WHAT GOALS ARE AHEAD? KERL: KERNEL MEAN EMBEDDING TO SCALE RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Goal-conditioned RL commonly uses hindsight relabeling to associate each state-action pair in a replay buffer with a goal reached later in the same trajectory. By mixing experience collected under different commanded goals, this implicitly samples from a distribution over goals that is rarely made explicit. We formalize it as the *posterior-averaged goal occupancy measure* and introduce KERL, an actor–critic method grounded in its kernel mean embedding. With a characteristic kernel, the kernel mean embedding uniquely determines the underlying distribution. In practice, KERL approximates it in finite dimensions by training the critic to predict the expected random Fourier features of future goals. These fixed features replace the learned goal encoder of contrastive RL, while a squared-error regression loss replaces its contrastive objective, reducing computational cost within the same actor–critic structure. The learned representation enables reconstruction of the goal distribution, revealing which goals are expected to follow a given state-action pair. Our experiments and ablation studies show that KERL scales effectively with network depth, achieving large gains on the hardest JaxGCRL tasks.

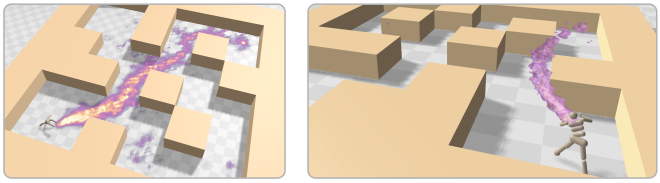

Figure 1: **What goals are ahead?** From KERL's representation $\mu^{\pi^*}(s, a)$, we recover the posterior-averaged goal occupancy $\rho^{\pi^*}(\cdot \mid s, a)$, which is the distribution of goals expected to follow a *single* state-action pair. This measure does **not** depend on a commanded goal g^c, it is a property of π^*, not the goal occupancy of $\pi^*(\cdot \mid \cdot, g^c)$.

1 INTRODUCTION

Goal-Conditioned Reinforcement Learning (GCRL) moves RL beyond hand-designed rewards by defining tasks in terms of goals. This opens a path toward general-purpose agents that master many tasks without dense supervision, as recently pursued in pretraining or fine-tuning Vision-Language-Action models (Intelligence et al., 2025; Zhang et al., 2026a). The challenge is sparse reward, as the agent receives no feedback until it reaches the goal, possibly after a long horizon. Early approaches adapted existing RL recipes, borrowing from imitation learning (Ghosh et al., 2019), hierarchical RL (Nachum et al., 2018), planning- or model-based methods (Mendonca et al., 2021). A common component is *hindsight relabeling* (Andrychowicz et al., 2017, HER), which extracts supervision even from unsuccessful attempts by treating goals actually reached as alternative objectives.

Recent methods use this supervision to learn directly from goals observed later in a trajectory. Contrastive RL (CRL) (Eysenbach et al., 2022) learns representations of state-action pairs and goals whose inner product serves as a critic, using a contrastive objective to associate each pair with its future goals. Survival-based methods (Tiofack et al., 2026) instead estimate goal-conditioned values from the distribution of reaching times. Both approaches have shown promising gains from increasing network depth (Wang et al., 2026; Nguimatsia-Tiofack et al., 2026). Despite their different objectives, both learn to associate state-action pairs with goals observed later along the same trajectories. Because these trajectories are collected under different commanded goals, the resulting conditional distribution of future goals mixes the occupancies induced by those commands. This distribution, which underlies hindsight relabeling, is rarely made explicit.

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.

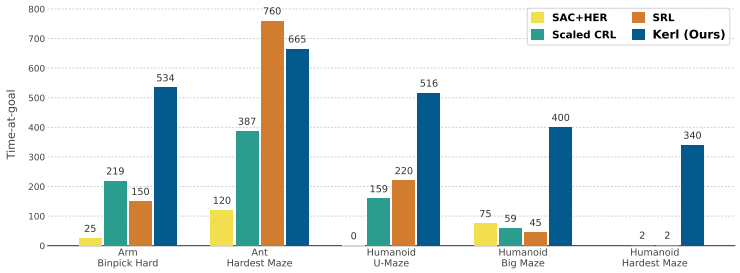

Figure 2: **KERL on the hardest JaxGCRL tasks**. Time-at-goal against SAC+HER, scaled CRL and SRL, averaged over the last 5 epochs and 5 seeds. Full curves in Fig. 4.

Our first contribution (see Sec. 2) is to precisely characterize it as the **posterior-averaged goal occupancy measure**. It averages the goal occupancies induced by the policy under different commanded goals, weighting each command by its posterior probability given the observed state-action pair. The measure thus describes which goals are expected to follow that pair after averaging over possible commands. In order to characterize the structure of this distribution, we illustrate on toy examples and use it to motivate evaluating actions by comparing their distribution of future goals with a point mass at the desired goal.

Our second contribution (see Sec. 3 and Sec. 4) is to represent each state-action pair by the kernel mean embedding of the posterior-averaged goal occupancy measure the pair induces. For a *characteristic* kernel, this embedding uniquely characterizes the corresponding distribution (Muandet et al., 2017). This leads to **KERL** (a contraction of Kernel and RL), a representation learning approach that approximates the infinite-dimensional feature map using fixed random Fourier features (Rahimi & Recht, 2007) and learns the mean embedding by least-squares regression. In Sec. 3.3, we provide concentration bounds for the approximations involved. We also show that the learned representation supports approximate reconstruction of the measure of interest, allowing us to visualize which goals the critic expects to follow a single state-action pair, as illustrated in Fig. 1.

As in CRL and prior work on successor states (Blier et al., 2021), the actor is trained to select actions by comparing the critic's state-action representation with the representation of the desired goal. We derive two formulations of the actor objective, based on similarity or distance between these representations. Implementing KERL requires only minor changes to CRL: the fixed goal features replace its learned goal encoder, and a squared regression loss replaces its contrastive objective. Using one critic network instead of two cuts the cost per gradient step by a factor of 1.5.

Finally, our third contribution (see Sec. 5) is an empirical study across the JaxGCRL suite. We notably show that KERL scales with depth as established by Wang et al. (2026), and provide a comprehensive ablation study to support these findings. Overall, KERL matches or improves the state of the art, with large margins on the hardest tasks (Fig. 2). While long-horizon control of a high-dimensional body has been a standing failure case for GCRL, KERL makes substantial progress. For instance, on `humanoid-U-maze`, KERL reaches a *time-at-goal* of 516, more than triple the 159 reported by Wang et al. (2026) as headline evidence that CRL scales. On `humanoid-hardest-maze`, which no prior work reports, we run scaled CRL and SRL, and both reach only a score of 2, while KERL reaches 340. As KERL connects several lines of work, from contrastive and successor-based representations to kernel methods in RL, we provide an extended discussion of related work in App. A.

2 POSTERIOR-AVERAGED GOAL OCCUPANCY MEASURE

This section introduces the posterior-averaged goal occupancy measure, the central quantity KERL targets, in four steps. We first review the value function of a policy conditioned on a commanded goal, then introduce the goal occupancy measure describing which goals that policy visits. Next, we define the posterior distribution over commanded goals given an observed state–action pair. Finally, we average the corresponding goal occupancy measures under this posterior to obtain the quantity our method learns. To illustrate the distinctions between these concepts, we accompany the definitions with a running example in Fig. 3. App. D provides a complementary tabular example with analytically computed values (Figs. 10 and 11). Figs. 1, 7, 18, 19 and 20 illustrate how the posterior-averaged goal occupancy measure approximated by KERL can effectively be recovered.

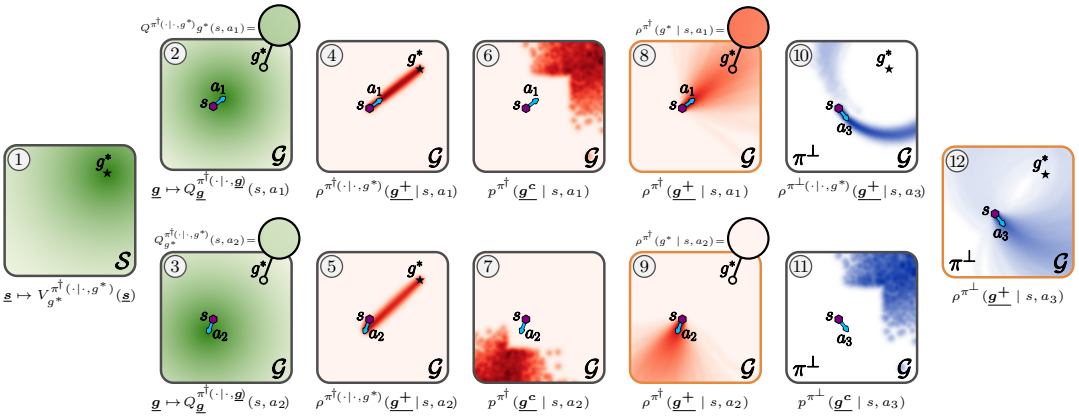

Figure 3: **Visual companion to Section 2.** We consider a 2D point environment, with $\mathcal{S} = \mathcal{G} = [-100, 100]^2$, and $\mathcal{A} = \mathbb{S}^1$ the unit circle. $\mathcal{P}$ maps state $s = (x, y)$ and action a to the Dirac at $s' = (x + cos(a), y + sin(a))$, $\gamma = 0.99$, and termination on success if $\|s' - g\|_2 \leq \varepsilon = 0.5$, or at $T = 200$. p_0 is uniform over $\mathcal{S}$ and p_g is uniform over $[-50, 50]^2 \setminus [-20, 20]^2$. We use two policies: $\pi^\dagger$ for the green and red plots and $\pi^\perp$ for the blue ones. $\pi^\dagger$ is a nearly optimal straight policy with small noise $\pi^\dagger(a \mid s, g) = \mathcal{N}(a \mid \mu = \texttt{atan2}(g - s), \sigma = 0.3)$. $\pi^\perp$ is a miscalibrated policy with an offset $\pi^\perp(a \mid s, g) = \pi^\dagger(a + \frac{\pi}{2} \mid s, g)$. Each panel corresponds to a specific function or measure, plotted over state space $\mathcal{S}$ for ① and goal space $\mathcal{G}$ for the others. In all cases the window is clipped to $[-40, 40]^2$. The panel captions highlight the changing variables in bold and underlined ($\underline{\boldsymbol{s}}$, $\underline{\boldsymbol{g}}$, $\underline{\boldsymbol{g^c}}$ and $\underline{\boldsymbol{g^+}}$). The other variables are fixed $s = (-8, 0)$, $g^* = (23, 23)$, $a_1 = 37°$, $a_2 = -110°$, $a_3 = -53°$. See Sec. C.5 for plotting details.

Problem setting. We consider the problem of goal-conditioned RL (Schaul et al., 2015), defined by a reward-free goal-conditioned Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, \mathcal{G}, \mathcal{P}, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A}$, goal space $\mathcal{G}$, transition dynamics $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to M_+^1(\mathcal{S})$, and discount factor $\gamma \in (0, 1)$, with $M_+^1(\mathcal{X})$ denoting the space of probability measures on $\mathcal{X}$. Following prior works (Huang et al., 2019; Zhang et al., 2021), we consider the setting where goals are defined from states using a state-to-goal mapping $\varphi : \mathcal{S} \to \mathcal{G}$. Desired goals are sampled from $p_g \in M_+^1(\mathcal{G})$ and initial states drawn from $p_0 \in M_+^1(\mathcal{S})$. The objective is to train a goal-conditioned policy $\pi : \mathcal{S} \times \mathcal{G} \to M_+^1(\mathcal{A})$ such that $\pi(\cdot \mid \cdot, g)$ drives the system toward states satisfying $\varphi(s) = g$. Throughout, g^c denotes the goal commanded to the policy when a trajectory is collected, g^+ a goal achieved later along that trajectory, and g the goal at which a critic is evaluated. In the illustrations, g^* is one fixed commanded goal. Distinguishing g^c from g^+ is what hindsight relabeling exploits, and what this section formalizes.

Value functions. Fig. 3 illustrates a specific MDP with two policies, $\pi^\dagger$ and $\pi^\perp$. Given a specific goal g^*, panel ① plots the value function of policy $\pi^\dagger(\cdot \mid \cdot, g^*)$ under the goal-reaching reward $r_{g^*}(s) = \mathbf{1}_{\{\|s - g^*\|_2 \leq \varepsilon\}}$.This is the expected return of $\pi^\dagger$ conditioned on g^* evaluated at reaching g^*: $\mathbb{E}_{\pi^\dagger(\cdot|\cdot, g^*)}\left[\sum_{t=0}^{\infty} \gamma^t r_{g^*}(s_t) \mid s_0 = s\right]$. As $\pi^\dagger$ mostly moves straight toward its commanded goal, this landscape is close to a γ-decaying exponential. Panels ② and ③ plot the state-action value function over the goal space $\mathcal{G}$ instead, for fixed state-action pairs and varying goals $\underline{g} \in \mathcal{G}$. Each point of these panels is the Q-value of a different policy $\pi^\dagger(\cdot \,|\, \cdot, \underline{g})$ under reward $r_{\underline{g}}(\cdot)$ (for additional details about the panels see Sec. C.5). They show the reachability of every goal from s after taking actions a_1 and a_2. The zooms in the top-right corners show the value of reaching goal g^*, highlighting the difficulty this paper addresses. Although a_1 is intuitively a better choice, its Q-value at g^* is nearly the same as that of a_2. The Q-function seems a poor way to compare actions, as it collapses everything that follows them into a single number for a given commanded goal.

Goal occupancy measure. Given commanded goal $g^c \in \mathcal{G}$ and state-action pair (s, a), the discounted goal occupancy measure induced by $\pi(\cdot \,|\, \cdot, g^c)$ is (Ho & Ermon, 2016; Zhang et al., 2020)

$$\rho^{\pi(\cdot|\cdot, g^c)}(g^+ \mid s, a) \triangleq (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t p_t^{\pi(\cdot|\cdot, g^c)}\big(\varphi(s_t) = g^+ \mid s, a\big), \tag{1}$$

where $p_t^{\pi(\cdot|\cdot, g^c)}(\cdot \,|\, s, a)$ is the probability density of the achieved goal after t time steps, starting from s, taking action a and following $\pi(\cdot \mid \cdot, g^c)$. It describes which goals are visited after (s, a) when the commanded goal is g^c. As for the successor measure (Dayan, 1993), it is defined for policies mapping states to actions, $S \to M_+^1(\mathcal{A})$, here $\pi(\cdot \mid \cdot, g^c)$ for a given g^c, **not** for a general policy $\pi : \mathcal{S} \times \mathcal{G} \to M_+^1(\mathcal{A})$. Panels ④ and ⑤ show the goal occupancy measures of $\pi^\dagger(\cdot \,|\, \cdot, g^*)$ at (s, a_1) and at (s, a_2). As for the Q-value, these panels are very similar, as $\pi^\dagger$ quickly recovers from a_2 when commanded by g^*. In particular, the density at g^* is close for both actions. As a complement,

panel ⑩ shows the goal occupancy measure of the miscalibrated policy $\pi^{\perp}(\cdot \mid \cdot, g^*)$ at (s, a_3), which circles around its commanded goal.

Posterior of commanded goals. $p^{\pi}(\cdot \mid s, a)$ is the posterior distribution over commanded goals given that the trajectory visited (s, a). Rollouts are collected forward: a goal g^c is drawn from p_g, an initial state from p_0, and $\pi(\cdot \mid \cdot, g^c)$ then selects the actions. The posterior $p^{\pi}(g^c \mid s, a)$ entails the inverse question: when sampling a pair (s, a) in the buffer, what commanded goals could have led the policy to (s, a)? Each trajectory is collected using one commanded goal, but a same pair (s, a) might appear under different commanded goals. Panels ⑥ and ⑦, for $\pi^{\dagger}$, the commanded goals likely to produce a visit to s followed by a_1 and a_2 are the ones ahead of each action. Panel ⑪, as $\pi^{\perp}$ turns on the right of its commanded goal, the mass of $p^{\pi^{\perp}}(\cdot \mid s, a_3)$ concentrates to the left of a_3.

The posterior-averaged goal occupancy measure. We can now define the core object of our approach, obtained by averaging the goal occupancy over the posterior of commanded goals,

$$\rho^{\pi}(g^+ \mid s, a) \triangleq \int_{\mathcal{G}} \rho^{\pi(\cdot \mid \cdot, g^c)}(g^+ \mid s, a) p^{\pi}(g^c \mid s, a) dg^c. \tag{2}$$

It mixes the occupancies induced by the policy for all goals that could have been commanded, weighted by their posterior given (s, a). The goal occupancy $\rho^{\pi(\cdot \mid \cdot, g^c)}$ is defined for one conditioning of the policy $\pi(\cdot \mid \cdot, g^c)$, whereas ρ^{π} depends on all possible conditioning $(\pi(\cdot \mid \cdot, g))_{g \in \mathcal{G}}$, on the transition dynamics of the MDP, and on the distributions p_g and p_0. To our knowledge, ρ^{π} has no established name, but it corresponds to the predominant sampling procedure in GCRL: draw (s, a) in the replay buffer, $t \sim \mathrm{Geom}(1-\gamma)$ and take $g^+ = \phi(s_t)$ observed t steps later along the same trajectory. The closest reference to ρ^{π} is Eysenbach et al. (2022, CRL) under the name *average stationary distribution* and notation $p^{\pi(\cdot \mid \cdot)}$, for a non goal-conditioned policy $\pi(a \mid s)$. Proposition 4.1 and App. B.3 discuss its relation to a state-wise average policy.

Panels ⑧ and ⑨ show $\rho^{\pi^{\dagger}}(\cdot \mid s, a_1)$ and $\rho^{\pi^{\dagger}}(\cdot \mid s, a_2)$ and close the argument of this section. The Q-values of a_1 and a_2 at g^* were nearly identical, and so were their goal occupancies. On the contrary, their posterior-averaged occupancies at g^*, $\rho^{\pi^{\dagger}}(g^* \mid s, a)$ are completely different, as the zooms in the top right corners show. What really distinguishes the two actions is not how well each reaches g^*, but which goals each leads to in the first place. Our approach, explained in the next section, captures this distribution by learning its kernel mean embedding, so we can evaluate the choice of action a at s by comparing the embedding of (s, a) to the one of g in the RKHS.

Two remarks help understand ρ^{π} correctly. First, $\rho^{\pi}(\cdot \mid s, a)$ describes the goals visited by (s, a), but it does **not** imply π knows how to reach these goals from (s, a). Panel ⑫, after (s, a_3), $\pi^{\perp}$ visits goals it would fail to reach if they were to be commanded. Even a policy that cannot reach any goal reliably might still provide evidence of how to reach the goals it visited. This motivates *hindsight relabeling*: leveraging a trajectory for the intermediate goals reached, not for the score of whether these were the ones asked. Second, $\rho^{\pi}(\cdot \mid s, a)$ does **not just** depend on which commanded goals might lead the policy to choose a once at s, that is $(\pi(\cdot \mid s, g^c))_{g^c \in \mathcal{G}}$, but also on the probability of arriving at s at all. Even in a state s where the policy always takes action a, independently of the commanded goal, $\rho^{\pi}(\cdot \mid s, a)$ can still be discriminative, with, for instance, only one commanded goal leading to s.

3 Learning $\rho^{\pi}(\cdot \mid s, a)$ via Kernel Mean Embedding

$\rho^{\pi}(\cdot \mid s, a)$ corresponds to the standard sampling procedure in HER-inspired pipelines: draw (s, a) from the replay buffer $\mathcal{B}$ of π's rollouts, $t \sim \mathrm{Geom}(1-\gamma)$ and take $g^+ = \varphi(s_t)$ observed t steps later. As we illustrate above, $\rho^{\pi}(g \mid s, a)$ depends heavily on a. This motivates learning an embedding of (s, a) that best captures the $\rho^{\pi}(\cdot \mid s, a)$ distribution, and only then comparing it against desired goals to select actions. For this we use kernel mean embeddings.

3.1 Kernel Mean Embeddings of $\rho^{\pi}(\cdot \mid s, a)$

A kernel is a symmetric positive-definite function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that measures similarities between data points. In our case, $\mathcal{X} = \mathcal{G}$, which we assume to be bounded, typically a subset of $\mathbb{R}^d$. By default, we use the Gaussian kernel $k_{\sigma}(x, x') = \exp(-\frac{1}{2\sigma^2}\|x - x'\|_2^2)$ and in the ablations we test Laplace and Cauchy kernels. Along kernel k, we consider its associated Repro-

ducing Kernel Hilbert Space (RKHS), the Hilbert space $\mathcal{H}$ of functions from $\mathcal{X}$ to $\mathbb{R}$ satisfying: $k(x,\cdot) \in \mathcal{H}$ and $f(x) = \langle f, k(x,\cdot)\rangle_{\mathcal{H}}$ for all $x \in \mathcal{X}$ and $f \in \mathcal{H}$ (Aronszajn, 1950). The canonical feature map $\psi : x \mapsto k(x,\cdot)$ allows the function $k(x,\cdot)$ to represent the data point x in the high (possibly infinite) dimensional feature space $\mathcal{H}$. Finally, kernel mean embedding extends the canonical feature map from individual points to probability measures. It represents a distribution by its expected feature representation in $\mathcal{H}$ (Smola et al., 2007; Berlinet & Thomas-Agnan, 2011):

$$\mu : M_+^1(\mathcal{X}) \to \mathcal{H}, \qquad \mathbb{P} \longmapsto \mu_{\mathbb{P}} = \int_{\mathcal{X}} \psi(x)\, d\mathbb{P}(x) = \mathbb{E}_{X\sim\mathbb{P}}\left[\psi(X)\right]. \tag{3}$$

We consider the kernel mean embedding of $\rho^\pi(\cdot \,|\, s,a)$:

$$\mu^\pi(s,a) \stackrel{\triangle}{=} \mu_{\rho^\pi(\cdot|s,a)} = \mathbb{E}_{g^+\sim\rho^\pi(\cdot|s,a)}\left[\psi(g^+)\right]. \tag{4}$$

Collapsing a distribution to a single element in $\mathcal{H}$ discards nothing when the kernel is *characteristic*, since $\mathbb{P} \mapsto \mu_{\mathbb{P}}$ is injective: $\|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|_{\mathcal{H}} = 0$ if and only if $\mathbb{P} = \mathbb{Q}$. This holds for the Gaussian, Laplace and Cauchy kernels (Steinwart, 2001; Sriperumbudur et al., 2011; Borgwardt et al., 2006). In our case, this means $\mu^\pi(s,a)$ captures all the information in $\rho^\pi(\cdot \mid s,a)$. Our method learns $\mu^\pi(s,a)$ and Sec. 4.2 shows that we can recover an approximation of $\rho^\pi(\cdot \,|\, s,a)$ from it.

3.2 Kernel Mean Embedding Approximation via Random Fourier Features

We introduce two levels of *features*: first, in the MDP definition in Sec. 2, φ is a mapping from states to goals we assume to be given. Then ψ is the canonical feature map that embeds goals into the infinite-dimensional RKHS. While this conceptual ψ grounds the theory, we use random Fourier features (RFF) as a practical approximation (Rahimi & Recht, 2007; Rudi & Rosasco, 2017). By Wendland (2004, Thm. 6.6), a continuous shift-invariant kernel admits the Fourier representation:

$$k(g,g') = \int_{\mathcal{G}} e^{\sqrt{-1}\,\omega^\top(g-g')}\, d\Lambda(\omega) = \mathbb{E}_{\omega\sim\Lambda}\big[\cos(\omega^\top(g-g'))\big], \tag{5}$$

with spectral distribution Λ. For a Gaussian kernel of bandwidth σ, $\Lambda_\sigma(\omega) \propto \exp(-\frac{1}{2}\sigma^2\|\omega\|_2^2)$. To avoid tuning the bandwidth, we construct a *multiscale kernel* $k_{\mathrm{MS}}(x,x') = \frac{1}{m}\sum_{\sigma\in B} k_\sigma(x,x')$ by averaging kernels of different bandwidths: $B = \{\sigma_1,\ldots,\sigma_n\} = \{100.0, 30.0, \ldots, 0.1, 0.03\}$ (see Fig. 9 in App. C.1). The corresponding spectral density is the mixture: $\Lambda_{\mathrm{MS}}(\omega) = \frac{1}{n}\sum_{\sigma\in B}\Lambda_\sigma(\omega)$.

Rahimi & Recht (2007, RFF) uses samples from Λ to construct the feature map:

$$\mathbf{z}(g) = \sqrt{\frac{1}{m}}\Big(\cos(\omega_1^\top g), \sin(\omega_1^\top g), \ldots, \cos(\omega_m^\top g), \sin(\omega_m^\top g)\Big)^\top, \qquad \omega_i \overset{\text{i.i.d.}}{\sim} \Lambda, \tag{6}$$

whose inner products form the unbiased estimator $\hat{k}(g,g') = \mathbf{z}(g)^\top\mathbf{z}(g')$ of $k(g,g')$. Alternatively:

$$\mathbf{z}(g) = \sqrt{\frac{2}{m}}\Big(\cos(\omega_1^\top g + b_1), \ldots, \cos(\omega_m^\top g + b_m)\Big)^\top, \quad (\omega_i, b_i) \overset{\text{i.i.d.}}{\sim} \Lambda \otimes \mathcal{U}([0,2\pi]). \tag{7}$$

We use the cosine bias variant by default (Eq. 7) and provide ablations using sine–cosine (Eq. 6) or random binning features Rahimi & Recht (2007), with similar results. We fixed m, the number of frequencies, to 1024, but our ablation study reveals that $m = 64$ is enough for the tasks considered. $\mathbf{z}$ is our practical goal embedding map, and leads to the estimator of $\mu^\pi(s,a)$:

$$\hat{\mu}^\pi(s,a) = \mathbb{E}_{g_+\sim\rho^\pi(\cdot|s,a)}\left[\mathbf{z}(g^+)\right]. \tag{8}$$

3.3 Concentration bounds

We define the following policy evaluation metrics:

$$\mathcal{K}^{\pi(\cdot|\cdot)}(s,a,g) = \langle \mu^\pi(s,a), \psi(g)\rangle_{\mathcal{H}}, \quad \mathcal{K}^{\pi(\cdot|\cdot)}_{\mathrm{MMD}}(s,a,g) = -\left\|\mu^\pi(s,a) - \psi(g)\right\|^2_{\mathcal{H}}, \tag{9}$$

and their approximations using feature map $\mathbf{z}$ instead of ψ: $\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}(s,a,g) = \left\langle \hat{\mu}^\pi(s,a), \mathbf{z}(g)\right\rangle_{\mathbb{R}^m}$ and $\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}_{\mathrm{MMD}}(s,a,g) = -\left\|\hat{\mu}^\pi(s,a) - \mathbf{z}(g)\right\|^2_{\mathbb{R}^m}$. As $\rho^\pi(\cdot \mid s,a)$ captures the posterior distribution of goals visited by π after (s,a), if a desired goal g is reliably achieved, then $\rho^\pi(\cdot \,|\, s,a)$ assigns a substantial probability mass to goals in the vicinity of g, and $\mu^\pi(s,a)$ has a large inner product with the feature map $\psi(g)$. Also, $-\mathcal{K}^{\pi(\cdot|\cdot)}_{\mathrm{MMD}}$ is the squared L^2 distance between the characteristic function of $\rho^\pi(\cdot \mid s,a)$ and the Dirac δ_g (Sriperumbudur et al., 2010; Muandet et al., 2017). Both metrics quantify how well the posterior of goals achieved after (s,a) aligns with the desired goal g. We now derive concentration bounds between the evaluation metrics (Eq. 9) and their approximations.

Lemma 3.1. *Let $k : \mathcal{G}\times\mathcal{G} \to \mathbb{R}$ be a normalized, continuous, shift-invariant positive-definite kernel, and $\Lambda \in \mathcal{M}_+^1(\mathcal{G})$ its spectral density, with $\sigma^2 = \int_{\mathcal{G}} \|\omega\|^2 \, d\Lambda(\omega) < +\infty$. Then, for all $\tau > 0$,*

$$\begin{aligned} \Pr\left(\sup_{s,a,g} \left|\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}(s,a,g) - \mathcal{K}^{\pi(\cdot|\cdot)}(s,a,g)\right| \geq \tfrac{h+\sqrt{2\tau}}{\sqrt{m}}\right) \leq e^{-\tau}, \\ \Pr\left(\sup_{s,a,g} \left|\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}_{\mathrm{MMD}}(s,a,g) - \mathcal{K}^{\pi(\cdot|\cdot)}_{\mathrm{MMD}}(s,a,g)\right| \geq 3\tfrac{h+\sqrt{2\tau}}{\sqrt{m}}\right) \leq e^{-\tau} \end{aligned} \tag{10}$$

where $h = 32\sqrt{2d\log(2|\mathcal{G}|+1)} + 32\sqrt{2d\log(\sigma+1)} + 16\sqrt{2d[\log(2|\mathcal{G}|+1)]^{-1}}$ depends on the dimension d of the goal space, its diameter $|\mathcal{G}|$, and the spectral width σ of Λ.

Lemma 3.1 establishes that both approximations $\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}$ and $\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}_{\mathrm{MMD}}$ converge uniformly to their true counterparts at a rate of $\mathcal{O}(1/\sqrt{m})$ as the number of random features m increases. These guarantees justify using the finite-dimensional RFF features $\mathbf{z}$, as they preserve the good properties of the high-dimensional RKHS embedding. The proof is in Appendix B.1 and holds for the sine–cosine features (Eq. 6), as it relies on the uniform convergence theorem for RFF from Sriperumbudur & Szabó (2015). For cosine bias (Eq. 7) or random binning feature maps, uniform convergence follows from Sutherland & Schneider (2015) and Rahimi & Recht (2007) but at a rate of $\mathcal{O}(\sqrt{m^{-1}\log(m)})$.

3.4 Policy training

We train the policy π_θ and critic μ_ω, parameterized by θ and ω respectively, within an actor–critic framework. The critic approximates the kernel mean embedding $\hat{\mu}^{\pi_\theta}$ (Eq. 8) by least-squares regression onto the features of goals sampled in $\rho^{\pi_\theta}(\cdot \,|\, s, a)$:

$$\min_{\omega} \; \mathcal{L}^{\mathcal{E}}(\omega) = \mathbb{E}_{(s,a)\sim\mathcal{B},\; t\sim\mathrm{Geom}(1-\gamma),\; g^+=\varphi(s_t)} \left[\left\|\mu_\omega(s,a) - \mathbf{z}(g^+)\right\|^2_{\mathbb{R}^m}\right]. \tag{11}$$

The actor maximizes the corresponding estimator of $\mathcal{K}^{\pi_\theta(\cdot|\cdot)}_{\mathrm{MMD}}$ (our default) or $\mathcal{K}^{\pi_\theta(\cdot|\cdot)}$, with entropy regularization (Haarnoja et al., 2018) ($\alpha = 0.5$ following Bortkiewicz et al. (2025)):

$$\max_{\theta} \mathcal{L}(\theta) = \mathbb{E}_{(s,g)\sim\mathcal{B},\; a\sim\pi_\theta(\cdot|s,g)} \left[\mathcal{K}^{\omega}_{\mathrm{MMD}}(s,a,g) - \alpha\log(\pi_\theta(a \mid s,g))\right] \tag{12}$$

$$\text{with} \quad \mathcal{K}^{\omega}(s,a,g) = \langle \mu_\omega(s,a), \mathbf{z}(g)\rangle^2_{\mathbb{R}^m} \quad \text{or} \quad \mathcal{K}^{\omega}_{\mathrm{MMD}}(s,a,g) = -\left\|\mu_\omega(s,a) - \mathbf{z}(g)\right\|^2_{\mathbb{R}^m}, \tag{13}$$

Alg. 1 presents the full KERL algorithm. The implementation resembles CRL, as it only requires changing the goal embedding to the RFF one instead of learning it jointly, and using a simple L^2-regression loss instead of a contrastive one. In terms of computational cost, KERL uses only 2 neural networks (μ_ω and π_θ) compared to 3 for CRL Eysenbach et al. (2022, ϕ, ψ, π_θ). KERL is a representation learning method like CRL, but with a fixed goal embedding $\mathbf{z}$. This feature should not be confused with the map φ we assume to be given. The latter is the topic studied by *successor features* methods and in zero-shot RL (Barreto et al., 2017; Blier et al., 2021).

4 Properties of $\mathcal{K}$ and Reconstruction of $\rho^\pi(\cdot \,|\, s, a)$

4.1 How does $\mathcal{K}$ relate to Q-functions?

Proposition 4.1. *For every $g \in \mathcal{G}$, $\mathcal{K}^{\pi(\cdot|\cdot)}(s,a,g)$ is the posterior average of the state-action value functions of the goal-conditioned policies $\{\pi(\cdot \,|\, \cdot, g^c)\}_{g^c\in\mathcal{G}}$ under reward $r_g(s) = (1-\gamma)k(\varphi(s), g)$:*

$$\mathcal{K}^{\pi(\cdot|\cdot)}(s,a,g) = \int_{\mathcal{G}} Q^{\pi(\cdot|\cdot,g^c)}_{r_g}(s,a) p^\pi(g^c \mid s,a) dg^c, \tag{14}$$

where

$$Q^{\pi(\cdot|\cdot,g^c)}_{r_g}(s,a) = \mathbb{E}_{\pi(\tau|g^c)}\left[\sum_{t=0}^{\infty} \gamma^t r_g(s_t) \;\middle|\; s_0 = s,\, a_0 = a\right] \tag{15}$$

is the state-action value function of the goal-conditioned policy $\pi(\cdot \,|\, \cdot, g^c)$ associated to reward r_g.

See App. B.2 for the proof. Eq. 14 mixes the Q-function of all policies $\{\pi(\cdot \,|\, \cdot, g^c)\}_{g^c\in\mathcal{G}}$ under the **same** reward r_g, it is **not** $Q^{\pi(\cdot|\cdot,g^c)}_{r_{g^c}}$. $\mathcal{K}^{\pi(\cdot|\cdot)}$ is generally not the value function of any individual $\pi(\cdot \,|\, \cdot, g^c)$, as it learns from experience collected under different commanded goals, as for Eysenbach et al. (2022); Ding et al. (2019). Additionally, we challenge a claim made in Eysenbach et al. (2022,

Algorithm 1: Kernel RL (KERL)

Require: Initialize policy π_θ, embeddings network μ_ω, rolling replay buffer $\mathcal{B}$

// Sample the random features of the multiscale kernel

for $i = 1$ ***to*** m **do**

  Sample $\sigma \sim \{\sigma_1, \dots, \sigma_n\}$ and $(\omega_i, b_i) \sim \Lambda_\sigma \otimes \mathcal{U}([0, 2\pi])$

function *RFF features* $\mathbf{z}(g)$

  return $\sqrt{\frac{2}{m}}\left(\cos(\omega_1^\top g + b_1), \dots, \cos(\omega_m^\top g + b_m)\right)^\top$ // Eq. 7

for *each training iteration* **do**

  // Data collection

  Sample $g^c \sim p_g$ and $s_0 \sim p_0$, rollout $\pi_\theta(\cdot \mid \cdot, g^c)$

  Store the (s, a) trajectories in $\mathcal{B}$ // Rolling buffer

  // Learning

  Sample $(s, a) \sim \mathcal{B}$, $t \sim \text{Geom}(1-\gamma)$ and $g^+ = \varphi(s_t)$ taken t steps later in $\mathcal{B}$

  $\omega \leftarrow w - \lambda_\omega \nabla_\omega \|\mu_\omega(s, a) - \mathbf{z}(g^+)\|_2^2$ // Eq. 11

  Sample $a_\theta \sim \pi_\theta(\cdot \mid s, g)$

  $\theta \leftarrow \theta + \lambda_\theta \nabla_\theta \left[\mathcal{K}^\pi_{\text{MMD}}(s, a_\theta, g) - \alpha \log(\pi_\theta(a_\theta \mid s, g))\right]$ // Eqs. 12 & 13

Algorithm 2: Estimating $\rho^\pi(\cdot \mid s, a)$ from $\mu_\omega(s, a)$ by solving a QP

Require: A pair (s, a), the estimate $\mu_\omega(s, a)$ and $\mathcal{G} \subset \mathbb{R}^d$ bounded

Discretize $\mathcal{G}$ in N box cells $\mathcal{G}_i$ of center c_i and side lengths $\Delta_{i,1}, \cdots, \Delta_{i,d}$

for $i = 1$ ***to*** N **do**

  $\bar{\mathbf{z}}_i^l = \frac{1}{\text{vol}(\mathcal{G}_i)} \int_{\mathcal{G}_i} \mathbf{z} = \sqrt{\frac{2}{m}} \cos(\omega_i^\top c_i + b_l) \prod_{r=1}^d \text{sinc}\left(\frac{\omega_i^r \Delta_{i,r}}{2}\right)$ // See App. C.2

$\mathbf{Z} = [\bar{\mathbf{z}}_1, \dots, \bar{\mathbf{z}}_N]$

// Ridge warm start

$\tilde{p}_0 \leftarrow \mathbf{Z}^\top (\mathbf{Z}\mathbf{Z}^\top + \lambda \mathbf{I})^{-1} \mu_\omega(s, a)$

$p_0 \leftarrow \frac{\max(\tilde{p}_0, \mathbf{0}) + \epsilon \mathbf{1}}{\mathbf{1}^\top [\max(\tilde{p}_0, \mathbf{0}) + \epsilon \mathbf{1}]}$

// Exponentiated Gradient

for $t = 1, \dots, T$ **do**

  $\mathbf{g}_t \leftarrow \mathbf{Z}^\top (\mathbf{Z}\, p_{t-1} - \mu_\omega(s, a))$

  $p_t \leftarrow p_{t-1} \odot \exp(-\eta\, \mathbf{g}_t)$

  $p_t \leftarrow p_t / (\mathbf{1}^\top p_t)$

return p_T // The final estimate $\hat{\rho}^\pi_h(g \mid s, a)$ Eq. 17

Lemma 4.1). It is true that if there exists a state-only policy $\bar{\pi} : \mathcal{S} \to M_+^1(\mathcal{A})$ whose goal occupancy measure is equal to $\rho^\pi(\cdot \mid s, a)$, then $\mathcal{K}^{\pi(\cdot|\cdot)}(s, a, g) = Q^{\bar{\pi}}_{r_g}(s, a)$. However, as detailed in App. B.3, $\bar{\pi}$ is generally **not** the state-wise average policy $\int_{\mathcal{G}} \pi(a \mid s, g^c) p^\pi(g^c \mid s) dg^c$. As for $\mathcal{K}^{\pi(\cdot|\cdot)}_{\text{MMD}}$, it inherits the above properties by extending the definition (Eq. 9):

$$\mathcal{K}^{\pi(\cdot|\cdot)}_{\text{MMD}}(s, a, g) = 2\mathcal{K}^{\pi(\cdot|\cdot)}(s, a, g) - \underbrace{\mathbb{E}_{g_1, g_2 \sim \rho^\pi(\cdot|s,a)}[k(g_1, g_2)]}_{\text{self-similarity penalty}} - \underbrace{k(g, g)}_{\text{constant}}. \tag{16}$$

$\mathcal{K}^{\pi(\cdot|\cdot)}_{\text{MMD}}$ depends on the geometry of ρ^π in the self-similarity term, and so favors dispersed actions.

4.2 Interpreting Learned Representations by Occupancy Reconstruction

Because we use a characteristic kernel k, $\mu^\pi(s, a)$ uniquely characterizes $\rho^\pi(\cdot \mid s, a)$. To visualize the distribution estimated by μ_ω in Fig. 1 and Section 5.4, we build an estimator of $\rho^\pi(\cdot \mid s, a)$. The idea is to discretize $\mathcal{G}$, and to estimate the average densities on the partitions by solving a quadratic programming (QP) problem. The interpretability algorithm is presented Alg. 2, it reveals the distribution estimated by the mean embedding $\mu_\omega(s, a)$. To solve the QP, we use the Exponentiated Gradient algorithm with a Ridge warm start. The proof of the following proposition is in App. B.4, it shows that this estimator has the same convergence rate as $\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}$ in Lemma 3.1, as the number of features m increases and the diameter of the discretization h decreases.

Proposition 4.2. *Let k satisfy Lemma 3.1, $\mathcal{G} = \bigsqcup_{i=1}^N \mathcal{G}_i$ be a partition of $\mathcal{G}$ with $\text{vol}(\mathcal{G}_i) > 0$ and $h = \max_i \text{diam}(\mathcal{G}_i)$, and set $\mathbf{Z} = [\bar{\mathbf{z}}_1, \dots, \bar{\mathbf{z}}_N]$, where $\bar{\mathbf{z}}_i = \frac{1}{\text{vol}(\mathcal{G}_i)} \int_{\mathcal{G}_i} \mathbf{z}(g)\, \text{d}g$. For (s, a), define:*

$$\hat{\rho}^\pi_h(g \mid s, a) \triangleq \sum_{i=1}^N \frac{\hat{p}^\pi_i(s, a)}{\text{vol}(\mathcal{G}_i)} \mathbb{1}_{\mathcal{G}_i}(g), \quad \textit{where} \quad \hat{p}^\pi(s, a) \in \underset{p \in \mathbb{R}_+^N,\ \mathbf{1}^\top p = 1}{\arg\min} \|\mathbf{Z}p - \hat{\mu}^\pi(s, a)\|_2^2 \tag{17}$$

then

$$\text{MMD}_k(\hat{\rho}^\pi_h(\cdot \mid s, a), \rho^\pi(\cdot \mid s, a)) \triangleq \left\|\mu_{\hat{\rho}^\pi_h(\cdot|s,a)} - \mu_{\rho^\pi(\cdot|s,a)}\right\|_{\mathcal{H}} \xrightarrow[h \to 0,\, m \to \infty]{\mathbb{P}} 0. \tag{18}$$

5 Experiments

We evaluate on JaxGCRL (Bortkiewicz et al., 2025), a Brax-based online GCRL benchmark. In total, we evaluate on 16 locomotion, navigation, and manipulation tasks, several of which prior state-of-the-art methods fail to solve. Following Wang et al. (2026), performance is the number of time steps spent near the goal during a 1,000-step episode, the *time-at-goal*. Scores are averaged over the last five epochs and five seeds. Network architectures follow Wang et al. (2026), hyperparameters are listed in App. C.4, and comparisons to additional methods are in App. E.1, Table. 3.

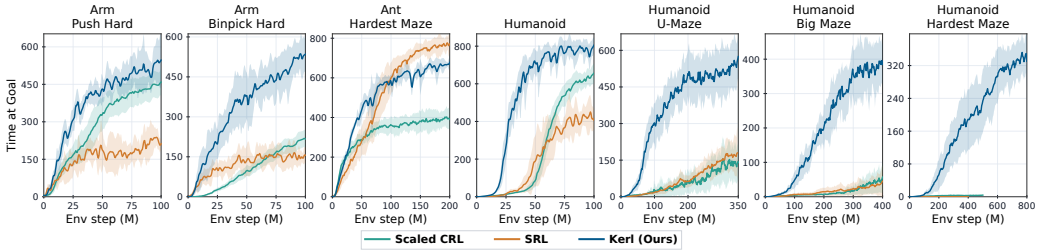

Figure 4: **Comparisons on JaxGCRL hardest tasks.** See Fig. 12 for other tasks.

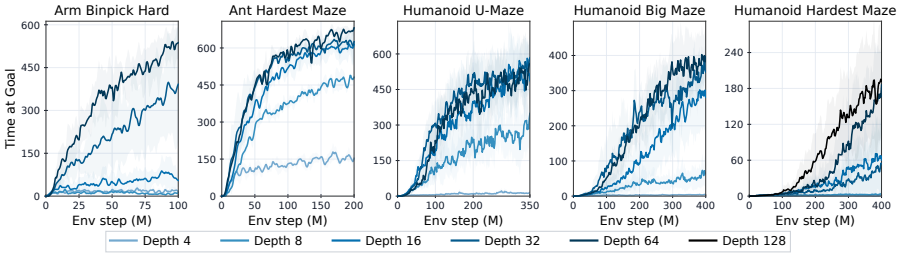

Figure 5: **Scaling network depth.** Performance improves consistently with depth across all challenging environments. See Fig. 14 for the five remaining environments.

5.1 New state of the art on challenging tasks

We compare KERL against scaled CRL (Wang et al., 2026) and scaled SRL (Nguimatsia-Tiofack et al., 2026), the two self-supervised GCRL methods shown to benefit consistently from increased network depth and competitive on these tasks. Fig. 4 compares the learning curves on the task marked `hard` and the humanoid suite, JaxGCRL's most challenging tasks, for additional tasks (16 in total), see Figs. 12 and 13. KERL matches or advances the state of the art, with large improvements on the hardest tasks: `humanoid-hardest-maze` (170×), `humanoid-big-maze` (6×), `humanoid-U-Maze` (4×), and `arm-binpick-hard` (2.5×). Fig. 2 summarizes the final scores of the hardest tasks and compared against SAC+HER. Beyond final scores, KERL is cheaper to run because its critic is a single encoder, whereas CRL uses two, as commented App. C.3.

5.2 Pushing the Frontier of Scalable Self-Supervised RL

We scale the actor and the kernel mean network jointly from 4 to 64 layers at a fixed hidden width of 256. All numbers in this subsection compare KERL against its own shallow version. Figs. 5 and 14 show that KERL benefits from depth on all ten environments. Relative to the 4-layer architecture, deeper networks yield gains of up to $100\times$ on the long-horizon, 268-dimensional humanoid navigation tasks, $1.5\times$ to $23\times$ on robotic manipulation, and $2\times$ to $4\times$ on the Ant mazes.

We pushed depth scaling to 128 layers on `humanoid-hardest-maze` in Figs. 4 and 5. A fixed 100-hour budget on one NVIDIA H100 allows KERL to perform 800M environment steps against 500M for CRL. At 128 layers, KERL reaches a time-at-goal of 340, whereas scaled CRL and SRL plateau at 2. Depth alone does not account for the gap, since CRL has been trained at up to 1000 layers by Wang et al. (2026). Next, we analyze what KERL gains from the added capacity.

5.3 Kernel Mean Network Depth Drives Performance Gains

We vary the actor and critic depths independently over $\{4, 8, 16, 32, 64\}$ on three tasks in Fig. 6. With network μ_ω fixed at 4 layers, scaling network π_θ changes little. Holding π_θ at 4 layers and scaling μ_ω instead yields an order-of-magnitude performance gain, from 15 to 811 on `humanoid`. Depth therefore has to go first into the kernel mean embedding network. The long-horizon task `ant-hardest-maze` shows that once μ_ω is deep enough, scaling actor depth increases performance from 348 to 665. This suggests two roles in scaling: first, critic depth has to be large enough to capture $\rho^\pi(\cdot \mid s, a)$, and then actor depth handles long horizons.

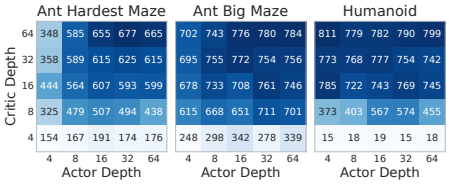

Figure 6: **Actor vs. critic network depth.**

5.4 Analysis of Posterior-Averaged Goal-Occupancy Modeling

We assess the effect of network depth on approximation quality by comparing posterior-averaged goal occupancies $\rho^{\pi_\theta}(\cdot \mid s_t, a_t)$ estimated using 4- and 64-layer networks μ_ω. We use Alg. 2 to produce the estimate $\hat{\rho}_h^{\pi_\theta}(\cdot \mid s_t, a_t)$ from $\mu_\omega(s_t, a_t)$, with a fine-grained discretization of the goal space (see App. C.2). Both networks are evaluated on the same three state-action pairs (s_t, a_t) generated by a near-optimal policy. As shown in Fig. 7, a shallow network yields myopic or misaligned predictions: high-probability regions remain clustered near the agent's current position or leak through

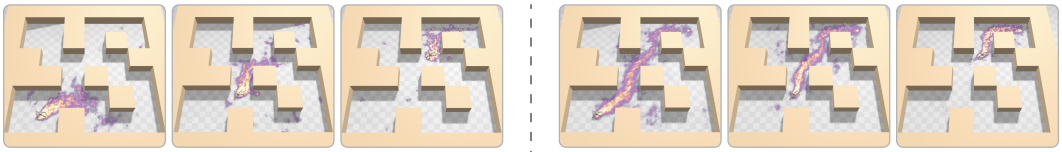

Figure 7: **Posterior-averaged goal-occupancy. Left (4 layers).** Shallow networks yield myopic, inaccurate estimates that fail to respect walls. **Right (64 layers).** Deeper kernel mean networks predict accurate, long-horizon goal distributions aligned with valid physical paths around obstacles.

impassable walls. Contrary, the deep architecture accurately models long-horizon goal distributions far into the future from a single state-action pair (for additional illustration, Figs. 1, 18, 19 and 20)

5.5 Ablation studies

Kernel methods raise a natural set of questions, and KERL is robust to most of them. Figs. 8 and 16 vary the kernel family and the feature map approximating ψ on `arm-push-hard`, `humanoid` and `ant-big-maze`. Gaussian, Laplace and Cauchy kernels perform equally well, as do the two RFF variants and random binning. However, the kernel bandwidth is not a free parameter, so for all our experiments we used a fixed multiscale kernel averaging over eight scales (see App. C.1). The number of random frequencies m is an approximation parameter with a visible threshold. Below this threshold, the representation cannot retain the distribution: at $m \in \{8, 16\}$ learning collapses on all three tasks. Beyond that, the curves are close with a modest edge for larger m, still, we fix $m = 1024$ as it does not cost much more. Fig. 17 in App. E.3 studies effect of increasing m as we increase networks' depth. The remaining choice is which evaluation metric in Eq. 13 the actor maximizes. $\mathcal{K}^{\omega}_{\text{MMD}}$ performs better on `ant-big-maze` (780 against 612) and `humanoid` (799 against 735) and weaker on `arm-push-hard` (563 against 659).

6 Conclusion

In this work, we provide a precise characterization of the posterior-averaged goal occupancy measure underlying hindsight relabeling and uses it to ground action evaluation. The resulting method, KERL, learns its kernel mean embedding by regressing onto fixed random Fourier features. For characteristic kernels, this embedding uniquely determines the distribution, while concentration bounds quantify approximation error. This representation supports both similarity- and distance-based actor objectives and approximate occupancy reconstruction, and the critic's predictions are fully interpretable. On JaxGCRL, KERL outperforms the state-of-the-art baselines scaled CRL and scaled SRL on 7 of 10 tasks, achieving 170× their time-at-goal on `humanoid-hardest-maze`. Extensive ablations attribute these gains primarily to kernel mean embedding network depth, and reconstructed occupancies show how deeper networks capture feasible long-horizon outcomes.

Limitations. Further progress will require addressing the dependence of approximation fidelity on the kernel and feature count, and testing beyond simulated tasks with prescribed goal representations. Finally, we hope that learning expressive, interpretable distributions of future outcomes will contribute to scaling RL more broadly, by clarifying which representations benefit most from increased model capacity.

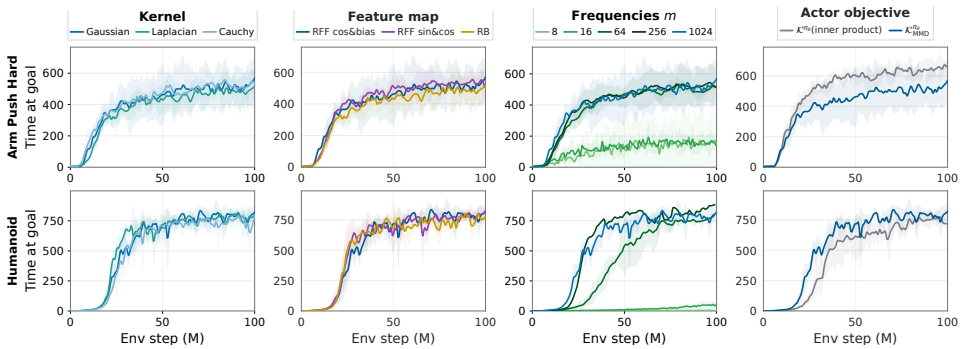

Figure 8: **Ablations.** Rows are environments, columns design choices. Mean over 5 seeds.

AI USE STATEMENT

In this work, we used generative AI tools for:

- Polish writing: we first wrote the paper by hand, including all sections and paragraphs. Then we used LLM feedback to polish the grammar and ensure idiomatic phrasing.
- Code to generate figures: we wrote the simulation script and the matplotlib plots for Fig. 3 by hand, then visually tuned them using LLM feedback. For the other figures, we used an LLM to adapt our scripts. In particular, for the reconstruction figures (Figs. 1, 7, 18, 19 and 20), we first made the code to generate the distribution (see next point) and asked an LLM to adapt Brax's HTML visualizer to insert it.
- The implementation of Alg. 1 was adapted by hand from prior work. For the implementation of Alg. 2, we used an LLM to draft the exponentiated gradient part (less than 100 lines of code), and three authors reviewed it carefully.
- For the related work, we asked an LLM to suggest papers we were not aware of, and then read all the papers we cite and checked for plagiarism through a manual literature survey.

We did not use generative AI tools for ideation or experimentation. In particular:

- We did **not** use an LLM to generate ideas presented in this work, and we conducted the experiments manually.
- Likewise, we made the proofs ourselves.
- As explained above, we did not use LLMs to draft sections of the paper.

Generating synthetic datasets does not apply to this work.

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 share all needed technical and theoretical details. All the hyperparameters used are listed explicitly throughout the paper and summarized in Table 2. Algs. 1 and 2 are written explicitly in the main paper to help understand and implement the method. Our code implementation is adapted from prior work, essentially Wang et al. (2026). We will make it open source upon paper acceptance. All our theoretical claims are proven in App. B: proof of Lemma 3.1 is in App. B.1, proof of Prop. 4.1 is in App. B.2; the claim made about the state-wise average policy is detailed in App. B.3, finally, the proof of Prop. 4.2 is in App. B.4.

REFERENCES

Siddhant Agarwal, Harshit Sikchi, Peter Stone, and Amy Zhang. Proto successor measure: Representing the behavior space of an rl agent. *arXiv preprint arXiv:2411.19418*, 2024.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Raghuram Mandyam Annasamy and Katia Sycara. Towards better interpretability in deep q-networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 4561–4569, 2019.

Nachman Aronszajn. Theory of reproducing kernels. *Transactions of the American mathematical society*, 68(3):337–404, 1950.

J Andrew Bagnell and Jeff Schneider. Policy search in kernel hilbert space. Technical Report CMU-RI-TR-03-45, Robotics Institute, Carnegie Mellon University, 2003.

Andre Barreto, Doina Precup, and Joelle Pineau. Reinforcement learning using kernel-based stochastic factorization. *Advances in Neural Information Processing Systems*, 24, 2011.

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

Alain Berlinet and Christine Thomas-Agnan. *Reproducing kernel Hilbert spaces in probability and statistics*. Springer Science & Business Media, 2011.

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

Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006.

Michał Bortkiewicz, Władysław Pałucki, Vivek Myers, Tadeusz Dziarmaga, Tomasz Arczewski, Łukasz Kuciński, and Benjamin Eysenbach. Accelerating goal-conditioned reinforcement learning algorithms and research. In *International Conference on Learning Representations*, volume 2025, pp. 15732–15754, 2025.

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

Henry Charlesworth and Giovanni Montana. Plangan: Model-based planning with sparse rewards and multiple goals. *Advances in Neural Information Processing Systems*, 33:8532–8542, 2020.

Boyuan Chen, Chuning Zhu, Pulkit Agrawal, Kaiqing Zhang, and Abhishek Gupta. Self-supervised reinforcement learning that transfers using random features. *Advances in Neural Information Processing Systems*, 36:56411–56436, 2023.

Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. In *International Conference on Machine Learning*, pp. 844–853. PMLR, 2017.

Sayak Ray Chowdhury and Aditya Gopalan. Online learning in kernelized markov decision processes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 3197–3205. PMLR, 2019.

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

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

Nicolae Dinculeanu. *Vector integration and stochastic integration in Banach spaces*. John Wiley & Sons, 2000.

Yiming Ding, Carlos Florensa, Pieter Abbeel, and Mariano Phielipp. Goal-conditioned imitation learning. *Advances in neural information processing systems*, 32, 2019.

Alexey Dosovitskiy and Vladlen Koltun. Learning to act by predicting the future. *arXiv preprint arXiv:1611.01779*, 2016.

Yaakov Engel, Shie Mannor, and Ron Meir. Bayes meets bellman: The gaussian process approach to temporal difference learning. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pp. 154–161, 2003.

Yaakov Engel, Shie Mannor, and Ron Meir. Reinforcement learning with gaussian processes. In *Proceedings of the 22nd international conference on Machine learning*, pp. 201–208, 2005.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Russ R Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems*, 35:35603–35620, 2022.

Jesse Farebrother, Jordi Orbay, Quan Vuong, Adrien Ali Taïga, Yevgen Chebotar, Ted Xiao, Alex Irpan, Sergey Levine, Pablo Samuel Castro, Aleksandra Faust, et al. Stop regressing: Training value functions via classification for scalable deep rl. *arXiv preprint arXiv:2403.03950*, 2024.

Yihao Feng, Tongzheng Ren, Ziyang Tang, and Qiang Liu. Accountable off-policy evaluation with kernel bellman statistics. In *International Conference on Machine Learning*, pp. 3102–3111. PMLR, 2020.

Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *2016 IEEE International conference on robotics and automation (ICRA)*, pp. 512–519. IEEE, 2016.

Kenji Fukumizu, Arthur Gretton, Bernhard Schölkopf, and Bharath K Sriperumbudur. Characteristic kernels on groups and semigroups. *Advances in Neural Information Processing Systems*, 21, 2008.

Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. *arXiv preprint arXiv:1912.06088*, 2019.

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

Steffen Grunewalder, Guy Lever, Luca Baldassarre, Massi Pontil, and Arthur Gretton. Modelling transition dynamics in mdps with rkhs embeddings. *arXiv preprint arXiv:1206.4655*, 2012.

Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Bernardo A Pires, and Rémi Munos. Neural predictive belief representations. *arXiv preprint arXiv:1811.06407*, 2018.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/haarnoja18b.html.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Zhiao Huang, Fangchen Liu, and Hao Su. Mapping state space using landmarks for universal goal reaching. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Michael Janner, Igor Mordatch, and Sergey Levine. Gamma-models: Generative temporal difference learning for infinite-horizon prediction. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1724–1735. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/12ffb0968f2f56e51a59a6beb37b2859-Paper.pdf.

Swaminathan S K, Damiya Gondha, Theyanesh Eswaramoorthy Rajahkrishnan, and Aritra Hazra. Direction-conditioned policies via compositional subgoal scoring for online goal-conditioned reinforcement learning. In *2nd Workshop on Compositional Learning: Safety, Interpretability, and Agents*, 2026. URL https://openreview.net/forum?id=WksFWDQXY6.

Purushottam Kar and Harish Karnick. Random feature maps for dot product kernels. In *Artificial intelligence and statistics*, pp. 583–591. PMLR, 2012.

Kaiqiang Ke, Qian Lin, Zongkai Liu, Shenghong He, and Chao Yu. Conservative offline goal-conditioned implicit v-learning. In *International Conference on Machine Learning*, pp. 29591–29607. PMLR, 2025.

George Konidaris, Sarah Osentoski, and Philip Thomas. Value function approximation in reinforcement learning using the fourier basis. In *Proceedings of the AAAI conference on artificial intelligence*, volume 25, pp. 380–385, 2011.

Michal Korniak, Kamil Dybek, Benjamin Eysenbach, Marco Bagatella, and Michał Bortkiewicz. Three steps at a time: Learning representations from action sequences in contrastive rl, 2026. URL https://arxiv.org/abs/2608.30640.

Tejas D Kulkarni, Ardavan Saeedi, Simanta Gautam, and Samuel J Gershman. Deep successor reinforcement learning. *arXiv preprint arXiv:1606.02396*, 2016.

Sascha Lange and Martin Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *The 2010 international joint conference on neural networks (IJCNN)*, pp. 1–8. IEEE, 2010.

Tuyen Pham Le, Vien Anh Ngo, P Marlith Jaramillo, and TaeChoong Chung. Importance sampling policy gradient algorithms in reproducing kernel hilbert space. *Artificial Intelligence Review*, 52 (3):2039–2059, 2019.

Adam Lee, M Ganesh Kumar, Blake Bordelon, and Cengiz Pehlevan. Completep for RL: Maintaining feature learning when scaling deep reinforcement learning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=3uXAGWqCny.

Hojoon Lee, Dongyoon Hwang, Donghu Kim, Hyunseung Kim, Jun Jet Tai, Kaushik Subramanian, Peter Wurman, Jaegul Choo, Peter Stone, and Takuma Seno. Simba: Simplicity bias for scaling up parameters in deep reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 46816–46848, 2025.

Guy Lever and Ronnie Stafford. Modelling policies in mdps in reproducing kernel hilbert space. In *Artificial intelligence and statistics*, pp. 590–598. PMLR, 2015.

Guopeng Li, Yiyang Duan, Yiru Jiao, and Chengcheng Xu. Arrive and survive: Scaling safe goal-conditioned policy learning from one-bit failure signals, 2026. URL https://arxiv.org/abs/2608.26571.

Yitao Liang, Marlos C Machado, Erik Talvitie, and Michael Bowling. State of the art control of atari games using shallow reinforcement learning. *arXiv preprint arXiv:1512.01563*, 2015.

Russell Mendonca, Oleh Rybkin, Kostas Daniilidis, Danijar Hafner, and Deepak Pathak. Discovering and achieving goals via world models. *Advances in neural information processing systems*, 34:24379–24391, 2021.

Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017.

Vivek Myers, Catherine Ji, and Benjamin Eysenbach. Horizon generalization in reinforcement learning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=BH8Nrt2dPf.

Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample efficient continuous control. *Advances in neural information processing systems*, 37:113038–113071, 2024.

Franki Nguimatsia-Tiofack, Fabian Schramm, Théotime Le Hellard, and Justin Carpentier. Survival reinforcement learning: Toward scalable self-supervised rl. *arXiv preprint arXiv:2605.31273*, 2026.

Thanh Tang Nguyen, Sunil Gupta, and Svetha Venkatesh. Distributional reinforcement learning with maximum mean discrepancy. *Association for the Advancement of Artificial Intelligence (AAAI)*, 2020.

Thanh Nguyen-Tang, Sunil Gupta, and Svetha Venkatesh. Distributional reinforcement learning via moment matching. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 9144–9152, 2021.

Dirk Ormoneit and Śaunak Sen. Kernel-based reinforcement learning. *Machine learning*, 49(2): 161–178, 2002.

Asim Osman, Sasha Abramowitz, Mark Bergh, Ulrich Armel Mbou Sob, Ruan John de Kock, Omayma Mahjoub, Oussama Hidaoui, Noah De Nicola, Arnol Manuel Fokam, Felix Chalumeau, et al. Self-supervised on-policy reinforcement learning via contrastive proximal policy optimisation. *arXiv preprint arXiv:2605.13554*, 2026.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. Hiql: Offline goal-conditioned rl with latent states as actions. *Advances in Neural Information Processing Systems*, 36:34866–34891, 2023.

Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes rl scalable. *Advances in Neural Information Processing Systems*, 38:8350–8389, 2025.

Santiago Paternain, Juan Andrés Bazerque, Austin Small, and Alejandro Ribeiro. Stochastic policy gradient ascent in reproducing kernel hilbert spaces. *IEEE Transactions on Automatic Control*, 66(8):3429–3444, 2020.

Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 239–247, 2013.

Joaquin Quinonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of machine learning research*, 6(Dec):1939–1959, 2005.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing solving sparse reward tasks from scratch. In *International conference on machine learning*, pp. 4344–4353. PMLR, 2018.

Alessandro Rudi and Lorenzo Rosasco. Generalization properties of learning with random features. *Advances in neural information processing systems*, 30, 2017.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pp. 1312–1320. PMLR, 2015.

Alex Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A hilbert space embedding for distributions. In *International conference on algorithmic learning theory*, pp. 13–31. Springer, 2007.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Bharath Sriperumbudur and Zoltán Szabó. Optimal rates for random fourier features. *Advances in neural information processing systems*, 28, 2015.

Bharath K Sriperumbudur, Arthur Gretton, Kenji Fukumizu, Bernhard Schölkopf, and Gert RG Lanckriet. Hilbert space embeddings and metrics on probability measures. *The Journal of Machine Learning Research*, 11:1517–1561, 2010.

Bharath K Sriperumbudur, Kenji Fukumizu, and Gert RG Lanckriet. Universality, characteristic kernels and rkhs embedding of measures. *Journal of Machine Learning Research*, 12(7), 2011.

Ingo Steinwart. On the influence of the kernel on the consistency of support vector machines. *Journal of machine learning research*, 2(Nov):67–93, 2001.

Felipe Petroski Such, Vashisht Madhavan, Rosanne Liu, Rui Wang, Pablo Samuel Castro, Yulun Li, Jiale Zhi, Ludwig Schubert, Marc G Bellemare, Jeff Clune, et al. An atari model zoo for analyzing, visualizing, and comparing deep reinforcement learning agents. *arXiv preprint arXiv:1812.07069*, 2018.

Danica J Sutherland and Jeff Schneider. On the error of random fourier features. *arXiv preprint arXiv:1506.02785*, 2015.

S K Swaminathan, Damiya Gondha, Theyanesh Eswaramoorthy Rajahkrishnan, and Aritra Hazra. Direction-conditioned policies via compositional subgoal scoring for online goal-conditioned reinforcement learning. In *2nd Workshop on Compositional Learning: Safety, Interpretability, and Agents (ICML)*, 2026. arXiv:2606.16515.

Gavin Taylor and Ronald Parr. Kernelized value function approximation for reinforcement learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 1017–1024, 2009.

Franki Nguimatsia Tiofack, Fabian Schramm, Théotime Le Hellard, and Justin Carpentier. SVL: Goal-conditioned reinforcement learning as survival learning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=qIOcJSCGn2.

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? *arXiv preprint arXiv:2209.14935*, 2022.

Michal Valko, Nathaniel Korda, Rémi Munos, Ilias Flaounas, and Nelo Cristianini. Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*, 2013.

Herke Van Hoof, Gerhard Neumann, and Jan Peters. Non-parametric policy search with limited information loss. *Journal of Machine Learning Research*, 18(73):1–46, 2017.

Ngo Anh Vien, Peter Englert, and Marc Toussaint. Policy search in reproducing kernel hilbert space. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pp. 2089–2096, 2016.

Han Wang, Erfan Miahi, Martha White, Marlos C Machado, Zaheer Abbas, Raksha Kumaraswamy, Vincent Liu, and Adam White. Investigating the properties of neural network representations in reinforcement learning. *Artificial Intelligence*, 330:104100, 2024.

Kevin Wang, Ishaan Javali, Michał Bortkiewicz, Tomasz Trzcinski, and Benjamin Eysenbach. 1000 layer networks for self-supervised rl: Scaling depth can enable new goal-reaching capabilities. *Advances in Neural Information Processing Systems*, 38:157643–157670, 2026.

Holger Wendland. *Scattered data approximation*, volume 17. Cambridge university press, 2004.

Xin Xu, Dewen Hu, and Xicheng Lu. Kernel-based least squares policy iteration for reinforcement learning. *IEEE transactions on neural networks*, 18(4):973–992, 2007.

Ge Yang, Anurag Ajay, and Pulkit Agrawal. Overcoming the spectral bias of neural value approximation. *arXiv preprint arXiv:2206.04672*, 2022.

Zhuoran Yang, Chi Jin, Zhaoran Wang, Mengdi Wang, and Michael Jordan. Provably efficient reinforcement learning with kernel and neural function approximations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 13903–13916. Curran Associates, Inc., 2020a. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/9fa04f87c9138de23e92582b4ce549ec-Paper.pdf.

Zhuoran Yang, Chi Jin, Zhaoran Wang, Mengdi Wang, and Michael Jordan. Provably efficient reinforcement learning with kernel and neural function approximations. *Advances in Neural Information Processing Systems*, 33:13903–13916, 2020b.

Lunjun Zhang, Ge Yang, and Bradly C Stadie. World model as a graph: Learning latent landmarks for planning. In *International conference on machine learning*, pp. 12611–12620. PMLR, 2021.

Shangtong Zhang, Bo Liu, and Shimon Whiteson. Gradientdice: Rethinking generalized offline estimation of stationary values. In *International Conference on Machine Learning*, pp. 11194–11203. PMLR, 2020.

Yang Zhang, Jiangyuan Zhao, Chenyou Fan, Fangzheng Yan, Tian Li, Haitong Tang, Sen Fu, Xuan'er Wu, Qizhen Weng, Weinan Zhang, et al. Prts: A primitive reasoning and tasking system via contrastive representations. *arXiv preprint arXiv:2604.27472*, 2026a.

Yilang Zhang, Bingcong Li, Niao He, and Georgios B Giannakis. Ancre: Adaptive neural connection reassignment for efficient depth scaling. *arXiv preprint arXiv:2602.09009*, 2026b.

Menghui Zhu, Minghuan Liu, Jian Shen, Zhicheng Zhang, Sheng Chen, Weinan Zhang, Deheng Ye, Yong Yu, Qiang Fu, and Wei Yang. Mapgo: Model-assisted policy optimization for goal-oriented tasks. *arXiv preprint arXiv:2105.06350*, 2021.

APPENDIX CONTENT

A Related work **17**

B Proof of paper results **19**

B.1 Proof of Lemma 3.1 19

B.2 Proof of Proposition 4.1 19

B.3 State-wise average policy 20

B.4 Proof of Proposition 4.2 20

C Method and implementation details **21**

C.1 Details about the kernels considered 21

C.2 Recovery algorithm details 21

C.3 Computation time 22

C.4 Hyperparameters 22

C.5 Details about the illustrative figure 22

D Illustration with a tabular goal-conditioned MDP **23**

E Additional results **26**

E.1 Results on other environments and comparison to more methods 26

E.2 Additional scaling experiments 27

E.3 Additional ablation studies 27

E.4 Additional analysis of posterior-averaged goal occupancies 28

A RELATED WORK

Representation learning. Learning representations that facilitate value estimation and control has long been a central theme in reinforcement learning (RL). A large body of work combines an auxiliary representation-learning objective with a conventional RL objective, typically temporal-difference (TD) learning (Annasamy & Sycara, 2019; Liang et al., 2015; Finn et al., 2016; Guo et al., 2018; Lange & Riedmiller, 2010). In goal-conditioned RL (GCRL), this paradigm is often combined with hindsight experience replay (HER) to alleviate sparse-reward challenges (Andrychowicz et al., 2017; Riedmiller et al., 2018; Park et al., 2023; Ke et al., 2025; Park et al., 2025; Tiofack et al., 2026). Other approaches learn models of the dynamics or reachability structure to guide goal-directed behavior (Dosovitskiy & Koltun, 2016; Mendonca et al., 2021; Charlesworth & Montana, 2020; Zhu et al., 2021). A distinct line of work formulates GCRL as a data-driven representation-learning problem rather than directly as reward maximization. In particular, Contrastive RL (CRL) (Eysenbach et al., 2022) jointly learns goal-conditioned representations and control objectives through contrastive learning. KERL belongs to this latter family in that its representation is itself tied directly to the control problem, rather than being learned through a separate auxiliary objective. However, unlike CRL, KERL does not learn an arbitrary latent goal embedding. Instead, it explicitly represents the posterior-average goal-occupancy measure in a characteristic RKHS. This formulation yields a simple regression objective and, importantly, makes the learned representation interpretable: because characteristic kernel mean embeddings uniquely identify distributions, the represented occupancy distribution can be recovered and visualized. This contrasts with the common practice of interpreting learned RL features through low-dimensional projections such as PCA or t-SNE (Annasamy & Sycara, 2019; Eysenbach et al., 2022; Such et al., 2018; Wang et al., 2024; 2026), which can obscure information contained in the original representation.

Successor representations and measures. KERL is also related to successor representations and successor measures (Dayan, 1993; Barreto et al., 2017; Blier et al., 2021; Touati & Ollivier, 2021; Touati et al., 2022). Successor methods encode the discounted distribution of future states or state-action pairs visited under a policy, and are commonly used to enable transfer or zero-shot adaptation across a family of reward functions (Kulkarni et al., 2016; Barreto et al., 2017; Janner et al., 2020; Agarwal et al., 2024). In contrast, KERL represents a posterior-averaged *goal*-occupancy measure. It neither targets the future state-action visitation distribution nor seeks a universal representation for optimizing a class of downstream reward functions. Rather, its occupancy embedding constitutes both the learned representation and the learning objective for goal-conditioned control.

Kernel methods in reinforcement learning. In parallel, kernel-based methods have been extensively studied in RL. Early work employed kernels for value-function approximation and kernelized TD (Ormoneit & Sen, 2002; Engel et al., 2003; 2005; Xu et al., 2007; Taylor & Parr, 2009; Barreto et al., 2011; Lever & Stafford, 2015; Feng et al., 2020). Other work has considered policy optimization in RKHS (Bagnell & Schneider, 2003; Vien et al., 2016; Lever & Stafford, 2015; Le et al., 2019; Paternain et al., 2020), used kernel assumptions to establish theoretical guarantees for RL algorithms (Vien et al., 2016; Yang et al., 2020a;b), used kernel-based metrics for distributional RL (Nguyen et al., 2020; Nguyen-Tang et al., 2021), or represented MDP transition dynamics using kernel mean embeddings (Grunewalder et al., 2012; Van Hoof et al., 2017).

Kernel mean embeddings in KERL. Our use of kernels differs from these approaches in two aspects. First, the kernel mean embedding is not merely a function approximator or an auxiliary tool for improving a standard RL objective: it directly represents the posterior-averaged goal-occupancy measure that defines KERL's control objective. Second, KERL uses random Fourier features to approximate the kernel embedding, retaining the relevant theoretical properties while practical and scaling to high-dimensional observations and long-horizon control problems. This distinguishes KERL from much of the earlier kernel-RL literature, which primarily emphasizes theoretical analysis or is evaluated in comparatively limited settings.

Random features in reinforcement learning. More broadly, a variety of handcrafted, learned, and random features have been used in RL for linear value approximation, improved TD learning, exploration, and transfer (Konidaris et al., 2011; Kulkarni et al., 2016; Burda et al., 2018; Chen et al., 2023). Random Fourier features have also been used to mitigate spectral bias in TD learning (Yang et al., 2022). These methods are complementary to KERL, which uses random features specifically to construct a scalable and interpretable embedding of goal occupancy. Additionally, kernel methods have likewise been extensively explored in contextual and multi-armed bandits, where they support principled algorithms and regret guarantees (Srinivas et al., 2009; Valko et al., 2013; Chowdhury & Gopalan, 2017; 2019); this setting is orthogonal to our focus.

Scaling reinforcement learning. Finally, recent work has investigated whether scaling network capacity and data can substantially improve RL performance (Farebrother et al., 2024; Nauman et al., 2024; Lee et al., 2025). While generic scaling often produces modest gains, several recent GCRL methods, including contrastive approaches (Wang et al., 2026; Zhang et al., 2026b; Swaminathan et al., 2026; Osman et al., 2026) and survival-based objectives (Nguimatsia-Tiofack et al., 2026)-report notable improvements with increased scale. KERL advances this direction by combining competitive scaling behavior with an explicitly interpretable representation. On the most challenging benchmarks, KERL achieves substantial improvements while providing a fully interpretable learned representation which further enables analysis of why increasing network depth can unlock goal-reaching capabilities.

B PROOF OF PAPER RESULTS

This integral in Eq. 3 is defined in the sense of Bochner (Dinculeanu, 2000), exists for bounded kernels by Muandet et al. (2017, Lemma 3.1).

B.1 PROOF OF LEMMA 3.1

Lets $(s, a, g) \in \mathcal{S} \times \mathcal{A} \times \mathcal{G}$

$$\begin{aligned}\left|\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}(s,a,g) - \mathcal{K}^{\pi(\cdot|\cdot)}(s,a,g)\right| &= \left|\mathbb{E}_{g^+\sim\rho^\pi(\cdot|s,a)}\left[\mathbf{z}(g^+)^\top \mathbf{z}(g) - \langle\psi(g^+),\psi(g)\rangle_{\mathcal{H}}\right]\right| \\ &= \left|\mathbb{E}_{g^+\sim\rho^\pi(\cdot|s,a)}\left[\hat{k}(g^+,g) - k(g^+,g)\right]\right| \\ &\leq \sup_{g_1,g_2\in\mathcal{G}}\left|\hat{k}(g_1,g_2) - k(g_1,g_2)\right|\end{aligned}$$

Additionally, using Eq. 16, we have:

$$\begin{aligned}&\left|\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}_{\text{MMD}}(s,a,g) - \mathcal{K}^{\pi(\cdot|\cdot)}_{\text{MMD}}(s,a,g)\right| \\ &= \left|2\left(\hat{\mathcal{K}}^{\pi(\cdot|\cdot)}(s,a,g) - \mathcal{K}^{\pi(\cdot|\cdot)}(s,a,g)\right) - \mathbb{E}_{g_1,g_2\sim\rho^\pi(\cdot|s,a)}\left[\hat{k}(g_1,g_2) - k(g_1,g_2)\right] - \underbrace{\hat{k}(g,g) - k(g,g)}_{1-1=0}\right| \\ &\leq 3\sup_{g_1,g_2\in\mathcal{G}}\left|\hat{k}(g_1,g_2) - k(g_1,g_2)\right|\end{aligned}$$

The concentration bounds in 10 then follow by applying the uniform convergence theorem for RFF from (Sriperumbudur & Szabó, 2015, Theorem 1).

B.2 PROOF OF PROPOSITION 4.1

By definition of the kernel mean embedding,

$$\begin{aligned}\mathcal{K}^{\pi(\cdot|\cdot)}(s,a,g) &= \left\langle \mu_{\rho^\pi(\cdot|s,a)}, \psi(g)\right\rangle_{\mathcal{H}} \\ &= \int_{\mathcal{G}} k(g^+,g)\rho^\pi(g^+ \mid s,a)dg^+ \\ &= \int_{\mathcal{G}} k(g^+,g)\left[\int_{\mathcal{G}} \rho^{\pi(\cdot|\cdot,g^c)}(g^+ \mid s,a)p^\pi(g^c \mid s,a)\,dg^c\right]dg^+ \\ &= \int_{\mathcal{G}}\left[\int_{\mathcal{G}} k(g^+,g)\rho^{\pi(\cdot|\cdot,g^c)}(g^+ \mid s,a)dg^+\right]p^\pi(g^c \mid s,a)\,dg^c\end{aligned}$$

Using the definition of the discounted goal occupancy measure induced by the policy $\pi(\cdot \mid \cdot, g^c)$,

$$\begin{aligned}\int_{\mathcal{G}} k(g^+,g)\rho^{\pi(\cdot|\cdot,g^c)}(g^+ \mid s,a)dg^+ &= (1-\gamma)\mathbb{E}_{\pi(\tau|g^c)}\left[\sum_{t=0}^{\infty}\gamma^t k(\varphi(s_t),g) \mid s_0 = s,\ a_0 = a\right] \\ &= Q^{\pi(\cdot|\cdot,g^c)}_{r_g}(s,a).\end{aligned}$$

Therefore,

$$\mathcal{K}^{\pi(\cdot|\cdot)}(s,a,g) = \int_{\mathcal{G}} Q^{\pi(\cdot|\cdot,g^c)}_{r_g}(s,a)\,p^\pi(g^c \mid s,a)\,dg^c.$$

■

B.3 STATE-WISE AVERAGE POLICY

Let us denote the state wise average policy by: $\tilde{\pi}(a \mid s) = \int_{\mathcal{G}} \pi(a \mid s, g^c) p^\pi(g^c \mid s) dg^c$ and suppose that $\tilde{\pi}(a \mid s) > 0$ for all $(s, a) \in \mathcal{S} \times \mathcal{A}$. We know that $\mathcal{K}^{\pi(\cdot|\cdot)}(\cdot, \cdot, g)$ is the state-action value function of $\tilde{\pi}$ for the reward r_g if and only if $\mathcal{T}^{\tilde{\pi}} \mathcal{K}^{\pi(\cdot|\cdot)}(s, a, g) = \mathcal{K}^{\pi(\cdot|\cdot)}(s, a, g)$ for $(s, a) \in \mathcal{S} \times \mathcal{A}$. Using the identity $\tilde{\pi}(a \mid s) p^\pi(g^c \mid s, a) = \pi(a \mid s, g^c) p^\pi(g^c \mid s)$, we have:

$$
\begin{aligned}
\mathcal{T}^{\tilde{\pi}} \mathcal{K}^{\pi(\cdot|\cdot)}(s, a, g) &= r_g(s) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|,s,a)} \left[\int_{\mathcal{A}} \tilde{\pi}(a' \mid s') \mathcal{K}^{\pi(\cdot|\cdot)}(s', a', g) da' \right] \\
&= r_g(s) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|,s,a)} \left[\int_{\mathcal{A}} \int_{\mathcal{G}} Q_{r_g}^{\pi(\cdot|\cdot,g^c)}(s', a') \tilde{\pi}(a' \mid s') p^\pi(g^c \mid s', a') dg^c da' \right] \\
&= r_g(s) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|,s,a)} \left[\int_{\mathcal{A}} \int_{\mathcal{G}} Q_{r_g}^{\pi(\cdot|\cdot,g^c)}(s', a') \pi(a' \mid s', g^c) p^\pi(g^c \mid s') dg^c da' \right] \\
&= r_g(s) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|,s,a)} \left[\int_{\mathcal{G}} \left(\int_{\mathcal{A}} Q_{r_g}^{\pi(\cdot|\cdot,g^c)}(s', a') \pi(a' \mid s', g^c) da' \right) p^\pi(g^c \mid s') dg^c \right] \\
&= r_g(s) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|,s,a)} \left[\int_{\mathcal{G}} V_{r_g}^{\pi(\cdot|\cdot,g^c)}(s') p^\pi(g^c \mid s') dg^c \right]
\end{aligned}
$$

Where $\mathcal{P}(\cdot \mid s, a)$ is the transition dynamics. And

$$
\begin{aligned}
\mathcal{K}^{\pi(\cdot|\cdot)}(s, a, g) &= \int_{\mathcal{G}} Q_{r_g}^{\pi(\cdot|\cdot,g^c)}(s, a) p^\pi(g^c \mid s, a) dg^c \\
&= \int_{\mathcal{G}} \left(r_g(s) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \left[V_{r_g}^{\pi(\cdot|\cdot,g^c)}(s') \right] \right) p^\pi(g^c \mid s, a) dg^c \\
&= r_g(s) + \gamma \mathbb{E}_{s' \sim \mathcal{P}(\cdot|s,a)} \left[\int_{\mathcal{G}} V_{r_g}^{\pi(\cdot|\cdot,g^c)}(s') p^\pi(g^c \mid s, a) dg^c \right]
\end{aligned}
$$

Therefore, $\mathcal{K}^{\pi(\cdot|\cdot)}(\cdot, \cdot, g)$ if the value function of $\tilde{\pi}$ for the reward r_g if and only if:

$$
\mathbb{E}_{s' \sim p(\cdot|s,a)} \left[\int_{\mathcal{G}} \left(p^\pi(g^c \mid s, a) - p^\pi(g^c \mid s') \right) V_{r_g}^{\pi(\cdot|\cdot,g^c)}(s') dg^c \right] = 0 \tag{19}
$$

Equality 19 is generally not satisfied unless we impose additional conditions, for example, a strong posterior-consistency condition, e.g $p^\pi(g^c \mid s, a) = p^\pi(g^c \mid s')$ for $\mathcal{P}(\cdot \mid s, a)-$almost every s' in $\mathcal{S}$. That is, a policy for which the distributions of commanded goals that could lead to (s, a) is the same as the one of commanded goals leading to s'. It is for instance not the case in the illustrative MDP, Fig. 3.

■

B.4 PROOF OF PROPOSITION 4.2

$\mathbf{Z}$ is a matrix where each column is the integral of the feature map $\mathbf{z}$ over the partition G_i: $\mathbf{Z} = [\bar{\mathbf{z}}_1, \ldots, \bar{\mathbf{z}}_N]$, where $\bar{\mathbf{z}}_i = \frac{1}{\mathrm{vol}(\mathcal{G}_i)} \int_{\mathcal{G}_i} \mathbf{z}(g) \, \mathrm{d}g$

For $p \in \Delta_{N-1} = \{p \in \mathbb{R}_+^N : p^\top \mathbf{1} = 1\}$, lets $\rho_p = \sum_{i=1}^N \frac{p_i}{\mathrm{vol}(\mathcal{G}_i)} \mathbb{1}_{\mathcal{G}_i} \in \mathcal{M}_+^1(\mathcal{G})$. Then

$$
\hat{\mu}_{\rho_p} = \mathbb{E}_{g \sim \rho_p}[\mathbf{z}(g)] = \mathbf{Z}p, \quad \text{and} \qquad \mathrm{MMD}_{\hat{k}}\big(\rho_p, \rho^\pi(\cdot \mid s, a)\big) = \|\mathbf{Z}p - \hat{\mu}^\pi(s, a)\|_{\mathbb{R}^m}.
$$

Since (the QP Alg. 2 solves)

$$
\hat{p}^\pi(s, a) \in \operatorname*{arg\,min}_{p \in \Delta_{N-1}} \|\mathbf{Z}p - \hat{\mu}^\pi(s, a)\|_2^2,
$$

we have

$$
\hat{\rho}_h^\pi(\cdot \mid s, a) = \rho_{\hat{p}^\pi(s,a)} \in \operatorname*{arg\,min}_{p \in \Delta_{N-1}} \mathrm{MMD}_{\hat{k}}\big(\rho_p, \rho^\pi(\cdot \mid s, a)\big). \tag{20}
$$

Let

$$
\epsilon_m = \sup_{g_1, g_2 \in \mathcal{G}} \left| k(g_1, g_2) - \mathbf{z}(g_1)^\top \mathbf{z}(g_2) \right|, \qquad \Pi_h \rho^\pi(\cdot \mid s, a) = \sum_{i=1}^N \frac{\rho^\pi(\mathcal{G}_i \mid s, a)}{\mathrm{vol}(\mathcal{G}_i)} \mathbb{1}_{\mathcal{G}_i}.
$$

Using $\big|\mathrm{MMD}_k(\mu,\nu) - \mathrm{MMD}_{\hat{k}}(\mu,\nu)\big| \leq 2\sqrt{\epsilon_m}$ (by developing the norm as in Sec. B.1) and by Eq. 20 $\hat{\rho}^\pi_h(\cdot \mid s,a)$ minimizes the MMD with $\rho^\pi(\cdot \mid s,a)\big)$, so as a smaller one than $\Pi_h\rho^\pi(\cdot \mid s,a)$:

$$\begin{aligned}\mathrm{MMD}_k\big(\hat{\rho}^\pi_h(\cdot \mid s,a), \rho^\pi(\cdot \mid s,a)\big) &\leq 2\sqrt{\epsilon_m} + \mathrm{MMD}_{\hat{k}}\big(\hat{\rho}^\pi_h(\cdot \mid s,a), \rho^\pi(\cdot \mid s,a)\big)\\ &\leq 2\sqrt{\epsilon_m} + \mathrm{MMD}_{\hat{k}}\big(\Pi_h\rho^\pi(\cdot \mid s,a), \rho^\pi(\cdot \mid s,a)\big)\\ &\leq 4\sqrt{\epsilon_m} + \mathrm{MMD}_k\big(\Pi_h\rho^\pi(\cdot \mid s,a), \rho^\pi(\cdot \mid s,a)\big),\end{aligned}$$

Let $b^\pi_{(s,a)}(h) = \mathrm{MMD}_k\big(\Pi_h\rho^\pi(\cdot \mid s,a), \rho^\pi(\cdot \mid s,a)\big)$. It remains to show that $b^\pi_{(s,a)}(h) \leq \sqrt{2\big(k(0)-k(h)\big)} \xrightarrow[h\to 0]{} 0$.

Let $X \sim \rho^\pi(\cdot \mid s,a)$ and define $I \in \{1,\dots,N\}$ by $I = i$ whenever $X \in \mathcal{G}_i$. Conditionally on $I = i$, let Y be uniformly distributed on $\mathcal{G}_i$. Then $Y \sim \Pi_h\rho^\pi(\cdot \mid s,a)$ and, since $\mathrm{diam}(\mathcal{G}_i) \leq h$,

$$\|X - Y\| \leq h \qquad \text{almost surely.}$$

Therefore,

$$\begin{aligned}\big(b^\pi_{(s,a)}(h)\big)^2 &= \big\|\mathbb{E}\big[\psi(X) - \psi(Y)\big]\big\|^2_{\mathcal{H}}\\ &\leq \mathbb{E}\left[\|\psi(X) - \psi(Y)\|^2_{\mathcal{H}}\right]\\ &\leq \sup_{\|g_1 - g_2\| \leq h} \|\psi(g_1) - \psi(g_2)\|^2_{\mathcal{H}}\\ &= 2\big(k(0) - k(h)\big).\end{aligned}$$

Consequently,

$$\mathrm{MMD}_k\big(\hat{\rho}^\pi(\cdot \mid s,a), \rho^\pi(\cdot \mid s,a)\big) \leq 4\sqrt{\epsilon_m} + \sqrt{2\big(k(0)-k(h)\big)}.$$

Finally, the uniform convergence result for random Fourier features (Sriperumbudur & Szabó, 2015; Sutherland & Schneider, 2015; Rahimi & Recht, 2007) yields $\epsilon_m \xrightarrow[m\to\infty]{\mathbb{P}} 0$, which concludes the proof.

■

Note: Proposition 4.2, as well as most of the analysis in this work, is stated for a continuous bounded goal space $\mathcal{G} \subset \mathbb{R}^d$, with Vol denoting the Lebesgue measure. But it can be extend to a discrete finite goal space by replacing the Lebesgue measure by the counting measure i.e $\mathrm{Vol}(\mathcal{G}_i) := |\mathcal{G}_i|$, $\mathrm{diam}(\mathcal{G}_i) = \max_{g_1,g_2} \|g_1 - g_2\|$ and $\bar{\mathbf{z}}_i = \frac{1}{|\mathcal{G}_i|}\sum_{g\in\mathcal{G}_i} \bar{\mathbf{z}}(g)$.

C Method and implementation details

C.1 Details about the kernels considered

By default, we use the Gaussian kernel, and in the ablations (Figs. 8 and 16) we show that the Laplace and Cauchy ones yield similar performance. These kernels and their spectral distributions are presented in Table B.3. To avoid tuning the bandwidth for each task, we construct a *multiscale kernel*, and we fix the bandwidths to $B = \{\sigma_1,\dots,\sigma_n\} = \{100.0, 30.0, 10.0, 3.0, 1.0, 0.3, 0.1, 0.03\}$. Fig. 9 plots the kernels considered.

C.2 Recovery algorithm details

In Alg. 2 from Sec. 4.2, to generate Figs. 1 and 7, we set $\Delta_{i,x} = \Delta_{i,y} \approx 0.01$ to partition $\mathcal{G}$ in rectangular cells, and $\lambda = 0.001$ for the Ridge warm start step. $\mathbf{Z}$ is a matrix where each column is the integral of the feature map $\mathbf{z}$ over the partition G_i: $\mathbf{Z} = [\bar{\mathbf{z}}_1,\dots,\bar{\mathbf{z}}_N]$, where $\bar{\mathbf{z}}_i = \frac{1}{\mathrm{vol}(\mathcal{G}_i)}\int_{\mathcal{G}_i} \mathbf{z}(g)\,\mathrm{d}g$. This quantity can be computed analytically. In particular, for the cosine-bias features (Eq. 7), and partition $\mathcal{G}_i$ of side lengths $\Delta_{i,1},\cdots,\Delta_{i,d}$ and center c_i:

$$\bar{\mathbf{z}}^l_i = \sqrt{\frac{2}{m}}\cos(\omega_l^\top c_i + b_l)\prod_{r=1}^{d}\mathrm{sinc}\left(\frac{\omega^r_l \Delta_{i,r}}{2}\right), \quad \text{where} \quad \mathrm{sinc}(x) = \begin{cases}\dfrac{\sin(x)}{x} & \text{for } x \neq 0\\ 1 & \text{for } x = 0\end{cases}$$

Table 1: **Uni-scale kernel and their corresponding spectral density distribution**, more examples can be found in Quinonero-Candela & Rasmussen (2005); Rahimi & Recht (2007); Fukumizu et al. (2008); Kar & Karnick (2012); Pham & Pagh (2013); Muandet et al. (2017).

Kernel	$k(\mathbf{x}, \mathbf{x}')$	$\Lambda(\boldsymbol{\omega})$
Gaussian	$\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|_2^2}{2\sigma^2}\right),\ \sigma > 0$	$\frac{1}{(2\pi/\sigma^2)^{d/2}} \exp\left(-\frac{\sigma^2\|\boldsymbol{\omega}\|_2^2}{2}\right)$
Laplace	$\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|_1}{\sigma}\right),\ \sigma > 0$	$\prod_{i=1}^{d} \frac{\sigma}{\pi(1+\omega_i^2)}$
Cauchy	$\prod_{i=1}^{d} \frac{1}{1+(x_i-x_i')^2/\sigma}$	$\exp\left(-\sigma\|\boldsymbol{\omega}\|_1\right)$

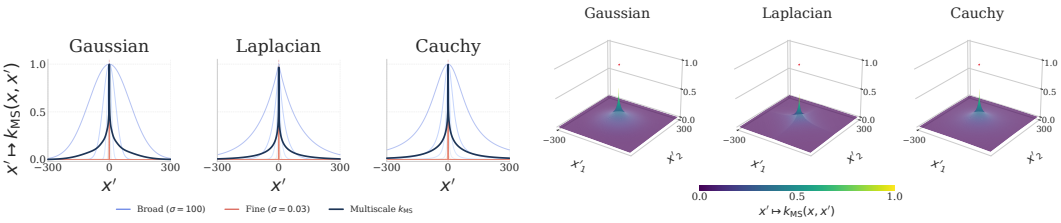

Figure 9: **Multiscale kernels.** Gaussian, Laplacian, and Cauchy kernels mixed across 8 log-spaced bandwidths, from broad ($\sigma = 100$) to fine ($\sigma = 0.03$). *Left:* 1D slices $x' \mapsto k_{\text{MS}}(x, x')$ around a fixed point $x = 0$. *Right:* the corresponding 2D surfaces around $x = (0, 0)$.

C.3 Computation time

We ran our experiments on different GPUs depending on task requirements. Specifically, for the *humanoid* suite, where we ran our most extensive experiments, we used one NVIDIA H100. On `humanoid` with depth 64, 100M training steps took 6.97 hours for KERL; on `humanoid-U-maze` 400M steps took 24.96 hours; on 400M steps `humanoid-big-maze` took 27.33 hours. Ideally, these numbers could be compared to those reported in Wang et al. (2026, Table 3) for scaled CRL (10.99h on `humanoid`, 46.74h on `humanoid-U-maze`, and 50.33h on `humanoid-big-maze`), but we did not find any reference to the GPU used. The only controlled comparison we can make is using the experiments we ran on `humanoid-hardest-maze` at depth 128: KERL took an average of 12.90h per 100M steps, while scaled CRL took 18.92h, so KERL was 1.46× faster. This empirical ratio matches the theory: at each training iteration, CRL and KERL collect the same number of transitions using the simulator, and then KERL trains 2 neural networks (μ_ω and π_θ) compared to 3 for CRL Eysenbach et al. (2022, ϕ, ψ, π_θ).

$$\frac{\textit{time CRL}}{\textit{time KERL}} \sim \frac{3 + \textit{data collection}}{2 + \textit{data collection}}$$

C.4 Hyperparameters

Except when explicitly stated otherwise, we kept all hyperparameters fixed at the default values introduced throughout the paper and summarized in Table 2.

C.5 Details about the illustrative figure

To generate Fig. 3, we simulated the described MDP and used 10.000 state-action pairs matching the given (s, a_i) (difference under 0.0001) to plot the goals that follow. The plots take 1 minute to generate on a regular computer. For these plots (but not for Prop. 4.1 nor the rest of the paper), the value functions $V_g^{\pi(\cdot|\cdot,g)}$ and $Q_g^{\pi(\cdot|\cdot,g)}$ are defined by:

$$V_g^{\pi(\cdot|\cdot,g)}(s) = \mathbb{E}_{\pi(\cdot|\cdot,g)}\left[\sum_{t=0}^{\infty} \gamma^t r_g(s_t) \mid s_0 = s\right] \tag{21}$$

Table 2: **Summary of KERL hyperparameters used in all experiments**, as listed throughout the paper (regular text) or kept as in Wang et al. (2026, Table 7) (texttt style).

Hyperparameter	Value
Training timesteps	100M-800M (depending on the task)
Network depth	4-128 (depending on the experiment)
Network width	256 (ablation: 181, 362, 512, 724)
Kernel k	Gaussian (ablation: Laplace, Cauchy)
Random features $\mathbf{z}$	Cosine bias (ablation: sine–cosine, random binning)
Multiscale bandwidths B	$\{100.0, 30.0, 10.0, 3.0, 1.0, 0.3, 0.1, 0.03\}$
Number of frequencies m	1024 (ablation: 8, 16, 64, 256)
Actor objectif	$\mathcal{K}^{\omega}_{\mathrm{MMD}}$ (ablation: $\mathcal{K}^{\omega}$)
Entropy regularization α	0.5
`update-to-data (UTD) ratio`	1:40
`max_replay_size`	10,000
`max_replay_size`	1,000
`episode_length`	1,000
`discounting` γ	0.99
`num_envs`	512
`batch_size`	512
`policy_lr`	3e-4
`critic_lr`	3e-4

$$Q_g^{\pi(\cdot|\cdot,g)}(s,a) = \mathbb{E}_{\pi(\cdot|\cdot,g)}\left[\sum_{t=0}^{\infty} \gamma^t r_g(s_t) \mid s_0 = s, a_0 = a\right] \tag{22}$$

with

$$r_g(s) = \mathbf{1}_{\{\|s-g\|_2 \le \varepsilon\}}.$$

Panels ①, ② and ③ illustrate these quantities for $\pi = \pi^\dagger$ and $g = g^*$.

D ILLUSTRATION WITH A TABULAR GOAL-CONDITIONED MDP

This section illustrates the quantities we have introduced in Sec. 2 and used in the paper, using a small stochastic goal-conditioned MDP where we compute all distributions and values exactly via linear algebra. The example emphasizes that $Q^\pi(\bar{s}, \bar{a}, \cdot)$ and $\rho^\pi(\cdot \mid \bar{s}, \bar{a})$ are different objects: the former is a goal-indexed value function under a fixed commanded goal, whereas the latter is a probability distribution over future achieved goals after marginalizing the unknown commanded goal. To lighten notation, we write h for the commanded goal g^c.

Environment. We consider a 5×5 obstacle-free grid. Every cell is both a state and a possible commanded or achieved goal:

$$\mathcal{S} = \mathcal{G}, \qquad |\mathcal{S}| = |\mathcal{G}| = 25.$$

The achieved-goal map is the identity,

$$\varphi(s) = s.$$

The action set is

$$\mathcal{A} = \{\mathrm{Up}, \mathrm{Right}, \mathrm{Down}, \mathrm{Left}, \mathrm{Stay}\}.$$

Both the initial-state distribution and the commanded-goal prior are uniform: $p_0(s) = p_g(h) = \frac{1}{25}$.

The dynamics are stochastic with reflecting boundaries. For a directional action, the intended displacement is executed with probability $1 - \varepsilon$, and each perpendicular displacement is executed with probability $\varepsilon/2$. In the experiments, $\varepsilon = 0.2$. For the action Stay, the agent remains in place with probability $1 - \varepsilon$ and executes each of the four directional displacements with probability $\varepsilon/4$. Any displacement leaving the grid is reflected, i.e., the agent remains in its current state. We denote the resulting dynamics by $P(s' \mid s, a)$. The discount factor is set to $\gamma = 0.9$.

Goal-conditioned policy families. The optimal policy family is computed separately for every commanded goal $h \in \mathcal{G}$ by value iteration for the goal-reaching reward. At every state, it is uniform over all maximizing actions:

$$\pi^\star(a \mid s, h) = \frac{\mathbf{1}\{a \in \arg\max_{b\in\mathcal{A}} Q^\star(s, b, h)\}}{|\arg\max_{b\in\mathcal{A}} Q^\star(s, b, h)|}.$$

Thus, the optimal policy is deterministic whenever the maximizing action is unique and stochastic only at ties. We also consider a suboptimal stochastic family obtained by mixing the optimal policy with the uniform action distribution:

$$\pi^{\mathrm{sub}}(a \mid s, h) = (1-\eta)\pi^\star(a \mid s, h) + \frac{\eta}{|\mathcal{A}|}, \qquad \eta = 0.2.$$

Policy-induced transition matrix. For a fixed commanded goal h, the Markov transition matrix induced by $\pi(\cdot \mid \cdot, h)$ is

$$P_h^\pi(s, s') = \sum_{a\in\mathcal{A}} \pi(a \mid s, h) P(s' \mid s, a).$$

All distributions below are represented as row vectors. Since the state space is finite and $\gamma < 1$, the discounted resolvent $(I - \gamma P_h^\pi)^{-1}$ exists.

Discounted state occupancy. Conditional on the commanded goal $H = h$, define the normalized discounted state occupancy

$$d_h^\pi(x) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t p_t^{\pi(\cdot\mid\cdot,h)}(s_t = x).$$

It is a probability distribution over states and satisfies

$$d_h^\pi = (1-\gamma) p_0 (I - \gamma P_h^\pi)^{-1}.$$

The corresponding discounted state-action occupancy is

$$d_h^\pi(x, a) = d_h^\pi(x)\pi(a \mid x, h).$$

Posterior over commanded goals. Conditioning on the observed pair $(\bar{s}, \bar{a})$ gives

$$p^\pi(h \mid \bar{s}, \bar{a}) = \frac{p_g(h)\, d_h^\pi(\bar{s})\, \pi(\bar{a} \mid \bar{s}, h)}{\sum_{\tilde{h}\in\mathcal{G}} p_g(\tilde{h})\, d_{\tilde{h}}^\pi(\bar{s})\, \pi(\bar{a} \mid \bar{s}, \tilde{h})}.$$

Conditional future-goal distribution. After taking the observed action $\bar{a}$ once at $\bar{s}$, the policy subsequently follows $\pi(\cdot \mid \cdot, h)$. Let $P_{\bar{a}}(\bar{s}, \cdot)$ denote the row of the environment transition matrix corresponding to tacking $\bar{a}$ at $\bar{s}$. The normalized discounted future-state distribution, conditional on $H = h$, is

$$\rho^{\pi(\cdot\mid\cdot,h)}(\cdot \mid \bar{s}, \bar{a}) = (1-\gamma) P_{\bar{a}}(\bar{s}, \cdot)(I - \gamma P_h^\pi)^{-1}.$$

Posterior-averaged future-goal distribution. The central distributional quantity marginalizes the unknown commanded goal:

$$\rho^\pi(g \mid \bar{s}, \bar{a}) = \sum_{h\in\mathcal{G}} p^\pi(h \mid \bar{s}, \bar{a})\, \rho^{\pi(\cdot\mid\cdot,h)}(g \mid \bar{s}, \bar{a}).$$

Equivalently,

$$\rho^\pi(\cdot \mid \bar{s}, \bar{a}) = \sum_{h\in\mathcal{G}} p^\pi(h \mid \bar{s}, \bar{a})\, \rho_h^\pi(\cdot \mid \bar{s}, \bar{a}).$$

Thus $\rho^\pi(\cdot \mid \bar{s}, \bar{a})$ is a probability distribution over future achieved goals, obtained by averaging future occupancies under the posterior distribution of commanded goals.

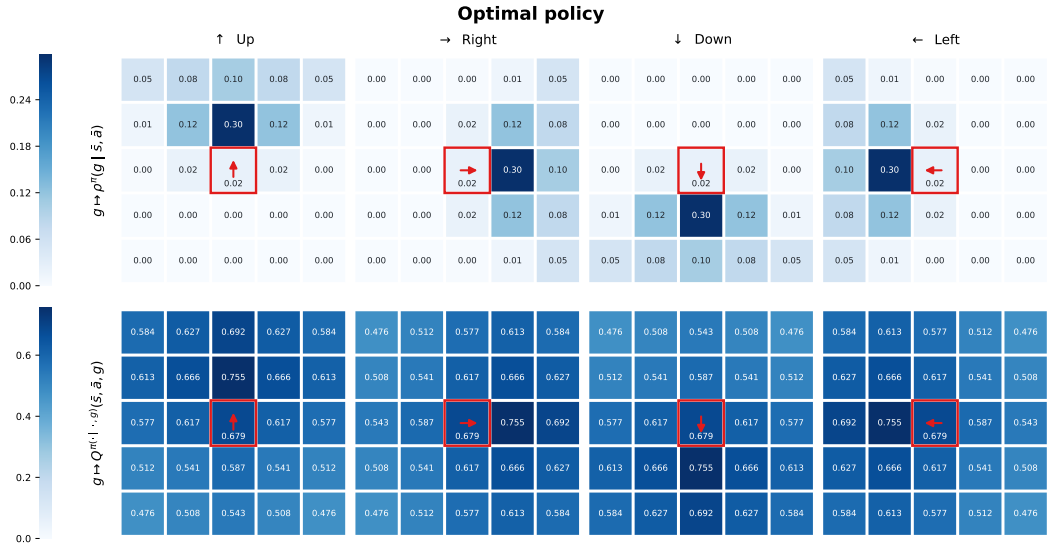

Figure 10:

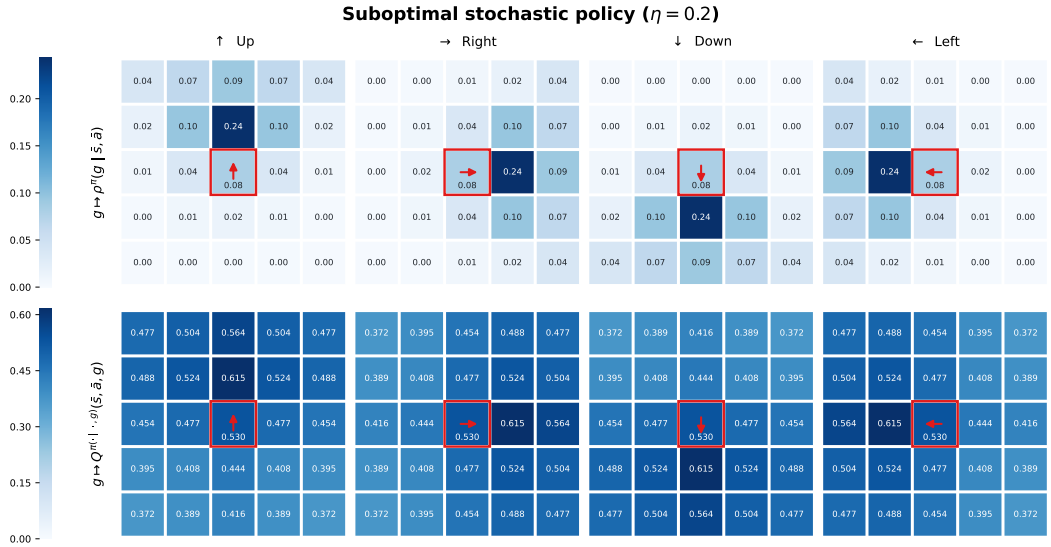

Figure 11:

Goal-conditioned value. The goal-conditioned value uses the reward $r_g(s, a, s') = (1-\gamma)\mathbf{1}\{s' = g\}$.

$$Q^{\pi(\cdot \mid \cdot, g)}(\bar{s}, \bar{a}, g) = \sum_{s'} P(s' \mid \bar{s}, \bar{a}) \left[(1-\gamma)\mathbf{1}\{s' = g\} + \gamma V_g^{\pi}(s')\right],$$

where

$$V_g^{\pi} = r_g^{\pi} + \gamma P_g^{\pi} V_g^{\pi}.$$

With this reward normalization, the value is exactly the diagonal element of the conditional future-goal distribution:

$$Q^{\pi(\cdot \mid \cdot, g)}(\bar{s}, \bar{a}, g) = \rho_g^{\pi}(g \mid \bar{s}, \bar{a}).$$

This identity makes the distinction between Q^π and ρ^π explicit. For each g, $Q^\pi(\bar{s}, \bar{a}, g)$ evaluates the discounted occupancy of g when g is itself the commanded goal. In contrast,

$$\rho^\pi(g \mid \bar{s}, \bar{a}) = \sum_{h \in \mathcal{G}} p^\pi(h \mid \bar{s}, \bar{a})\, \rho^\pi_h(g \mid \bar{s}, \bar{a})$$

aggregates occupancy at g over all possible commanded goals h. Hence ρ^π includes both occupancy generated while pursuing g and occupancy generated while pursuing other goals whose trajectories visit g. The figures display, for each observed directional action,

$$g \mapsto \rho^\pi(g \mid \bar{s}, \bar{a}), \qquad g \mapsto Q^\pi(\bar{s}, \bar{a}, g),$$

for the optimal and suboptimal policy families.

E Additional results

E.1 Results on other environments and comparison to more methods

To position KERL as broadly as possible, we compare it against recent GCRL methods we found that report time-at-goal on JaxGCRL tasks. As running all of them ourselves was not feasible, Tab. 3 reports the scores published in the original papers. Plain values are read directly from tables. Values marked with $\sim$ are estimated from the published learning curves by averaging over the last five epochs, the same protocol we use for KERL. Tasks on which a method was not evaluated are marked as "–".

Table 3: **Time-at-goal comparison with published GCRL methods on JaxGCRL.** Plain values are taken from tables of the original papers, $\sim$ values are estimated from their learning curves and – tasks not reported.

Task	SAC	SAC+HER	Scaled CRL	ANCRe	CRL+AC	Safe-CRL	DCP	SRL	KERL
`ant-u4-maze`	∼0	∼20	286	–	–	–	–	∼800	778
`ant-u5-maze`	∼0	∼80	61	–	–	–	–	∼800	692
`ant-big-maze`	0	84	441	∼500	∼350	371	115	∼790	780
`ant-hardest-maze`	∼10	∼120	387	–	∼200	–	63	∼760	665
`arm-push-easy`	428	423	762	–	–	–	298	∼850	935
`arm-push-hard`	∼260	∼175	410	∼550	–	–	281	∼220	544
`arm-binpick-hard`	∼15	∼25	219	∼250	–	–	–	∼150	534
`humanoid`	358	390	649	∼700	∼10	266	–	∼500	799
`humanoid-U-maze`	∼250	∼0	159	–	∼5	200	4	∼220	516
`humanoid-big-maze`	∼175	∼75	59	–	∼0	–	–	∼50	400
`humanoid-hardest-maze`	–	–	2	–	∼0	–	–	2	340

Sources. We get the scores for SAC, SAC+HER, and scaled CRL from Wang et al. (2026). The remaining baselines are ANCRe (Zhang et al., 2026b), CRL with action chunking (Korniak et al., 2026, CRL+AC), Safe-CRL (Li et al., 2026), direction-conditioned policies (K et al., 2026, DCP) and scaled SRL (Nguimatsia-Tiofack et al., 2026).

Methods not included. Wang et al. (2026) also report TD3+HER, GCSL, and GCBC but we omit them in the table since they are evaluated at a reasonable depth 64 on only three of our tasks. Several related methods are evaluated on JaxGCRL or similar environments but report metrics that are not comparable to time-at-goal. Contrastive PPO (Osman et al., 2026), DISCOVER (Diaz-Bone et al., 2026) and normalizing-flow policies (Ghugare & Eysenbach, 2025) report success rates, Myers et al. (2025) use modified Ant and Humanoid mazes, and CompleteP (Lee et al., 2026) focuses on scaling PPO, mostly with dense rewards.

Learning curves. Fig. 12 shows the full learning curves of KERL, scaled CRL, and scaled SRL on 10 challenging tasks.

Other JaxGCRL environments. For completeness, Fig. 13 reports KERL with depth 16 on six further JaxGCRL tasks outside our main benchmark: `reacher`, `pusher-easy`, `pusher-hard`, `ant`, `ant-u-maze` and `ant-u5-maze`. KERL solves all of them, reaching near-maximal time-at-goal on `reacher` within a few million steps and stable plateaus on the others.

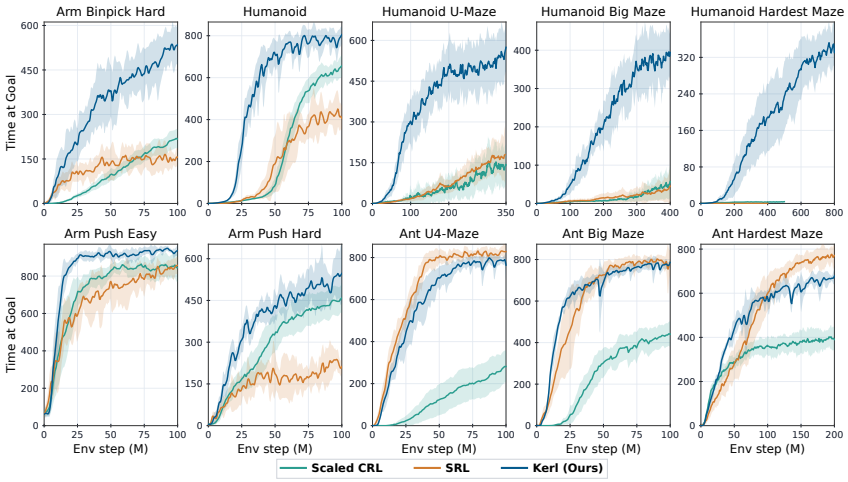

Figure 12: **All comparisons between KERL, CRL and SRL.**

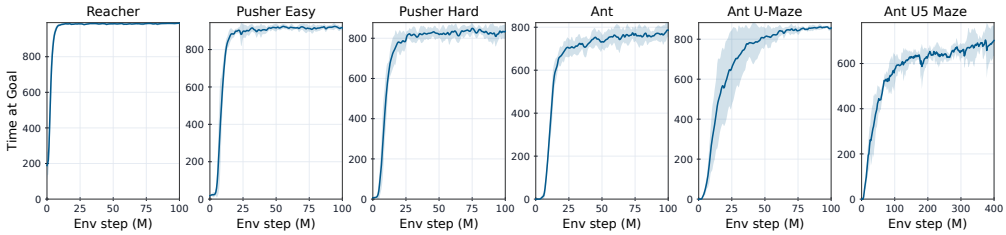

Figure 13: **KERL results on other JaxGCRL environments.**

E.2 ADDITIONAL SCALING EXPERIMENTS

Fig. 14 complements Fig. 5 with the remaining environments, on which KERL also benefits consistently from depth. To separate the effect of depth from parameter count, Fig. 15 compares architectures with the same number of parameters ($\approx$ 2.1M): trading width for depth accelerates learning and improves final performance.

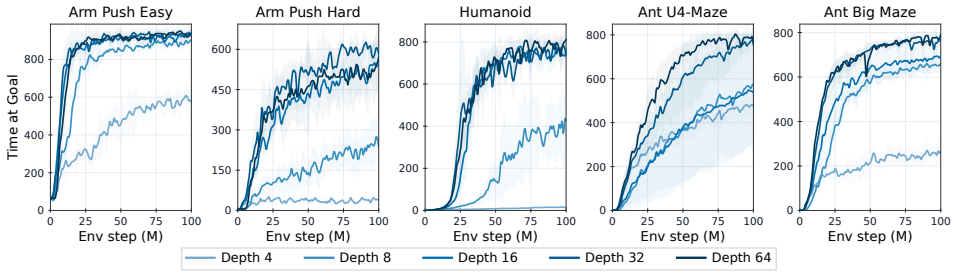

Figure 14: **Additional depth scaling experiments.** Complement to Fig. 5.

E.3 ADDITIONAL ABLATION STUDIES

Fig. 16 completes the ablations of Fig. 8 on `ant-big-maze`, with the same conclusions: the kernel family and the feature map have little influence, learning collapses for $m \in \{8, 16\}$, but here the inner product outperforms the MMD objective. Fig. 17 crosses the number of random frequencies m with network depth.

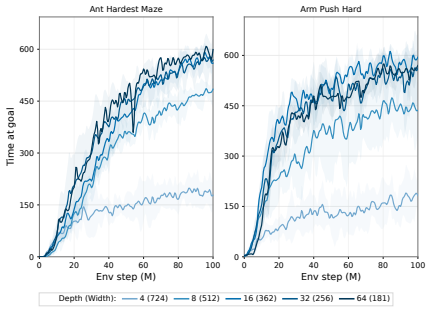

Figure 15: **Deeper architectures consistently learn faster and achieve higher final performance under a fixed parameter budget.** With the same number of parameters ($\approx$ 2.1M), increasing network depth while reducing network width consistently accelerates learning and improves final performance on both environments.

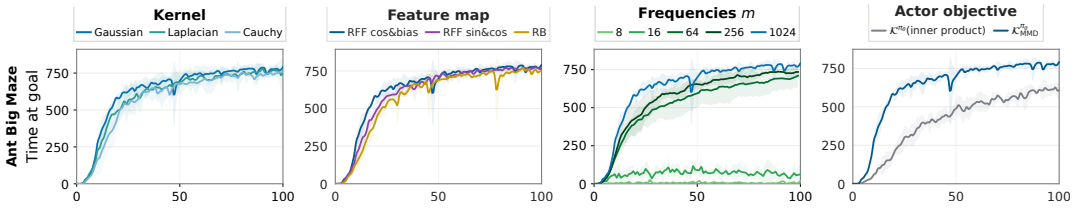

Figure 16: **Ant Big Maze ablations.** Mean over 5 seeds, shaded region one standard deviation.

E.4 Additional analysis of posterior-averaged goal occupancies

Figs. 18, 19 and 20 complement Figs. 1 and 7 with additional environments and views. In each figure, the top row corresponds to 64 layers, the bottom one to 4 layers. Fig. 18 shows `ant-big-maze` and `ant-hardest-maze`, Fig. 19 shows `ant-u4-maze` and `humanoid-big-maze`, finally Fig. 20 shows different state-action pairs for `humanoid-big-maze`. The conclusions carry over: the 64-layer kernel mean embedding network recovers long-horizon occupancies that follow the corridors, while the 4-layer network stays myopic and leaks through walls.

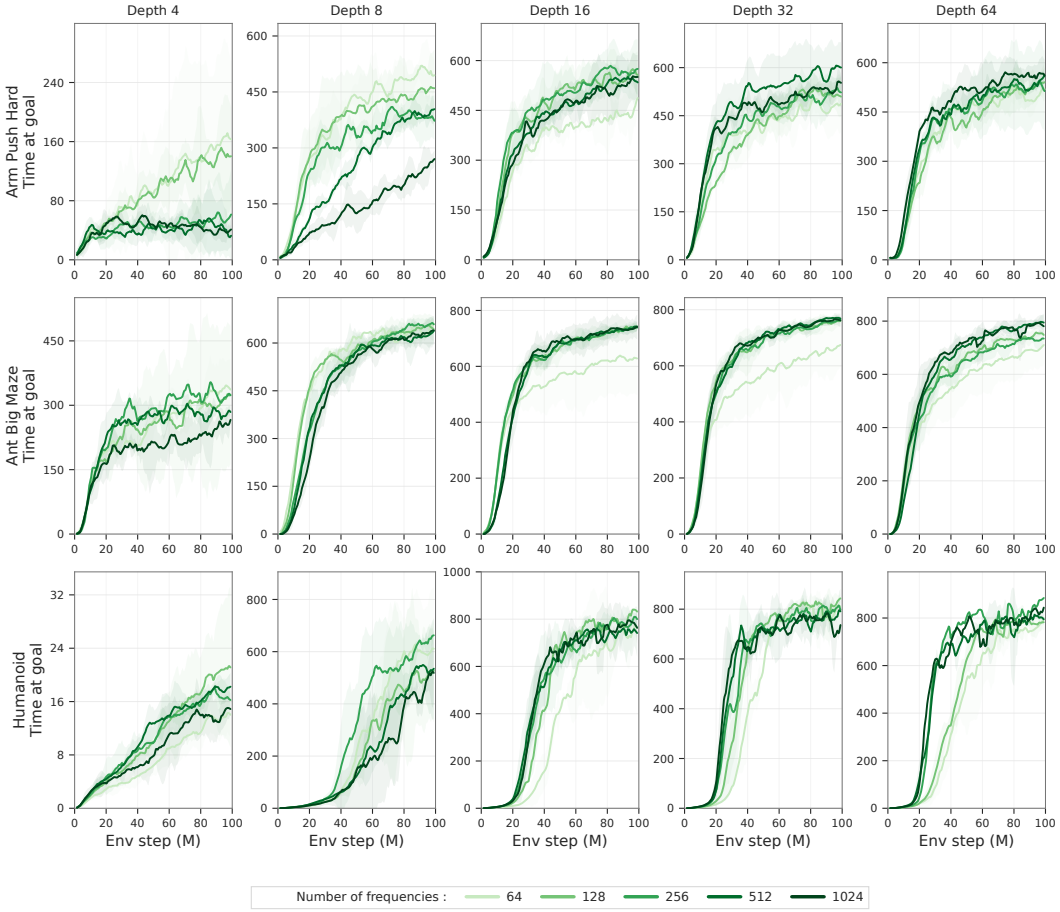

Figure 17: **Network depth vs. number of random frequencies m.** Rows are environments, columns depths. Shallow networks (4–8 layers) do as well or better with few frequencies, whereas from 16 layers on, larger m matches or improves performance and speeds up learning, e.g. on `humanoid`. A richer goal representation only pays off once the kernel mean network is deep enough to fit it.

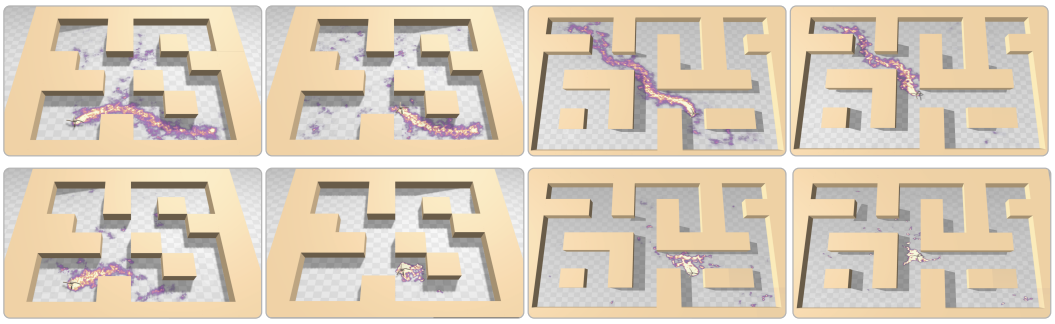

Figure 18: **Reconstructed posterior-averaged goal-occupancy measures** Top (64 layers), bottom (4 layers). First two columns `ant-big-maze`, last ones `ant-hardest-maze`. Complement Fig. 7.

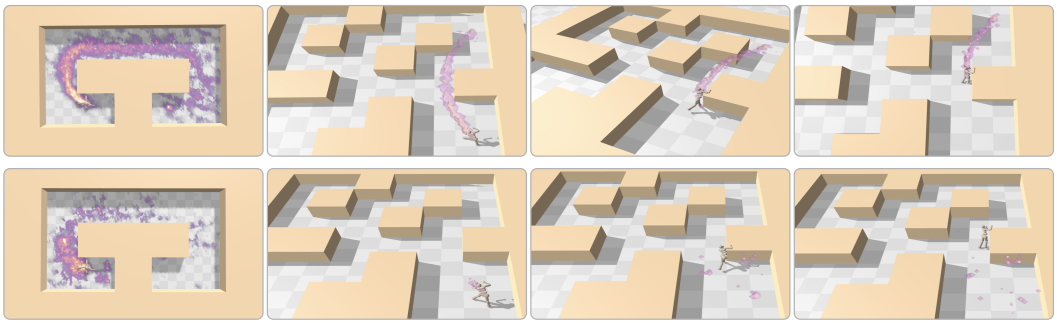

Figure 19: **Reconstructed posterior-averaged goal-occupancy measures** Top (64 layers), bottom (4 layers). First column `ant-u4-maze`, last ones `humanoid-big-maze`. Complement Fig. 7.

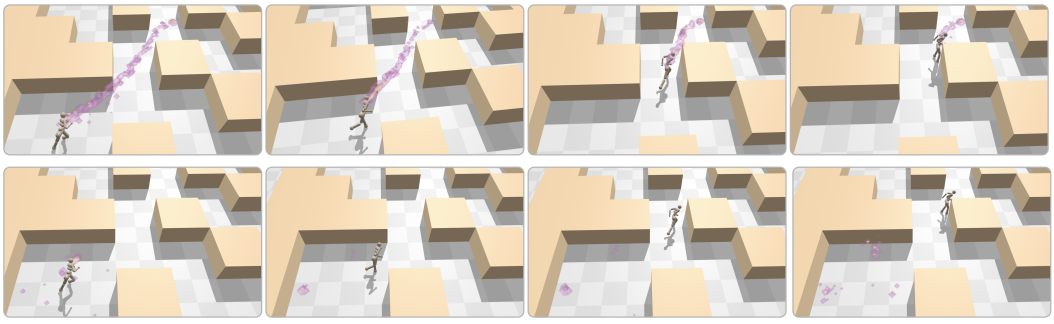

Figure 20: **Reconstructed posterior-averaged goal-occupancy measures** Top (64 layers), bottom (4 layers). Different state-action pair for `humanoid-big-maze`, complement Fig. 7.