., 2021) combines hyper-state novelty with prediction-error bonuses to learn informative adaptation strategies. BOReL (Dorfman et al., 2021) learns belief-conditioned exploration from offline data and characterizes *MDP ambiguity*, where observed transitions fail to separate competing underlying environments. This highlights the need to collect targeted evidence that resolves representation ambiguity. AdaBFM brings this belief-aware perspective to reward-free BFMs. Rather than meta-learning an exploration policy across predefined tasks, it leverages successor-belief information gain to select task-compatible exploratory behaviors at deployment while performing budgeted updates to the context encoder, successor ensemble, and policy.

C Reward-based online adaptation

Tables 12 and 11 show that reward-free continual training can recover BFM performance under unseen dynamics. When rewards are available, we consider two approaches. (i) Initialize the policy and critic through offline task inference (Figure 1), then apply online RL to directly optimize the reward, following Kim et al. (2025). (ii) Extend LOLA (Sikchi et al., 2025) by alternating between reward-based search in latent z-space and continual training of the policy and successor features, holding the selected z fixed during each training phase. For the latter, we use online rewards to guide gradient-based search within the feasible set defined in eq. 7.

Environment	Laplacian	ICVF*	HILP	FB	RLDP	BYOL*	BYOL-γ*	TD-JEPA	AdaBFM$_{-\mathrm{LCL}}$	AdaBFM
Medium-Navigate	50.00±4.94	79.80±2.62	**83.60±2.63**	73.00±2.72	74.60±4.15	58.40±2.00	51.40±1.55	70.40±3.72	77.7±4.5	79.1±8.7
Large-Navigate	21.60±3.90	58.40±1.90	52.60±3.86	36.80±4.28	36.40±4.66	26.60±3.03	21.80±3.57	57.20±4.25	**68.9±8.2**	54.6±7.5
Medium-Stitch	21.40±4.32	39.00±3.30	50.60±2.46	**70.40±3.95**	58.40±3.29	60.60±5.07	45.60±2.84	61.56±4.53	63.2±5.3	51.2±9.1
Large-Stitch	11.80±1.47	13.20±1.64	12.20±1.75	**49.80±5.64**	19.60±2.73	25.80±4.28	20.20±1.80	40.60±2.51	38.2±8.7	22.5±4.6
Scene	7.80±1.28	45.40±2.29	43.80±1.90	12.80±1.61	11.60±1.57	15.40±1.37	14.40±2.32	38.44±1.37	40.2±4.7	**46.5±4.9**
$\mathrm{OGBench}_{\mathrm{avg}}$	22.52±1.58	47.16±1.08	48.56±1.18	48.56±1.74	40.12±1.55	37.36±1.54	30.68±1.13	53.64±1.56	**57.64±2.92**	50.78±3.22

Table 15: State-based OGBench zero-shot success (%).



(a) Relative policy improvement during TTA

(b) Cosine similarity of z to its zero-shot value

Figure 13: **Policy drift vs. performance improvement.** Relative performance improvement over the zero-shot policy and the corresponding change in cosine similarity of z to its initial zero-shot value. Curves represent averages over six distinct dynamics tasks for both Ant and HalfCheetah.

Figure 13 compares these two approaches along with our reward-free TTA. Despite access to dense rewards, direct online optimization exhibits an early performance collapse from which it does not recover within the interaction budget. This collapse coincides with a sharp decrease in the cosine similarity between the initial task coordinate and the reward-derived coordinate during adaptation, indicating that the selected behavior moves away from the pretrained zero-shot policy. Similar instability from updating pretrained policies with limited online data has been associated with unlearning (Zhou et al., 2025; Sikchi et al., 2025).

AdaBFM instead keeps the inferred task coordinate z^* fixed. Exploratory coordinates z_i^E are used only to select behaviors from the pretrained policy family, and the feasible set $\mathcal{Z}_\epsilon$ excludes candidates with low predicted value under z^*. The collected transitions then update the dynamics and control components without changing the task specification. Thus, the latent search acts as a task-conditioned regularizer during reward-free continual training.

D Implementation Details

D.1 Latent Context Learning

Given an interaction history H_t, the LCL encoder (Grigsby et al., 2026) produces a causal context c_t. The context c_t represents the dynamics currently inferred from that history. LCL is trained from reward-free transitions using three self-supervised prediction heads: the latent-dynamics head $D_\omega(h_t, a_t, c_t)$ predicts the next history state, the inverse-dynamics head $I_\omega(h_t, h_{t+1}, c_t, c_{t+1})$ predicts the action, and the forward-dynamics head $G_\omega(s_t, a_t, c_t)$ predicts the state increment. Its objective is $\mathcal{L}_{\mathrm{LCL}} = \mathcal{L}_{\mathrm{LDM}} + \mathcal{L}_{\mathrm{IDM}} + \mathcal{L}_{\mathrm{FDM}}$, where

$$\mathcal{L}_{\mathrm{LDM}} = 1 - \cos(D_\omega(h_t, a_t, c_t), \mathrm{sg}[h_{t+1}])\,, \tag{10}$$

$$\mathcal{L}_{\mathrm{IDM}} = \|I_\omega(h_t, \mathrm{sg}[h_{t+1}], c_t, c_{t+1}) - a_t\|_2^2\,, \tag{11}$$

$$\mathcal{L}_{\mathrm{FDM}} = \|s_t + G_\omega(s_t, a_t, \mathrm{sg}[c_t]) - s_{t+1}\|_2^2\,. \tag{12}$$

Here, $\mathrm{sg}[\cdot]$ denotes stop-gradient. LDM and IDM train the history and context encoders to retain action-dependent dynamics information. FDM trains G_ω to predict the observed state change; because c_t is detached, it evaluates whether the inferred context explains physical dynamics without updating the context encoder through state reconstruction.

The LCL objective remains separate from the BFM objectives. The policy and successor predictor are conditioned on (h_t, c_t), but their gradients are stopped at these representations. Thus, the LCL

LCL hyperparameter	PointMaze	AntMaze	Locomotion	Adroit Pen	OGBench
Local feature dimension $\dim h$	64	64	64	64	64
Context dimension $\dim c$	64	128	64	64	64
Context window	64	64 / 128	128	128	128
Transformer (width, layers, heads)	$(128, 2, 4)$	$(256, 3, 8)$	$(128, 2, 4)$	$(128, 2, 4)$	$(128, 2, 4)$
LCL learning rate	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}
$(\lambda_{\text{LDM}}, \lambda_{\text{IDM}}, \lambda_{\text{FDM}})$	$(1, 1, 1)$	$(1, 1, 1)$	$(1, 1, 1)$	$(1, 1, 1)$	$(1, 1, 1)$
LCL target-network rate	0.005	0.005	0.005	0.005	0.005

Table 16: **LCL hyperparameters.** PointMaze context windows correspond to historical $K = 30$ and $K = 5/10$ runs; AntMaze windows correspond to medium and large mazes, respectively.

encoder is updated through its self-supervised dynamics losses, while the policy and successor predictor are updated through their respective BFM losses.

D.2 Policy Regularization

Following TD-JEPA (Bagatella et al., 2025) and Flow Q-learning (Park et al., 2025b), we regularize the belief-conditioned policy using a flow-based behavioral cloning (BC) model. The BC flow conditions on the state and context $\bar{c}_t$, without the task coordinate z. For a replay action a_t, noise $\xi \sim \mathcal{N}(0, I)$, and $\tau \sim \mathcal{U}[0, 1]$, let $x_\tau = (1 - \tau)\xi + \tau a_t$. The flow-matching objective is

$$\mathcal{L}_{\text{Flow}-\text{BC}}(\omega) = \mathbb{E}_{\mathcal{D},\, \xi,\, \tau}\left[\|v_\omega(s_t, \bar{c}_t, x_\tau, \tau) - (a_t - \xi)\|_2^2\right]. \tag{13}$$

Sampling from this flow policy requires iterative integration. We therefore distill it into a one-step generative actor, $a_t^\pi = f_\theta(s_t, \bar{c}_t, b_t, z, \xi)$, which additionally conditions on the belief b_t and task coordinate z. The belief-weighted policy objective is applied to this actor:

$$\mathcal{L}_\pi(\theta) = -\mathbb{E}\left[Q_b(\phi_t, c_t, a_t^\pi, z)\right], \tag{14}$$

where Q_b denotes the belief-weighted critic in eq. 4. The regularized actor objective is

$$\mathcal{L}_\pi^{\text{reg}}(\theta) = \mathcal{L}_\pi(\theta) + \lambda_{\text{BC}}\, \mathbb{E}\left[\left\| a_t^\pi - \xi + \int_0^1 v_\omega(s_t, \bar{c}_t, \tilde{x}_\tau, \tau)\, d\tau \right\|_2^2\right], \tag{15}$$

where $\tilde{x}_0 = \xi$ (Gaussian noise) and $d\tilde{x}_\tau/d\tau = v_\omega(s_t, \bar{c}_t, \tilde{x}_\tau, \tau)$ define the sampled flow trajectory. The actor and flow policy use the same noise sample for distillation. We use the same BC hyperparameters as TD-JEPA.

D.3 Architectures and Hyperparameters

Table 16 reports the LCL configuration. LCL is trained jointly with the BFM, while its encoder is updated only through the self-supervised dynamics losses described in Section D.1.

AdaBFM builds on TD-JEPA (Bagatella et al., 2025) and retains its shared encoder and successor-predictor architecture across baselines. Table 17 summarizes the resulting hyperparameters. AdaBFM mainly differs through its LCL module and the number of successor ensemble memebers (Sections 4.1 and 4.2).

D.4 Computation Costs

Our implementation uses PyTorch, with preprocessed datasets loaded into GPU memory to reduce data-loading overhead. Computation consists of two stages: *offline pretraining* and *test-time adaptation*. Pretraining is performed once per seed, and the resulting checkpoint is reused across downstream tasks and dynamics shifts.

Table 18 summarizes approximate runtimes on an NVIDIA L40S GPU. Pretraining takes hours, whereas adaptation takes minutes under our fixed interaction and update budgets. The causal Transformer adds computational overhead to both MetaBFM and AdaBFM. The simpler GRU-based context encoder is generally faster. AdaBFM additionally evaluates an ensemble of successor

Hyperparameter	PointMaze	AntMaze	Locomotion	Adroit Pen	OGBench
ϕ-encoder output dimension	256	256	256	256	256
ψ-encoder output dimension	50	50	50	50	50
Encoder/predictor hidden width	512	512	512	512	512
Successor-feature discount γ	0.99	0.999	0.99	0.99	0.99
Belief update rate η	0.1	0.1	0.1	0.1	0.1
Successor-ensemble size K	30	10	5	5	10

Table 17: **AdaBFM hyperparameters.**

	Pretraining (hours)		TTA (minutes)	
Method	PointMaze	Locomotion	PointMaze	Locomotion
TD-JEPA	3	4	1.3	7
BFB	18	19	2.1	9
GRU-TD-JEPA	5	4	2.2	8
MetaBFM	9	9	2.8	12
AdaBFM	12–15	12–15	3	12

Table 18: **Approximate computational costs.** Estimated pretraining time per seed for one million gradient updates, and TTA time per deployment using 2,000 PointMaze or 10,000 locomotion transitions.

predictors during belief updates and exploration. These computations are batched on the GPU and require no additional environment interactions.

On `PM-rand30`, pretraining for one million updates per seed took approximately 12 hours for both $K = 5$ and $K = 10$, while $K = 30$ took about 14 hours with four seeds running concurrently on a single NVIDIA L40S.