GEOMETRIC TIME SCORE ESTIMATORS: BRIDGING SELF-PREDICTIVE REPRESENTATIONS AND GENERATIVE OCCUPANCY MODELS FOR REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In reinforcement learning (RL), temporal reasoning is essential for estimating the density of future outcomes under a policy's occupancy measure. While self-predictive representations have been a dominant paradigm for occupancy density estimation, we find that they yield surprisingly poor estimates. On the other hand, generative occupancy models can produce remarkably accurate samples from the occupancy measure. Yet, they cannot easily recover occupancy densities, limiting their use for control despite their expressivity. In this paper, we propose **Geometric Time Score Estimators (GTSE)**, a framework that bridges the two paradigms by learning occupancy density estimators along probability paths. We analyze how GTSE implicitly induces better occupancy estimators through improved representation learning and show how to directly extract self-predictive representations from this framework. Through experiments across several locomotion and manipulation tasks, we show that GTSE improves performance over state-of-the-art methods by $\mathbf{2}\times$ to $\mathbf{6}\times$ in goal-conditioned RL settings, and that the representations learned using our framework transfer effectively to downstream RL tasks.

1 INTRODUCTION

Predicting the consequences of actions by reasoning about the future is central to reinforcement learning (RL) (Sutton, 1988; Schultz et al., 1997). Self-predictive representation learning methods (Dayan, 1993; Eysenbach et al., 2022; Lawson et al., 2026; Bagatella et al., 2025) utilize temporal reasoning to estimate the probability density of a desired future outcome under the occupancy measure, yet are limited by representational capacity (Xie et al., 2025; Zheng et al., 2026). In contrast, generative occupancy models produce high-fidelity samples from the occupancy measure by learning along probability paths (Farebrother et al., 2025; Zheng et al., 2025). However, such models struggle with occupancy density estimation because they are not explicitly trained to estimate densities (Song et al., 2021; Lu et al., 2022). This leaves their expressivity largely untapped for the temporal reasoning that downstream control requires. These complementary limitations raise the question: *can occupancy density estimation along probability paths bridge the gap between self-predictive representations and generative occupancy models?*

In this paper, we answer this question by leveraging the time score, the time derivative of a time-varying log-density along a probability path (Choi et al., 2022). We propose Geometric Time Score Estimators (GTSE), a framework that uses temporal difference learning to learn the time score of the occupancy measure. GTSE estimates the occupancy density by integrating the time score, enabling explicit temporal reasoning about future outcomes while exploiting the expressivity afforded by learning along probability paths.

We then analyze how learning along probability paths yields improved expressivity. We show that GTSE solves a continuum of future-predictive learning problems. Under a bilinear parameterization, the time-score integral admits an exact decomposition into an infinite sum of inner-product terms, each corresponding to a pair of future-predictive representations. By capturing these representations implicitly, GTSE exhibits improved representational capacity as a result of its higher available

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.

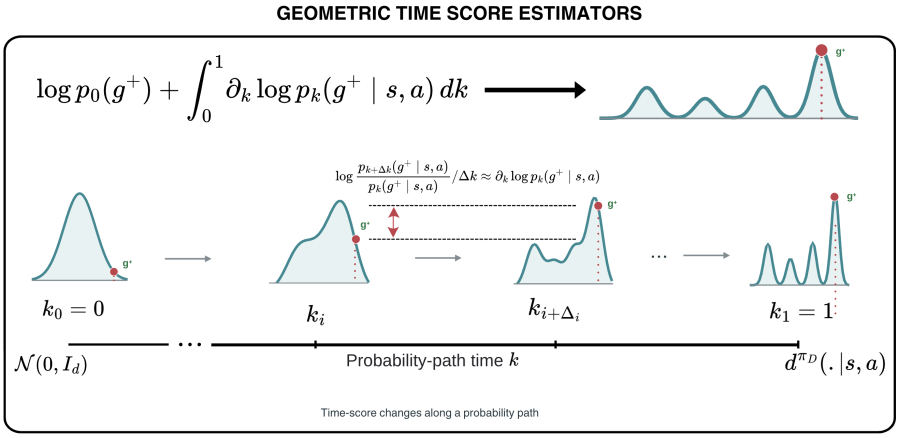


Figure 1: The time score function, $\partial_k \log p_k(g^+ \mid s,a)$, captures the instantaneous rate of change of the log-density of the occupancy measure along a probability path. Integrating it induces accurate occupancy density estimates useful for downstream control applications.

rank. Leveraging this insight, we finally propose a variant of GTSE that yields explicit state-action and goal representations learned along probability paths, bridging the gap between self-predictive representations and generative occupancy models.

Our experiments demonstrate that GTSE estimates occupancy density better than previous state-of-the-art methods and that this translates to significantly better downstream policy performance. On a variety of locomotion and manipulation goal-conditioned RL tasks, GTSE improves performance by $\mathbf{2}\times$ to $\mathbf{6}\times$ over baselines. We also show that representations learned using our method transfer effectively, improving performance on downstream reward-based RL tasks, and we analyze key design choices through ablations.

2 Related Work

The defining challenge of reinforcement learning lies in how to learn which actions lead to good outcomes. While there are many approaches to this problem that focus on reward maximization (Sutton, 1988; Schulman et al., 2015; Feinberg et al., 2018), our paper studies the more general approach of uncovering the structure of the environment using all transitions (Dayan, 1993; Sikchi et al., 2023b;a; Eysenbach et al., 2022; Venugopal et al., 2026). The structure of the environment is typically learned through self-predictive representations. Prior methods do so using density ratio estimation (DRE), factorized successor representations, and latent metric learning.

DRE methods frame RL as a density ratio estimation problem, learning a classifier to predict whether an observation comes from the future state distribution rather than the marginal state distribution (Eysenbach et al., 2022; Zheng et al., 2023; Eysenbach et al., 2020; Liu et al., 2025; Mazoure et al., 2023). Although DRE methods provide a simple approach to goal reaching, they exhibit high variance in high-dimensional settings, and learned density ratio estimates can be unstable with poor dataset coverage (Zheng et al., 2024; Poole et al., 2019). Alternative methods avoid explicit density ratio estimation by learning factorized successor representations via forward-backward (FB) representations (Touati & Ollivier, 2021; Touati et al., 2022), utilizing quasimetric constraints (Myers et al., 2026; 2024), or learning temporal distances in a latent space (Bagatella et al., 2025). These methods typically recover densities via an inner product of representations. While they enjoy improved training stability, they are still limited by representational expressivity (Zheng et al., 2026).

A recent class of methods forgoes learning self-predictive representations and instead employs continuous-time generative modeling over probability paths to estimate velocity fields that transport noise to data (Farebrother et al., 2025; Zheng et al., 2025), which, when integrated, reconstruct samples from the occupancy measure. While these methods are highly expressive, density estimation remains a challenge. Just as continuous normalizing flows can be used to estimate densities by integrating the trace of the Jacobian (Chen et al., 2018; Ghugare & Eysenbach, 2025), flow-matching models can recover occupancy densities in the same way using the Hutchinson trace estimator (Hutchinson, 1989). However, our experiments (Sec. 4, Sec. 7) echo prior work (Song et al., 2021; Lu et al., 2022) in showing that such methods are not particularly effective in practice. GTSE

instead learns how the log-density changes along probability paths, providing a different approach to estimating these densities that works much better in practice.

3 BACKGROUND

Reinforcement Learning. Our setting is a Markov decision process (MDP) (Puterman, 2014) $(\mathcal{S}, \mathcal{A}, r, \mu, p, \gamma)$, where $\mathcal{S}$, $\mathcal{A}$, and γ are the state space, action space, and discount factor, respectively. Here, μ is the distribution over initial states, $p(s' \mid s, a)$ is the transition probability density, and $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. Each policy $\pi(a \mid s) \colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ induces a distribution $d^\pi(s^+ \mid s, a)$ over future states s^+:

$$d^\pi(s^+ \mid s, a) = (1-\gamma) \sum_{t=0}^{\infty} \gamma^t \, \mathbb{P}(s_{t+1} = s^+ \mid s, a, \pi), \tag{1}$$

which we refer to as the occupancy measure. The aim is to learn a policy to maximize the Q-function $Q^\pi(s,a) = r(s,a) + \frac{\gamma}{1-\gamma} \mathbb{E}_{s^+ \sim d^\pi(\cdot \mid s,a)} \left[r(s^+, a \sim \pi(\cdot \mid s^+)) \right]$ (Janner et al., 2020). In this paper, we focus on the offline RL setting with a dataset, $\mathcal{D}$, collected using the behavior policy $\pi_D(a \mid s)$. The dataset occupancy measure has the following recursive form:

$$d^{\pi_D}(s^+ \mid s, a) = (1-\gamma)\, p(s' = s^+ \mid s, a) + \gamma \, \mathbb{E}_{s' \sim p(\cdot \mid s,a),\ a' \sim \pi_D(\cdot \mid s')} \left[d^{\pi_D}(s^+ \mid s', a') \right]. \tag{2}$$

When r is not known, we use the offline goal-conditioned RL (GCRL) (Kaelbling, 1993; Schaul et al., 2015) formulation: the aim is to reach any goal $g \in \mathcal{S}$ from another state $s \in \mathcal{S}$ in the minimum number of steps. The optimal goal-conditioned policy $\pi^*(a \mid s, g)$ maximizes the goal-conditioned Q-function: $Q^\pi(s, a, g) = \mathbb{E}_{\tau \sim p(\tau \mid s_0 = s,\, a_0 = a,\, g)} \left[\sum_{t=0}^{\infty} \gamma^t r_g(s_t, a_t) \right]$, where $p(\tau \mid s_0 = s, a_0 = a, g) = p(s_1 \mid s, a) \prod_{t=1}^{\infty} \pi(a_t \mid s_t, g)\, p(s_{t+1} \mid s_t, a_t)$ and $r_g(s_t, a_t) := (1-\gamma)\, p(s_{t+1} = g \mid s_t, a_t)$. Note that under r_g, $d^\pi(s^+ = g \mid s, a) = Q^\pi(s, a, g)$ (Eysenbach et al., 2022).

Time Score Matching. Let p_0 and p_1 correspond to two arbitrary probability distributions. Time Score Matching (Choi et al., 2022; Yu et al., 2025) enables density ratio estimation between p_0 and p_1 by learning the instantaneous rate of change of log-density, called the time score, along a sequence of intermediate distributions $\{p_k(\cdot)\}_{k \in [0,1]}$ corresponding to a probability path of time-dependent densities that smoothly interpolate between p_0 and p_1. It can be understood by noting that for any $x_1 \sim p_1(\cdot)$, $\log \frac{p_{k_{i+1}}(x_1)}{p_{k_i}(x_1)}$ can be summed over i to recover $\log \frac{p_1(x_1)}{p_0(x_1)}$ for $0 = k_0 < k_1 < \cdots < k_M = 1$. In the limiting case, as Δk becomes infinitesimal, the instantaneous rate of change of the log-density is $\log \frac{p_{k+\Delta k}(x_1)}{p_k(x_1)} / \Delta k \approx \partial_k \log p_k(x_1)$, the time score. Integrating it over k recovers $\log \frac{p_1(x_1)}{p_0(x_1)}$ (Choi et al., 2022). The time score is generally estimated using a neural network s_θ. While the target $\partial_k \log p_k(x_k)$ is intractable, conditioning on the endpoint $x_1 \sim p_1(\cdot)$ enables computing the conditional time score in closed form, giving the training objective (Yu et al., 2025):

$$\mathcal{L}(\theta) = \mathbb{E}_{k, x_1 \sim p_1, x_k \sim p_k(\cdot \mid x_1)} \left[\lambda(k) \, \| \partial_k \log p_k(x_k \mid x_1) - s_\theta(x_k, k) \|_2^2 \right],$$

where $\lambda(k)$ is a positive weighting function. The time k is defined with respect to the probability path and is different from the MDP timestep t. In this paper, we use the variance-preserving (VP) probability path (Song et al., 2020; Lipman et al., 2024; Choi et al., 2022; Yu et al., 2025), along which samples are drawn from p_k as follows: $x_k = \sqrt{1 - \alpha_k^2}\, x_0 + \alpha_k x_1$, where $x_0 \sim p_0$, $x_1 \sim p_1$, $k \sim \mathcal{U}[0,1]$ and $\alpha_k \in [0,1]$ is monotonically increasing in k.

In Sec. 4, we examine how faithfully state-of-the-art methods estimate occupancy density and evaluate how their limitations affect downstream control. In Sec. 5, we develop GTSE, our framework to address these limitations via occupancy modeling along probability paths. In Sec. 6, we connect GTSE to self-predictive representation learning to show how it leads to occupancy density estimators with better representational capacity. We also propose a variant of GTSE that learns explicit state-action and goal representations along probability paths. Sec. 7 evaluates GTSE and its variants in downstream control applications.

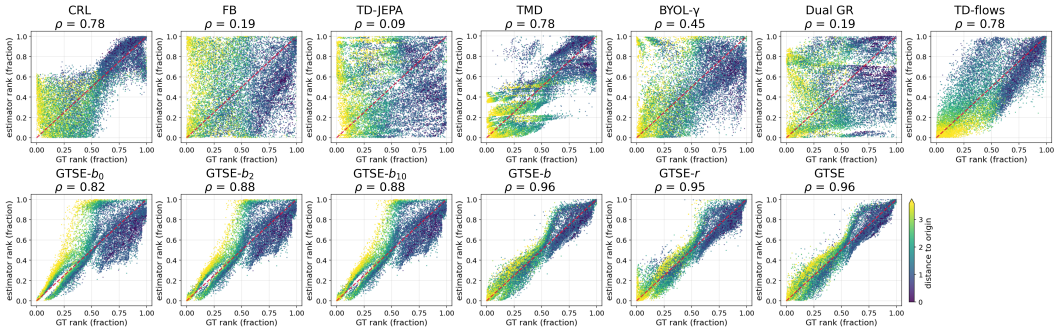


Figure 2: Normalized rank of estimated occupancy density vs. the normalized rank of ground-truth $\log d^{\pi_D}$ over 50 goals and 300 state-action pairs per goal on a 5-step toy MDP (colored by distance of state from origin). GTSE variants are introduced in Secs. 5–6.

4 How well do existing methods estimate occupancy density?

Self-predictive representation learning is motivated by the central role of representations in effective downstream control through RL. While this motivation is well-founded, state-of-the-art methods typically rely on learning low-rank approximations of the occupancy measure. Generative flow-matching occupancy models (Farebrother et al., 2025) instead produce full reconstructions of future states. Although a recipe exists to recover the occupancy density from a learned velocity field (Grathwohl et al., 2018), its effectiveness is unclear. In this section, we examine the limitations of several state-of-the-art methods for estimating occupancy density and study whether these limitations translate to lower effectiveness in downstream control applications.

To this end, we first design a 5-step toy MDP with $\mathcal{S} = \mathcal{A} = \mathbb{R}^2$ and s_0 chosen uniformly from $\{(0,0),(0,1),(1,0),(-1,0),(0,-1)\}$. At $t = 2$, a branch variable $b \sim \mathcal{U}\{-1,+1\}$ is sampled, and the trajectory commits to this branch until $t = 5$, leading to a multimodal occupancy measure. The dynamics are defined as $s_{t+1} = s_t + a_t + \epsilon_t$, where $\epsilon_t \sim \mathcal{N}(0, \sigma^2 I)$. Actions are chosen as $a_t = \eta_t$ for $t < 2$ and $a_t = b\mu + \eta_t$ for $t \geq 2$, where $\eta_t \sim \mathcal{N}(0, \sigma_a^2 I)$ and μ is a fixed offset.

On a dataset of 20k trajectories, we evaluate the following methods: CRL (Eysenbach et al., 2022) uses contrastive learning; TMD (Myers et al., 2026) uses quasimetric RL; FB (Touati & Ollivier, 2021), TD-JEPA (Bagatella et al., 2025) and BYOL-γ (Lawson et al., 2026) use TD learning to learn self-predictive representations. Dual Goal Representations (Dual GR) (Park et al., 2026) learns representations using an objective based on Implicit Q-Learning (Kostrikov et al., 2021). TD-flows (Farebrother et al., 2025) learns an occupancy model using conditional flow matching; we use the Hutchinson trace estimator (Hutchinson, 1989; Grathwohl et al., 2018) to recover occupancy densities with the trained velocity field. See Appendix D.2 for details.

Fig. 2 plots the normalized rank of the predictions of each model against the normalized rank of the ground-truth $\log d^{\pi_D}$ over 50 goal states and 300 state-action pairs per goal. Points are colored by the distance of the state from the origin. Fig. 2 also reports the overall Spearman correlation coefficient (ρ) of predictions against the ground truth. The first row of Fig. 2 reveals, somewhat surprisingly, that even in a simple MDP, most baselines estimate $\log d^{\pi_D}$ poorly. While CRL and TMD ($\rho = 0.78$) capture $\log d^{\pi_D}$ near the origin (where most of the probability mass is concentrated) relatively better, they poorly estimate $\log d^{\pi_D}$ at states farther away. TD-flows ($\rho = 0.78$) performs similarly to CRL and TMD, but it still exhibits estimation errors as it is not explicitly trained to estimate densities.

To understand whether these estimation errors limit effectiveness in downstream control, we train the methods on a high-coverage dataset collected in the **pointmaze-large** environment of OGBench (Park et al., 2024). In **pointmaze-large**, both state and action spaces are $\mathbb{R}^2$ and correspond to (x, y) positions and (x, y) position increments, respectively. The aim is to navigate a point-mass agent to a specified goal state in the maze. We call this dataset **pointmaze-large-diverse**. See Appendix D.1 for details on dataset collection and ground-truth occupancy density estimation.

We evaluate the action-ranking capability of each method on this dataset by measuring ρ against the ground truth over 500 randomly drawn state-goal pairs and 16 actions at each pair. For all methods, actions are sampled using a learned flow-matching policy (Appendix C). We study how estimation quality varies with goal distance by separately measuring ρ for nearby goals likely to have a higher $d^{\pi_D}(g \mid s, a)$ and far-away goals likely to have a lower $d^{\pi_D}(g \mid s, a)$. To this end, we split goals

based on the median distance computed over the 500 state-goal pairs. We then measure the success rate over 5 test-time goals, where each method uses best-of-M action selection over the predicted occupancy density during inference (M=16). Table 1 summarizes the results.

Consistent with Fig. 2, Table 1 shows that the baselines (left 7 columns) estimate the occupancy density poorly and that estimation quality generally degrades with distance to the goal. CRL and TMD are relatively better for nearby goals ($\rho = 0.56$), but ρ falls to at most 0.30 (TMD) for farther goals. While TD-flows is comparable to CRL and TMD on the toy MDP, we find that it diverges on pointmaze due to estimation errors in TD learning of flows. The last row of Table 1 demonstrates how estimation errors negatively affect the success rate, with even the best-performing baseline (CRL) reaching only a 3% success rate. Taken together, Fig. 2 and Table 1 highlight how existing methods can be ineffective even in simple control tasks because they struggle to model the occupancy measure accurately. We now introduce GTSE to address this limitation.

5 Geometric Time Score Estimators

We introduce Geometric Time Score Estimators (GTSE) in Sec. 5.1. In Sec. 5.2, we provide a goal-conditioned RL algorithm based on GTSE for use in downstream control applications. We then demonstrate its ability to effectively estimate occupancy density in Sec. 5.3.

5.1 Learning GTSE through occupancy modeling along probability paths

We aim to estimate the occupancy measure density, $d^{\pi_D}(s^+ \mid s, a)$, for any s^+. Estimating $d^{\pi_D}(s^+ \mid s, a)$ is difficult because it is often a complex multimodal distribution over future states. Fortunately, recent successes of generative models in RL (Schramm & Boularias, 2024; Zheng et al., 2025) provide a valuable insight: rather than estimating the occupancy density directly, we can decompose it into simpler subproblems that, when solved together, recover it.

Leveraging this insight, we define a simple base distribution p_0 of known density, independent of (s, a): $p_0(\cdot \mid s, a) = p_0(\cdot) = \mathcal{N}(0, I)$. We then construct a sequence of intermediate distributions $\{p_k(\cdot \mid s, a)\}_{k\in[0,1]}$ along the VP probability path between $p_0(\cdot)$ and $p_1(\cdot \mid s, a) \equiv d^{\pi_D}(\cdot \mid s, a)$. With $k \sim \mathcal{U}[0, 1]$, $s_0^+ \sim p_0$, and $s_1^+ = s^+ \sim p_1$, we can draw samples from the probability path:

$$s_k^+ = \alpha_k s^+ + \sqrt{1-\alpha_k^2}\, s_0^+, \qquad p_k(s_k^+ \mid s, a) = \int \mathcal{N}\big(s_k^+; \alpha_k s^+, (1-\alpha_k^2)I\big)\, d^{\pi_D}(s^+ \mid s, a)\, ds^+.$$

This lets us define the time score corresponding to the occupancy measure as $\partial_k \log p_k(s^+ \mid s, a)$. The time score is integrated to recover the log occupancy density: $\log d^{\pi_D}(s^+ \mid s, a) = \int_0^1 \partial_k \log p_k(s^+ \mid s, a)\, dk + \log p_0(s^+)$. We train an estimator of the time score by evaluating it at noised samples s_k^+. Conditioning p_k on $s_1^+ = s^+$ following Yu et al. (2025) reduces it to $p_k(s_k^+ \mid s^+, s, a) = p_k(s_k^+ \mid s^+) = \mathcal{N}(s_k^+; \alpha_k s^+, (1-\alpha_k^2)\, I)$. The conditional time score now admits the closed form:

$$\partial_k \log p_k(s_k^+ \mid s^+) \;=\; \dim(\mathcal{S})\, \frac{\alpha_k \alpha_k'}{1-\alpha_k^2} \;-\; \frac{\alpha_k \alpha_k'}{1-\alpha_k^2}\, \|\epsilon\|^2 \;+\; \frac{1}{\sqrt{1-\alpha_k^2}}\, \epsilon^\top \alpha_k' s^+, \tag{3}$$

where $\alpha_k' = d\alpha_k/dk$, $\epsilon = \frac{s_k^+ - \alpha_k s^+}{\sqrt{1-\alpha_k^2}}$ and $\dim(\mathcal{S})$ is the dimensionality of s^+. See Appendix A.1 for the derivation. With $\lambda(k) \propto 1 - \alpha_k^2$, we can train a neural network s_θ to minimize:

$$\mathcal{L}(\theta) = \mathbb{E}_{s^+ \sim d^{\pi_D}(\cdot \mid s,a)}\, \mathbb{E}_{k,\, s_k^+ \mid s^+} \Big[\lambda(k) \big(\partial_k \log p_k(s_k^+ \mid s^+) - s_\theta(s_k^+, k, s, a) \big)^2 \Big], \quad \forall (s, a) \in \mathcal{D}. \tag{4}$$

Table 1: Spearman rank correlation ρ with ground-truth $\log d^{\pi_D}$ for nearby and far-away goals along with overall average success rate (%) across the 5 test-time goals on **pointmaze-large-diverse** over 4 seeds. We report the 95% bootstrapped C.I. half-width after the $\pm$ sign. GTSE variants are introduced in Secs. 5–6.

	CRL	**FB**	**TD-JEPA**	**BYOL-γ**	**Dual GR**	**TMD**	**TD-flows**	**GTSE-b_0**	**GTSE-b_2**	**GTSE-b_{10}**	**GTSE-b**	**GTSE-r**	**GTSE**
ρ (near)	0.56	0.11	0.25	0.33	−0.07	0.56	0.00	0.56	0.60	0.60	0.60	0.60	**0.63**
ρ (far)	0.14	0.00	0.08	0.16	−0.09	0.30	0.05	0.50	0.51	0.52	0.53	0.52	**0.55**
Success rate	3 ± 6	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	12 ± 5	21 ± 10	19 ± 9	25 ± 4	25 ± 2	$\mathbf{38 \pm 4}$

An estimator learned using Eq. 4, however, is unable to capture temporal linkages between states that are reachable only by stitching together segments of different trajectories, which limits the accuracy of occupancy density estimation (Ghugare et al., 2024). To address this, we exploit the recursive nature of the occupancy measure to propagate time-score predictions of future states using temporal difference learning. We thus arrive at the geometric time score matching objective:

Geometric Time Score Matching objective

$$\mathcal{L}_{\text{GTSE}}(\theta) = (1-\gamma)\,\mathcal{L}_{\text{next}}(\theta) + \gamma\,\mathcal{L}_{\text{future}}(\theta), \qquad \forall\,(s,a,s',a') \in \mathcal{D}, \tag{5a}$$

$$\mathcal{L}_{\text{next}}(\theta) = \mathbb{E}_{k,\,s'_k|s'}\Big[\lambda(k)\big(\partial_k \log p_k(s'_k \mid s') - s_\theta(s'_k,k,s,a)\big)^2\Big], \tag{5b}$$

$$\mathcal{L}_{\text{future}}(\theta) = \mathbb{E}_{s^+\sim d^{\pi_D}(\cdot|s',a')}\,\mathbb{E}_{k,\,s^+_k|s^+}\Big[\lambda(k)\big(\text{sg}\left[s_{\bar\theta}(s^+_k,k,s',a')\right] - s_\theta(s^+_k,k,s,a)\big)^2\Big]. \tag{5c}$$

While $\mathcal{L}_{\text{next}}(\theta)$ uses the observed next state s' to construct a time-score target, $\mathcal{L}_{\text{future}}(\theta)$ regresses $s_\theta(s^+_k,k,s,a)$ towards a bootstrapped target $s_{\bar\theta}(s^+_k,k,s',a')$, propagating time scores from future states. Here, $s_{\bar\theta}$ is a copy of s_θ with time-delayed parameters, and sg denotes the stop-gradient operator. See Appendix A.3 for the derivation of Eq. 5. During inference, the log-density of any s^+ under d^{π_D} can be recovered by approximating the time-score integral:

$$\log \hat{d}^{\pi_D}(s^+ \mid s,a) = \frac{1}{N}\sum_{i=0}^{N-1} s_\theta(s^+, {}^i\!/_N, s, a) + \log p_0(s^+). \tag{6}$$

In practice, we truncate the integral at $K_{\max} = 1-\varepsilon$ for stability (Sec. 7).

5.2 Practical algorithm

We provide a complete algorithm for goal-conditioned RL in Alg. 1 (Appendix B). Alg. 1 trains GTSE together with $\pi_\omega(\cdot \mid s,g)$, a goal-conditioned flow-matching behavioral cloning (BC) policy (Park et al., 2025b). During evaluation, for a task defined by goal g, we use best-of-M action selection to sample $\hat{a}_1,\ldots,\hat{a}_M \sim \pi_\omega(\cdot \mid s,g)$ and choose $\text{argmax}_{\hat{a}_i} \log \hat{d}^{\pi_D}(g \mid s,\hat{a}_i)$ (Chen et al., 2022; Park et al., 2025a; Dong et al., 2025). See Appendix C for implementation details.

5.3 Is GTSE a better estimator of the occupancy measure?

Fig. 2 and Table 1 evaluate GTSE's ability to estimate occupancy density. In the 5-step toy MDP, GTSE ($\rho = 0.96$) is substantially better than the best baselines, TD-flows, CRL and TMD ($\rho = 0.78$). We see a similar trend on **pointmaze-large-diverse**, where GTSE captures the occupancy measure best for nearby ($\rho = 0.63$) as well as far-away goals ($\rho = 0.55$) and achieves the highest success rate ($38 \pm 4\%$), over $12.5\times$ that of CRL.

6 Bridging self-predictive representations and generative occupancy modeling

In this section, we first show how GTSE can be interpreted as multiscale future prediction along probability paths. This connection allows us to understand how GTSE's improved representational capacity arises. We then use this insight to propose a variant, GTSE-r, which explicitly learns state-action and goal representations along probability paths.

6.1 Geometric Time Score Matching as multiscale future prediction

The GTSE objective (Eq. 5) provides dense targets across varying noise levels along a probability path for learning the time score of the occupancy measure. To understand how this leads to an effective occupancy density estimator with better representational capacity, we first show that we can interpret this objective as a future-prediction problem over a continuum of noise levels:

Lemma 1 (Informal). *The estimator that minimizes Eq. 4 is given by:*

$$s^*_\theta(s^+_k, k, s, a) = \mathbb{E}_{s^+\sim p_k(\cdot|s^+_k,s,a)}\left[\partial_k \log p_k(s^+_k \mid s^+)\right], \qquad \epsilon = \frac{s^+_k - \alpha_k s^+}{\sqrt{1-\alpha_k^2}},$$
$$= \mathbb{E}_{s^+\sim p_k(\cdot|s^+_k,s,a)}\left[\frac{\alpha_k \alpha'_k}{1-\alpha_k^2}\left(\dim(\mathcal{S}) - \|\epsilon\|^2\right) + \frac{\alpha'_k}{\sqrt{1-\alpha_k^2}}\,\epsilon^\top s^+\right]. \tag{7}$$

See Appendix A.2 for the proof. Intuitively, Lemma 1 means that at each noise level k, GTSE must average time-score targets over plausible future states from the posterior $p_k(\cdot \mid s^+_k, s, a)$. Learning this estimator with Eq. 5 requires encoding future states that are plausible from (s, a) given the noised sample s^+_k, using the model's own predictions at (s', a') through $\mathcal{L}_{\text{future}}(\theta)$.

6.2 Connecting multiscale future prediction to representational capacity

Building on Lemma 1, we show that the occupancy density estimated by integrating the time score is equivalent to a sum over inner products of implicit self-predictive representations, under a bilinear parameterization of the time score:

Proposition 1 (Legendre decomposition of the time score). *Assume the time score factorizes as*

$$\partial_k \log p_k(g \mid s, a) \;=\; \phi(s, a, k)^\top \psi(g, k),$$

and that both ϕ and ψ are δ-dimensional and square-integrable in $k \in [0, 1]$. Let $\{e_l(k)\}_{l\geq 0}$ denote the shifted orthonormal Legendre basis on $[0, 1]$. Then, for any goal $g \in \mathcal{D}$,

$$\log d^{\pi_D}(g \mid s, a) \;=\; \sum_{l=0}^{\infty} \hat{\phi}_l(s, a)^\top \hat{\psi}_l(g) \;+\; \log p_0(g), \tag{8}$$

where $\hat{\phi}_l(s, a)$ and $\hat{\psi}_l(g)$ are the l-th Legendre coefficient vectors of $\phi(s, a, \cdot)$ and $\psi(g, \cdot)$ in k.

See Appendix A.4 for the proof. The time score integral can thus be decomposed into an infinite sum with each term having a rank-δ contribution. By truncating this sum at order L, we recover a log occupancy density ratio estimate of rank at most $(L + 1)\delta$, explaining GTSE's representational capacity as a consequence of both its training procedure and its inference procedure. In contrast, methods evaluated in Sec. 4 learn rank-δ approximations, structurally similar to the special case of $L = 0$. For instance, Touati & Ollivier (2021) learn $F(s, a)^\top B(g) \approx d^{\pi_D}(g \mid s, a)/p_D(g)$ while $L = 0$ gives $\log \frac{d^{\pi_D}(g|s,a)}{p_0(g)} \approx \hat{\phi}_0(s, a)^\top \hat{\psi}_0(g)$, where p_D is the marginal state distribution of $\mathcal{D}$.

To empirically verify whether the Legendre decomposition mechanism holds, we train the bilinear variant of GTSE on the toy MDP and **pointmaze-large-diverse** introduced in Sec. 4. We call this GTSE-b. Once trained, we extract Legendre-truncated versions from GTSE-b **post hoc**, during inference, using the Gauss-Legendre quadrature method (details in Appendix C). We call the Legendre-truncated versions GTSE-b_L, where L is the truncation order. Fig. 2 and Table 1 show that as we go from GTSE-b_0 through GTSE-b_{10} to GTSE-b, ρ increases in toy MDP experiments (0.82 to 0.96). Similarly, on **pointmaze-large-diverse**, ρ and success rates generally increase with L (12 ± 5 on GTSE-b_0 to 25 ± 4 on GTSE-b). GTSE performs slightly better than GTSE-b overall, owing to its representational capacity not being limited by a bilinear architecture. These experiments provide further empirical evidence to demonstrate GTSE's improved representational capacity.

6.3 GTSE-r: Explicit self-predictive representations

While the methods in Sec. 4 learn state-action and goal representations, neither $\hat{\phi}_l(s, a)$ nor $\hat{\psi}_l(g)$ from Proposition 1 is explicitly instantiated during training or inference. The strong performance of GTSE-b_0, a post-hoc truncation of GTSE-b, suggests that explicitly training rank-δ approximations of the occupancy measure using geometric time score matching could lead to effective density estimators. In fact, by parameterizing the time score as $s_\theta(g, k, s, a) = \phi_\theta(s, a)^\top \psi_\theta(g, k)$, we can interpret the time score integral during inference as an inner product of a state-action representation shared across noise levels and a goal representation:

$$\frac{1}{N}\sum_{i=0}^{N-1} s_\theta(g, {}^i\!/_N, s, a) = \phi_\theta(s, a)^\top \frac{1}{N}\sum_{i=0}^{N-1} \psi_\theta(g, {}^i\!/_N) = \phi_\theta(s, a)^\top \bar{\psi}_\theta(g). \tag{9}$$

Table 2: GCRL tasks: Overall average success rate (%) across the 5 test-time goals over 8 seeds per task per algorithm. We report the 95% bootstrapped C.I. half-width after the ± sign.

Dataset	GCBC	CRL	FB	TD-JEPA	BYOL-γ	Dual GR	TMD	SMORE	TD-flows	NF	GTSE-r	GTSE
scene-play	20 ± 4	25 ± 5	6 ± 2	31 ± 4	17 ± 2	19 ± 15	7 ± 2	13 ± 4	35 ± 4	9 ± 3	60 ± 5	$\mathbf{69 \pm 6}$
scene-noisy	10 ± 1	7 ± 3	2 ± 1	24 ± 3	9 ± 3	12 ± 7	7 ± 1	3 ± 2	19 ± 4	5 ± 2	37 ± 4	$\mathbf{55 \pm 6}$
puzzle-4x4-play	1 ± 1	2 ± 1	22 ± 8	$\mathbf{36 \pm 3}$	2 ± 1	17 ± 6	1 ± 1	0 ± 1	4 ± 2	1 ± 1	26 ± 6	$\mathbf{40 \pm 4}$
puzzle-4x4-noisy	1 ± 0	0 ± 0	6 ± 2	9 ± 3	0 ± 0	2 ± 3	0 ± 0	0 ± 0	1 ± 0	1 ± 0	9 ± 2	$\mathbf{22 \pm 2}$
cube-triple-play	5 ± 2	2 ± 1	1 ± 0	1 ± 1	0 ± 1	13 ± 2	1 ± 1	2 ± 3	16 ± 3	0 ± 0	7 ± 1	$\mathbf{18 \pm 1}$
cube-triple-noisy	2 ± 1	1 ± 1	0 ± 1	0 ± 0	0 ± 1	4 ± 2	0 ± 0	1 ± 1	$\mathbf{8 \pm 2}$	1 ± 1	4 ± 1	$\mathbf{8 \pm 1}$
antmaze-large-navigate	34 ± 4	73 ± 6	15 ± 8	29 ± 6	32 ± 5	43 ± 6	29 ± 4	35 ± 3	30 ± 9	51 ± 16	$\mathbf{78 \pm 4}$	53 ± 4
Mean	10.4	15.7	7.4	18.6	8.6	15.7	6.4	7.7	16.1	9.7	31.6	**37.9**

This parameterization, which we call GTSE-r, allows us to interpret GTSE as self-predictive representation learning over multiple noise levels along probability paths. GTSE-r is a strong occupancy density estimator on the toy MDP ($\rho = 0.95$) as well as on **pointmaze-large-diverse** (success rate of $25 \pm 2\%$).

7 Performance in downstream control tasks

Secs. 5 and 6 demonstrate that GTSE and its variants are better estimators of occupancy density through experiments on simple environments. We now extend our empirical analysis by evaluating GTSE and GTSE-r on downstream control applications to answer the following research questions:

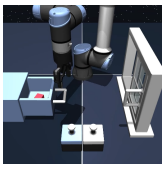
1) Do GTSE and GTSE-r improve downstream policy performance on challenging long-horizon goal-conditioned RL tasks?

Experimental setup. We conduct experiments in offline multi-task GCRL settings from OGBench (Park et al., 2024), broadly categorized into maze navigation, cube manipulation, puzzle manipulation, and scene tasks. We pick a representative subset of datasets that contains challenging long-horizon tasks including noisy datasets with highly suboptimal data, and perform multi-goal evaluation. See Appendix D.1 for a detailed discussion on tasks and datasets. In addition to the baselines introduced in Sec. 4, we evaluate against goal-conditioned behavioral cloning (GCBC) (Park et al., 2025b), Dual RL (SMORE) (Sikchi et al., 2023a), and Normalizing Flows (NF) (Ghugare & Eysenbach, 2025). We use best-of-M action selection with a flow-matching policy in our experiments because we aim to isolate the effectiveness of the estimator. We build on the codebase of OGBench. All algorithms use hindsight relabeling (Kaelbling, 1993; Andrychowicz et al., 2017). Appendix D contains additional details on tasks, baselines, and hyperparameters used.

Results. We report average success rates on 5 test-time goals per task over 8 seeds in Table 2. GTSE achieves an average success rate of 37.9%, $\mathbf{6}\times$ that of the weakest baseline TMD (6.4%) and $\mathbf{2}\times$ that of TD-JEPA, the strongest baseline (18.6%). GTSE-r (31.6%) performs substantially better than the baselines on average, outperformed only by GTSE. GTSE consistently achieves the best performance in manipulation tasks, even in very hard ones such as **cube-triple** ($18 \pm 1\%$). While GTSE also performs strongly on **antmaze-large-navigate**, GTSE-r performs better. We find that this could be due to goal density being determined almost exclusively by the first two state dimensions that correspond to the agent's (x, y) position, making the occupancy measure inherently low-dimensional (Appendix E.1). The bilinear GTSE-r perhaps captures this better. Overall, we see that occupancy density estimators learned along probability paths exhibit significantly better performance for downstream control tasks.

2) Can the representations learned by GTSE-r effectively transfer across reward-based RL tasks?

Table 3: Reward-based RL tasks with frozen representation encoders on **scene-play**. We report success rate (%) with the 95% bootstrapped C.I. half-width after the ± sign, averaged over 4 seeds.



Task	CRL	FB	TD-JEPA	BYOL-γ	Dual GR	TMD	Vanilla FQL	GTSE-r
task 1	67 ± 6	58 ± 33	98 ± 5	51 ± 27	37 ± 7	1 ± 2	$\mathbf{100 \pm 0}$	$\mathbf{100 \pm 1}$
task 2	14 ± 12	17 ± 19	32 ± 22	8 ± 8	4 ± 3	0 ± 0	$\mathbf{83 \pm 23}$	$\mathbf{81 \pm 12}$
task 3	27 ± 13	1 ± 2	85 ± 11	10 ± 6	11 ± 9	0 ± 0	$\mathbf{99 \pm 3}$	94 ± 5
task 4	10 ± 5	0 ± 0	$\mathbf{18 \pm 14}$	10 ± 4	7 ± 4	0 ± 0	9 ± 7	$\mathbf{36 \pm 18}$
task 5	2 ± 2	0 ± 0	0 ± 0	0 ± 0	4 ± 7	0 ± 0	0 ± 0	0 ± 0
Mean	24.0	15.2	46.6	15.8	12.6	0.2	58.2	**62.2**

Experimental setup. We have demonstrated that GTSE-r is significantly better than the baselines for downstream control, suggesting that its representations encode control-relevant future-predictive information. We now evaluate whether the representations $\phi(s, a)$ learned by GTSE-r (Eq. 9) transfer to reward-based offline RL tasks. Following Park et al. (2025b), we choose the reward-based datasets of **scene-play** (Appendix D.1), because this domain involves long-horizon planning to sequentially manipulate various objects to target test-time configurations not present in the training data. Effective transfer therefore requires representations to encode future-predictive features that enable combinatorial generalization. For each method, we use a ***frozen representation encoder*** to encode either (s, a) or s (based on whether it learns state-action or state-only representations), in which case the action is concatenated. This is then fed to the critic network in Flow Q-Learning (FQL) (Park et al., 2025b). We evaluate over 5 tasks, each with a distinct reward function.

Results. We report average success rates on each task over 4 seeds in Table 3 after 1M update steps. GTSE-r achieves the highest average success rate overall (62.2%), exceeding the strongest baselines TD-JEPA (46.6%), and vanilla FQL (58.2%). While GTSE-r matches vanilla FQL on tasks 1–3, where the performance is near saturation, it substantially improves performance on task 4 by around $4\times$, despite using a frozen representation encoder, whereas vanilla FQL trains end-to-end. All methods perform poorly on task 5. Overall, these results suggest that self-predictive representations learned along probability paths retain control-relevant information that enables efficient transfer to downstream RL tasks.

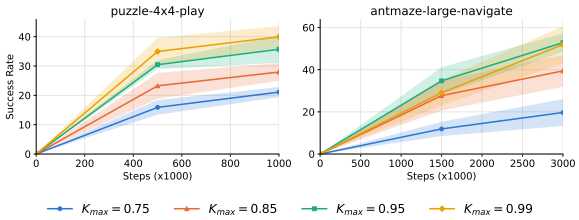


Figure 3: **Effect of K_{max}.** GTSE performance over varying K_{max}, the upper limit of the time-score integral, on puzzle-4x4-play (left) and antmaze-large-navigate (right). Lines show the mean over 8 seeds; shaded regions show 95% C.I.

3) What parameters affect GTSE performance?

Finally, we study the impact of some key parameters. As $k \to 1$, the time score targets grow in magnitude (Eq. 3), making learning more difficult at the extreme tail end. In our GCRL experiments, we truncate the time-score integral at a maximum k, K_{max}, slightly below 1 for reliable estimation and stability. Fig. 3 shows that in general, performance improves with K_{max}, but gains diminish and sometimes plateau (Fig. 3 (right)), potentially because the predicted time score is noisier as $k \to 1$. In Fig. 4, we show the effect of varying the number of time-score evaluations N (Eq. 6) during inference. We see a similar trend to that in Fig. 3, with performance increasing with N and saturating between $N{=}20$ and $N{=}30$. We also evaluate the wall-clock time of all methods (Appendix D.5) and ablate how and when additional representational capacity translates to better downstream control (Appendix E.2).

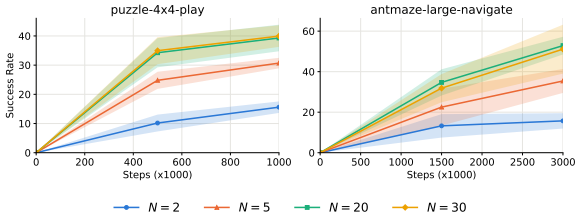


Figure 4: **Effect of N.** GTSE performance over varying N, time-score evaluations during inference, on puzzle-4x4-play (left) and antmaze-large-navigate (right). Lines show the mean over 8 seeds; shaded regions show 95% C.I.

8 Conclusion

In this paper, we presented GTSE, a framework that bridges self-predictive representation learning and generative occupancy models for occupancy density estimation. We developed a mechanism to explain GTSE's improved representational capacity and proposed a variant, GTSE-r, that learns self-predictive representations. Our experiments show that GTSE and its variants enable significantly better downstream control.

Limitations. While we characterize how additional representational capacity translates to downstream RL performance, GTSE does not dynamically allocate representational capacity depending on task difficulty or uncertainty, which is an important direction for future work.

AI usage: In this work, we used generative AI tools to aid in formatting while writing the paper, to assist with code implementation and to assist with refining proofs. We have reviewed all AI-assisted work for correctness including code, formatting, and proofs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility Statement. We provide an anonymized .zip file containing our code implementation. We run control experiments over multiple random seeds and report 95% bootstrapped CIs as error bars in tables and plots throughout Secs. 4–7. We provide additional details on the implementation of all algorithms and the hyperparameters used in Appendices C–D.

REFERENCES

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.

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

Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. *arXiv preprint arXiv:2209.14548*, 2022.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

Kristy Choi, Chenlin Meng, Yang Song, and Stefano Ermon. Density ratio estimation via infinitesimal classification. In *International Conference on Artificial Intelligence and Statistics*, pp. 2552–2573. PMLR, 2022.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

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

Perry Dong, Chongyi Zheng, Chelsea Finn, Dorsa Sadigh, and Benjamin Eysenbach. Value flows. *arXiv preprint arXiv:2510.07650*, 2025.

Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. C-learning: Learning to achieve goals via recursive classification. *arXiv preprint arXiv:2011.08909*, 2020.

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, Matteo Pirotta, Andrea Tirinzoni, Rémi Munos, Alessandro Lazaric, and Ahmed Touati. Temporal difference flows. *arXiv preprint arXiv:2503.09817*, 2025.

Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I Jordan, Joseph E Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. *arXiv preprint arXiv:1803.00101*, 2018.

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

Raj Ghugare, Matthieu Geist, Glen Berseth, and Benjamin Eysenbach. Closing the gap between td learning and supervised learning-a generalisation point of view. In *International Conference on Learning Representations*, volume 2024, pp. 13774–13792, 2024.

Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

Michael Janner, Igor Mordatch, and Sergey Levine. Gamma-models: Generative temporal difference learning for infinite-horizon prediction. *Advances in neural information processing systems*, 33: 1724–1735, 2020.

Leslie Pack Kaelbling. Learning to achieve goals. In *IJCAI*, volume 2, pp. 1094–8, 1993.

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

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Daniel Lawson, Adriana Hugessen, Charlotte Cloutier, Glen Berseth, and Khimya Khetarpal. Self-predictive representations for combinatorial generalization in behavioral cloning. In *International Conference on Learning Representations*, volume 2026, pp. 101559–101586, 2026.

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky TQ Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code. *arXiv preprint arXiv:2412.06264*, 2024.

Grace Liu, Michael Tang, and Benjamin Eysenbach. A single goal is all you need: Skills and exploration emerge from contrastive rl without rewards, demonstrations, or subgoals. In *International Conference on Learning Representations*, volume 2025, pp. 78599–78621, 2025.

Cheng Lu, Kaiwen Zheng, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Maximum likelihood training for score-based diffusion odes by high order denoising score matching. In *International conference on machine learning*, pp. 14429–14460. PMLR, 2022.

Bogdan Mazoure, Benjamin Eysenbach, Ofir Nachum, and Jonathan Tompson. Contrastive value learning: Implicit models for simple offline rl. In *Conference on Robot Learning*, pp. 1257–1267. PMLR, 2023.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Vivek Myers, Chongyi Zheng, Anca Dragan, Sergey Levine, and Benjamin Eysenbach. Learning temporal distances: Contrastive successor features can provide a metric structure for decision-making. *arXiv preprint arXiv:2406.17098*, 2024.

Vivek Myers, Bill Zheng, Benjamin Eysenbach, and Sergey Levine. Offline goal-conditioned reinforcement learning with quasimetric representations. *Advances in Neural Information Processing Systems*, 38:19654–19679, 2026.

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

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. *arXiv preprint arXiv:2410.20092*, 2024.

Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes rl scalable. *arXiv preprint arXiv:2506.04168*, 2025a.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. *arXiv preprint arXiv:2502.02538*, 2025b.

Seohong Park, Deepinder Mann, and Sergey Levine. Dual goal representations. In *International Conference on Learning Representations*, volume 2026, pp. 153307–153324, 2026.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International conference on machine learning*, pp. 5171–5180. PMLR, 2019.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

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

Liam Schramm and Abdeslam Boularias. Bellman diffusion models. *arXiv preprint arXiv:2407.12163*, 2024.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

Wolfram Schultz, Peter Dayan, and P Read Montague. A neural substrate of prediction and reward. *Science*, 275(5306):1593–1599, 1997.

Harshit Sikchi, Rohan Chitnis, Ahmed Touati, Alborz Geramifard, Amy Zhang, and Scott Niekum. Score models for offline goal-conditioned reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2023a.

Harshit Sikchi, Qinqing Zheng, Amy Zhang, and Scott Niekum. Dual rl: Unification and new methods for reinforcement and imitation learning. *arXiv preprint arXiv:2302.08560*, 2023b.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. *Advances in neural information processing systems*, 34:1415–1428, 2021.

Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3 (1):9–44, 1988.

Paul N Swarztrauber. On computing the points and weights for gauss–legendre quadrature. *SIAM Journal on Scientific Computing*, 24(3):945–954, 2003.

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.

Hado Van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

Aravind Venugopal, Jiayu Chen, Xudong Wu, Chongyi Zheng, Benjamin Eysenbach, and Jeff Schneider. Occupancy reward shaping: Improving credit assignment for offline goal-conditioned reinforcement learning. *arXiv preprint arXiv:2604.20627*, 2026.

Eva Yi Xie, Nathaniel D Daw, and Benjamin Eysenbach. Low-rank successor representations capture human-like generalization. In *UniReps: 3rd Edition of the Workshop on Unifying Representations in Neural Models*, 2025.

Hanlin Yu, Arto Klami, Aapo Hyvärinen, Anna Korba, and Omar Chehab. Density ratio estimation with conditional probability paths. *arXiv preprint arXiv:2502.02300*, 2025.

Chongyi Zheng, Ruslan Salakhutdinov, and Benjamin Eysenbach. Contrastive difference predictive coding. *arXiv preprint arXiv:2310.20141*, 2023.

Chongyi Zheng, Benjamin Eysenbach, Homer Walke, Patrick Yin, Kuan Fang, Ruslan Salakhutdinov, and Sergey Levine. Stabilizing contrastive rl: Techniques for robotic goal reaching from offline data. In *International Conference on Learning Representations*, volume 2024, pp. 28236–28264, 2024.

Chongyi Zheng, Seohong Park, Sergey Levine, and Benjamin Eysenbach. Intention-conditioned flow occupancy models. *arXiv preprint arXiv:2506.08902*, 2025.

Chongyi Zheng, Royina Karegoudra Jayanth, and Benjamin Eysenbach. Can we really learn one representation to optimize all rewards? *arXiv preprint arXiv:2602.11399*, 2026.

Appendices

A Proofs

A.1 Time Score Target

$$\partial_k \log p_k(s_k^+ \mid s^+) \;=\; \dim(\mathcal{S})\,\frac{\alpha_k \alpha_k'}{1-\alpha_k^2} \;-\; \frac{\alpha_k \alpha_k'}{1-\alpha_k^2}\,\|\epsilon\|^2 \;+\; \frac{1}{\sqrt{1-\alpha_k^2}}\,\epsilon^\top \alpha_k' s^+$$

The analytical form for the time score target follows directly from Yu et al. (2025).

Observing that $p_k(s_k^+ \mid s^+) = \mathcal{N}(s_k^+; \alpha_k s^+, (1-\alpha_k^2)\, I)$:

$$\log p_k(s_k^+ \mid s^+) = -\frac{1}{2}\log\det((1-\alpha_k^2)I) - \frac{1}{2}(s_k^+ - \alpha_k s^+)^\top (1-\alpha_k^2)^{-1}(s_k^+ - \alpha_k s^+) + \text{const} \tag{10}$$

$$= -\frac{1}{2}\log(1-\alpha_k^2)^{\dim(\mathcal{S})} - \frac{1}{2(1-\alpha_k^2)}\|s_k^+ - \alpha_k s^+\|_2^2 + \text{const} \tag{11}$$

$$= -\frac{\dim(\mathcal{S})}{2}\log((1-\alpha_k^2)) - \frac{1}{2(1-\alpha_k^2)}\|s_k^+ - \alpha_k s^+\|_2^2 + \text{const}. \tag{12}$$

$$\begin{aligned}\partial_k \log p_k(s_k^+ \mid s^+) &= -\frac{\dim(\mathcal{S})}{2}\frac{\partial_k(1-\alpha_k^2)}{1-\alpha_k^2} + \frac{\partial_k(1-\alpha_k^2)}{2(1-\alpha_k^2)^2}\left\|s_k^+ - \alpha_k s^+\right\|_2^2 \\ &\quad + \frac{1}{1-\alpha_k^2}\left(s_k^+ - \alpha_k s^+\right)^\top \partial_k \alpha_k\, s^+ \\ &= \dim(\mathcal{S})\,\frac{\alpha_k \alpha_k'}{1-\alpha_k^2} - \frac{\alpha_k \alpha_k'}{1-\alpha_k^2}\,\|\epsilon\|_2^2 + \frac{\alpha_k'}{\sqrt{1-\alpha_k^2}}\,\epsilon^\top s^+. \end{aligned} \tag{13}$$

A.2 Optimal Geometric Time Score Estimator

Lemma. *The estimator that minimizes Eq. 4 is given by:*

$$\begin{aligned} s_\theta^*(s_k^+, k, s, a) &= \mathbb{E}_{s^+ \sim p_k(\cdot \mid s_k^+, s, a)}\left[\partial_k \log p_k(s_k^+ \mid s^+)\right] \\ &= \mathbb{E}_{s^+ \sim p_k(\cdot \mid s_k^+, s, a)}\left[\frac{\alpha_k \alpha_k'}{1-\alpha_k^2}\left(\dim(\mathcal{S}) - \|\epsilon\|^2\right) + \frac{\alpha_k'}{\sqrt{1-\alpha_k^2}}\,\epsilon^\top s^+\right], \qquad \epsilon = \frac{s_k^+ - \alpha_k s^+}{\sqrt{1-\alpha_k^2}}. \end{aligned} \tag{14}$$

We start by re-writing Eq. 4:

$$\mathbb{E}_{s^+ \sim d^{\pi_D}(\cdot \mid s,a)}\, \mathbb{E}_{k,\, s_k^+ \mid s^+}\left[\lambda(k)\left(\partial_k \log p_k(s_k^+ \mid s^+) - s_\theta(s_k^+, k, s, a)\right)^2\right] \tag{15}$$

We know that

$$p_k(s_k^+ \mid s^+, s, a) = p_k(s_k^+ \mid s^+) = \mathcal{N}\left(s_k^+; \alpha_k s^+, (1-\alpha_k^2)\, I\right)$$

(note that $s^+ = s_1^+$).

By Bayes rule:

$$p_k(s_k^+ \mid s^+, s, a) d^{\pi_D}(s^+ \mid s, a) = p_k(s^+ \mid s_k^+, s, a) p_k(s_k^+ \mid s, a).$$

For a fixed (s, a, k), this allows us to rewrite the expectation in Eq. 4 as:

$$\mathbb{E}_{s_k^+ \sim p_k(\cdot \mid s,a)}\, \lambda(k) \mathbb{E}_{s^+ \sim p_k(\cdot \mid s_k^+, s, a)}\left[\left(\partial_k \log p_k(s_k^+ \mid s^+) - s_\theta(s_k^+, k, s, a)\right)^2\right]. \tag{16}$$

For ease of notation, let

$$s_\theta(s_k^+, k, s, a) = f, \qquad (s, a) = c, \qquad \partial_k \log p_k(s_k^+ \mid s^+) = y(s_k^+ \mid s^+).$$

Then:

$$l = \mathbb{E}_{s_k^+ \sim p_k(\cdot|c)}\, \lambda(k) \mathbb{E}_{s^+ \sim p_k(\cdot|s_k^+, c)} \Big[\big(y(s_k^+ \mid s^+) - f \big)^2 \Big]. \tag{17}$$

Now, by taking the derivative with respect to f and setting it to 0, we get:

$$f^* = \mathbb{E}_{s^+ \sim p_k(\cdot|s_k^+, c)} y(s_k^+ \mid s^+).$$

Since the second derivative is $2\lambda(k)$, the stationary point is a minimum when $\lambda(k) > 0$.

By expanding the time score target (Eq. 3), we get:

$$\begin{aligned} s_\theta^*(s_k^+, k, s, a) &= \mathbb{E}_{s^+ \sim p_k(\cdot|s_k^+, s, a)} \left[\partial_k \log p_k(s_k^+ \mid s^+) \right] \\ &= \mathbb{E}_{s^+ \sim p_k(\cdot|s_k^+, s, a)} \left[\frac{\alpha_k \alpha_k'}{1 - \alpha_k^2} \big(\dim(\mathcal{S}) - \|\epsilon\|^2 \big) + \frac{\alpha_k'}{\sqrt{1 - \alpha_k^2}} \epsilon^\top s^+ \right], \qquad \epsilon = \frac{s_k^+ - \alpha_k s^+}{\sqrt{1 - \alpha_k^2}}. \end{aligned} \tag{18}$$

Furthermore, noting that

$$p_k(s_k^+ \mid c) = \int p_k(s_k^+ \mid s^+) d^{\pi_D}(s^+ \mid c)\, ds^+,$$

we can see that this is equivalent to the marginal time score of the occupancy measure:

$$\begin{aligned} s_\theta^*(s_k^+, k, c) &= \int p_k(s^+ \mid s_k^+, c)\, \partial_k \log p_k(s_k^+ \mid s^+)\, ds^+ \\ &= \int \frac{p_k(s_k^+ \mid s^+)\, d^{\pi_D}(s^+ \mid c)}{p_k(s_k^+ \mid c)} \frac{\partial_k p_k(s_k^+ \mid s^+)}{p_k(s_k^+ \mid s^+)}\, ds^+ \\ &= \frac{1}{p_k(s_k^+ \mid c)} \int \partial_k p_k(s_k^+ \mid s^+)\, d^{\pi_D}(s^+ \mid c)\, ds^+ \\ &= \frac{1}{p_k(s_k^+ \mid c)}\, \partial_k \int p_k(s_k^+ \mid s^+)\, d^{\pi_D}(s^+ \mid c)\, ds^+ \\ &= \frac{\partial_k p_k(s_k^+ \mid c)}{p_k(s_k^+ \mid c)} \\ &= \partial_k \log p_k(s_k^+ \mid s, a). \end{aligned}$$

A.3 Temporal Difference loss for Geometric Time Score Matching

The GTSE loss for occupancy density estimation can be written as

$$\mathcal{L}_{\text{GTSE}}(\theta) = (1 - \gamma)\, \mathcal{L}_{\text{next}}(\theta) + \gamma\, \mathcal{L}_{\text{future}}(\theta), \qquad \forall\, (s, a, s', a') \in \mathcal{D},$$

where

$$\begin{aligned} \mathcal{L}_{\text{next}}(\theta) &= \mathbb{E}_{k,\, s_k'|s'} \Big[\lambda(k) \big(\partial_k \log p_k(s_k' \mid s') - s_\theta(s_k', k, s, a) \big)^2 \Big], \\ \mathcal{L}_{\text{future}}(\theta) &= \mathbb{E}_{s^+ \sim d^{\pi_D}(\cdot|s', a')}\, \mathbb{E}_{k,\, s_k^+|s^+} \Big[\lambda(k) \big(\text{sg} \left[s_{\bar{\theta}}(s_k^+, k, s', a') \right] - s_\theta(s_k^+, k, s, a) \big)^2 \Big]. \end{aligned}$$

We know that (Eq. 4):

$$\mathcal{L}(\theta) = \mathbb{E}_{s^+ \sim d^{\pi_D}(\cdot|s,a)} \mathbb{E}_{k, s_k^+|s^+} \Big[\lambda(k) \big(\partial_k \log p_k(s_k^+ \mid s^+) - s_\theta(s_k^+, k, s, a) \big)^2 \Big],$$

noting that $d^{\pi_D}(s^+ \mid s, a)$ has a recursive form (Eq. 2). Accordingly, s^+ is sampled from a mixture of distributions; with a probability of $(1 - \gamma)$, $s^+ = s'$ and with a probability of γ, $s^+ \sim d^{\pi_D}(\cdot \mid s', a')$. By the law of total expectation, we can then write:

$$\mathcal{L}(\theta) = \mathbb{E}_{s'\sim p(\cdot|s,a),\, a'\sim\pi_D(\cdot|s')}\Big[(1-\gamma)\,\mathcal{L}_{\text{next}}(\theta) + \gamma\,\mathcal{L}_{\text{future}}(\theta)\Big], \qquad \forall\,(s,a)\in\mathcal{D},$$

where

$$\mathcal{L}_{\text{next}}(\theta) = \mathbb{E}_{k,\, s'_k|s'}\Big[\lambda(k)\big(\partial_k \log p_k(s'_k \mid s') - s_\theta(s'_k,k,s,a)\big)^2\Big],$$

$$\mathcal{L}_{\text{future}}(\theta) = \mathbb{E}_{s^+\sim d^{\pi_D}(\cdot|s',a')}\,\mathbb{E}_{k,\, s^+_k|s^+}\Big[\lambda(k)\big(\partial_k \log p_k(s^+_k \mid s^+) - s_\theta(s^+_k,k,s,a)\big)^2\Big].$$

By Lemma 1, $\mathcal{L}_{\text{future}}$ is minimized by an estimator that averages time-score targets over $d^{\pi_D}(\cdot \mid s', a')$. Since this estimator is not available in closed form, we instead bootstrap it with the estimate of s_θ at (s', a'). In practice, we find that for $\mathcal{L}_{\text{future}}$, bootstrapping the output of a time-delayed version of the time-score network, $s_{\bar\theta}(s^+_k, k, s', a')$, enables cross-trajectory stitching when estimating densities and stabilizes temporal difference learning (Mnih et al., 2015; Van Hasselt et al., 2018). We therefore rewrite $\mathcal{L}_{\text{future}}$ as:

$$\mathcal{L}_{\text{future}}(\theta) = \mathbb{E}_{s^+\sim d^{\pi_D}(\cdot|s',a')}\,\mathbb{E}_{k,\, s^+_k|s^+}\Big[\lambda(k)\big(\operatorname{sg}\big[s_{\bar\theta}(s^+_k,k,s',a')\big] - s_\theta(s^+_k,k,s,a)\big)^2\Big].$$

A.4 Legendre Decomposition of Time Score (Proposition 1)

Proposition (Legendre decomposition of the time score)**.** *Assume the time score factorizes as*

$$\partial_k \log p_k(g \mid s,a) \;=\; \phi(s,a,k)^\top \psi(g,k),$$

and that both ϕ and ψ are δ-dimensional and square-integrable in $k\in[0,1]$. Let $\{e_l(k)\}_{l\ge 0}$ denote the shifted orthonormal Legendre basis on $[0,1]$. Then, for any $g\in\mathcal{D}$,

$$\log d^{\pi_D}(g \mid s,a) \;=\; \sum_{l=0}^{\infty} \hat\phi_l(s,a)^\top \hat\psi_l(g) \;+\; \log p_0(g), \tag{19}$$

where $\hat\phi_l(s,a)$ and $\hat\psi_l(g)$ are the l-th Legendre coefficient vectors of $\phi(s,a,\cdot)$ and $\psi(g,\cdot)$ in k.

We assume that both $\phi(s,a,\cdot)$ and $\psi(g,\cdot)$ are square-integrable functions of $k\in[0,1]$:

$$\int_0^1 \|\phi(s,a,k)\|^2\,dk < \infty, \qquad \int_0^1 \|\psi(g,k)\|^2\,dk < \infty.$$

Each component of ϕ and ψ thus lives in the Hilbert space $\mathbf{L}^2(0,1)$ with inner product: $\langle u, v\rangle = \int_0^1 u(k)v(k)\,dk$. With P_l being the Legendre polynomial of degree l, the shifted, normalized Legendre polynomials form an orthonormal basis for $\mathbf{L}^2(0,1)$:

$$e_l(k) = \sqrt{2l+1}\,P_l(2k-1), \qquad l = 0,1,2,\ldots$$

This allows us to expand ϕ and ψ as follows:

$$\phi(s,a,k) = \sum_{l=0}^{\infty} \hat\phi_l(s,a)\,e_l(k), \qquad \hat\phi_l(s,a) = \int_0^1 \phi(s,a,k)\,e_l(k)\,dk$$

$$\psi(g,k) = \sum_{l=0}^{\infty} \hat\psi_l(g)\,e_l(k), \qquad \hat\psi_l(g) = \int_0^1 \psi(g,k)\,e_l(k)\,dk.$$

By integrating the time score to recover log-density and applying Parseval's identity elementwise, for any $g\in\mathcal{D}$:

$$\log\frac{d^{\pi_D}(g \mid s,a)}{p_0(g)} = \int_0^1 \phi^\top\psi\,dk = \int_0^1 \left(\sum_j \hat\phi_j e_j\right)^\top \left(\sum_i \hat\psi_i e_i\right) dk$$

$$= \sum_{j,i} \hat\phi_j^\top \hat\psi_i \underbrace{\int_0^1 e_j e_i\,dk}_{=\,\mathbf{1}[i=j]}$$

$$\boxed{\log d^{\pi_D}(g \mid s,a) = \sum_{l=0}^{\infty} \hat\phi_l(s,a)^\top \hat\psi_l(g) + \log p_0(g).}$$

B ALGORITHM PSEUDOCODE

Algorithm 1: Geometric Time Score Estimators

Input: Time score s_θ, target time score $s_{\bar{\theta}}$, BC flow policy π_ω, dataset $\mathcal{D}$, learning rate η

for *each iteration* **do**
- Sample $(s, a, s', a', s^+, g) \sim \mathcal{D}$;
- Sample noise $s'_0, s^+_0 \sim \mathcal{N}(0, I)$ and $k \sim \mathcal{U}[0, 1]$;
- Compute $\mathcal{L}(\theta)$ (Eq. 5) and update $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$;
- Compute $\mathcal{L}_{\text{BC}}(\omega)$ (Eq. 20; Appendix C) and update $\omega \leftarrow \omega - \eta \nabla_\omega \mathcal{L}_{\text{BC}}(\omega)$;
- Update the target $s_{\bar{\theta}}$ using Polyak averaging;

end

C IMPLEMENTATION DETAILS

BC Policy: The goal-conditioned BC policy transforms d_a-dimensional noise $a_0 \sim \mathcal{N}(0, I_{d_a})$ to samples from $a_1 \sim \pi_D(.|s, g)$. It is trained using the BC flow-matching loss from Park et al. (2025b):

$$\mathcal{L}_{BC}(\omega) = \mathbb{E}_{\substack{s,\, a=a_1, g\sim\mathcal{D},\\ a_0\sim\mathcal{N}(0,I_{d_a}),\\ t\sim\text{Unif}([0,1])}} \left\| v_\omega(t, s, a_t, g) - (a_1 - a_0) \right\|_2^2, \tag{20}$$

where t is the ODE time, $v_\omega(t, s, a, g) : [0, 1] \times \mathcal{S} \times \mathcal{S} \times \mathbb{R}^{d_a} \to \mathbb{R}^{d_a}$ is a state- and time-dependent vector field ω. At inference time, we generate samples by running an Euler sampler for the ODE defined by v_ω as in Park et al. (2025b).

Time score. We use $\alpha_k = k$ in all our experiments. For GTSE, we train the vectorized version of the time score (Yu et al., 2025), i.e., each state dimension is supervised by a denser dimension-wise target rather than a scalar target:

$$\text{vec}\, \partial_k \log p_k(s^+_k \mid s^+) \;=\; \frac{\alpha_k \alpha'_k}{1 - \alpha_k^2} \;-\; \frac{\alpha_k \alpha'_k}{1 - \alpha_k^2}\, \epsilon^{\odot 2} \;+\; \frac{1}{\sqrt{1 - \alpha_k^2}}\, \epsilon \odot \left(\alpha'_k s^+\right), \tag{21}$$

where $\epsilon^{\odot 2}$ and $\epsilon \odot (\alpha'_k s^+)$ denote elementwise operations.

We sample $s^+ \sim d^{\pi_D}(\cdot \mid s', a')$ as a future in-trajectory state of the trajectory at (s', a') or at random from $\mathcal{D}$ with varying probabilities. In practice, we also truncate the maximum k the time-score network is evaluated at during inference (Eq. 6) to a value slightly lower than 1 and rescale the contribution at each k accordingly. We call this K_{max}. We find that doing so leads to better downstream performance. We also add a small amount of gaussian noise to the goal to stabilise training near $k = 1$, especially when bootstrapping from states close to the goal.

Legendre-truncated versions GTSE-b_L. To estimate the Legendre-decomposed critic for each L, the coefficient vectors $\hat{\phi}_l$ and $\hat{\psi}_l$ are estimated using the Gauss-Legendre quadrature method (Swarztrauber, 2003). Specifically we use the `numpy` package function `np.polynomial.legendre.leggauss`. For each L, we evaluate ψ and ϕ at $2(L+1)$ quadrature points on $[0, K_{max}]$ with appropriate rescaling. The log occupancy density estimate for any $(s, a, g) \in \mathcal{D}$ is computed as:

$$\log \hat{d}^{\pi_D}(g \mid s, a) = \sum_{l=0}^{L} \hat{\phi}_l^\top \hat{\psi}_l + \log p_0(g), \tag{22}$$

with $\hat{\phi}_l = \sum_i w_i\, \phi(s, a, k_i)\, e_l(k_i)$, and similarly for $\hat{\psi}_l$, where k_i correspond to quadrature points and w_i correspond to quadrature weights.

D Additional Information

D.1 Tasks and Datasets

We perform experiments on OGBench (Park et al., 2024) for offline GCRL tasks.

Pointmaze dataset. We generate the pointmaze dataset using the official OGBench data generation code. Specifically, we use the **large** pointmaze. At each episode, an agent defined by a point mass spawns in a random free cell. Every 10 steps, the agent chooses a random direction and takes noisy actions to move in the chosen direction. We collect 1000 episodes, each of which is 501 steps. To compute the ground-truth log-occupancy density, we discretize the maze into a grid, compute the transition matrix for the grid and with the maze partitioned into n grid cells, count transitions between cells in the dataset and row-normalize to obtain the transition matrix P. We then compute the discounted occupancy matrix as:

$$D = (1-\gamma)\sum_{t\geq 0}\gamma^t P^t = (1-\gamma)(I-\gamma P)^{-1}.$$

where t is the episode timestep.

OGBench GCRL tasks. As mentioned in Sec. 7, OGBench tasks can be broadly categorized as maze navigation, cube manipulation, puzzle manipulation and scene tasks. We perform experiments on selected tasks from each category:

- Maze navigation: We use **antmaze-large-navigate**, a maze dataset containing navigation data from an 8 degrees of freedom (DoF) Ant robot. Notably, **antmaze-large** has trajectories upto 1000 timesteps long and contains noisy expert trajectories. The agent must learn to navigate to specific goal states purely from offline data, necessitating effective long-horizon credit assignment.
- Scene: The scene tasks involve manipulating multiple different objects: cube, window, drawer, locks and buttons. At test time, the agent has to arrange objects into a given goal configuration different from that present in the training dataset. This task therefore tests sequential long-horizon planning capabilities. We perform experiments on the noisy expert (**scene-play**) and highly suboptimal (**scene-noisy**) datasets.
- Puzzle manipulation: These tasks test the combinatorial generalization abilities of the agent under long horizons and require solving the Lights Out puzzle (Park et al., 2024). The agent has to achieve an appropriate goal configuration of puzzle cell colors by pressing buttons by controlling a robot arm. We perform experiments on the noisy expert (**puzzle-4x4-play**) and highly suboptimal (**puzzle-4x4-noisy**) datasets.
- Cube manipulation: These tasks involve learning robot arm manipulation to pick, place and stack cubes according to the task specification. We perform experiments with **cube-triple**, which is considered to be a very hard task with near-zero performance throughout the bench (Park et al., 2024). We use both noisy expert (**cube-triple-play**) and highly suboptimal (**cube-triple-noisy**) datasets.

OGBench reward-based tasks. Each reward-based task corresponds to a single evaluation goal. The dataset is generated by relabelling with semi-sparse rewards. The semi-sparse reward function is the number of subtasks remaining in order to reach the goal state (Park et al., 2024; 2025b). In **antmaze-large-navigate** rewards are -1 at all states except the goal. In **scene-play**, rewards range between $-n_{task}$ and 0, where n_{task} is the number of subtasks.

D.2 Additional Information on Baselines

- **CRL** (Eysenbach et al., 2022): CRL uses contrastive learning with the infoNCE (Oord et al., 2018) loss to train a pair of state-action ($\phi(s,a)$) and goal ($\psi(g)$) representations:

$$\max_{\phi,\psi} \mathbb{E}_{\substack{(s_i,a_i)\sim p_D(s,a),\\ s^{+(i)}\sim d_D^\pi(\cdot|s_i,a_i),\quad i=1,\ldots,N}} \left[\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp\big(\phi(s_i,a_i)^\top\psi\big(s^{+(i)}\big)\big)}{\sum_{j=1}^{N}\exp\big(\phi(s_j,a_j)^\top\psi\big(s^{+(i)}\big)\big)}\right]. \quad (23)$$

- **FB** (Zheng et al., 2026): We train a version of one-step FB from Zheng et al. (2026) with a state-action conditioned forward representation $F(s, a)$ and a goal-conditioned backward representation $B(g)$.
- **TD-JEPA** (Bagatella et al., 2025): TD-JEPA uses a joint-embedding predictive architecture to learn a state representation $\phi(s)$, a goal representation $\psi(g)$ and two predictors T_ϕ and T_ψ. In our implementation, the occupancy density estimator is given by $T_\phi\big(\phi(s), a, \psi(g)\big)^\top \psi(g)$ and trained as follows:

$$\mathcal{L}(\phi, T_\phi, \psi) = \mathbb{E}_{(s,a,s',a',g)\sim\mathcal{D}}\left[\left\|T_\phi\big(\phi(s), a, \bar{\psi}(g)\big) - \big(\bar{\psi}(s') + \gamma\, \bar{T}_\phi\big(\bar{\phi}(s'), a', \bar{\psi}(g)\big)\big)\right\|_2^2\right],$$

$$\mathcal{L}(\psi, T_\psi, \phi) = \mathbb{E}_{(s,a,s',a',g)\sim\mathcal{D}}\left[\left\|T_\psi\big(\psi(s), a, \bar{\psi}(g)\big) - \big(\bar{\phi}(s') + \gamma\, \bar{T}_\psi\big(\bar{\psi}(s'), a', \bar{\psi}(g)\big)\big)\right\|_2^2\right],$$

  where $\bar{\psi}$ and $\bar{\phi}$ are target networks.
- **BYOL-γ** (Lawson et al., 2026): Our implementation of BYOL-γ trains ψ_b and ψ_f to estimate a score $S(s, a, g) = \big\langle \psi_f\big(\phi(s), a\big), \phi(g)\big\rangle$ to rank actions during inference. The representations are trained as follows, based on (Lawson et al., 2026) :

$$\mathcal{L}_{\text{BYOL-}\gamma}(\psi_f, \psi_b) = \mathbb{E}_{s,a,g\sim\mathcal{D}}\left[f\big(\psi_f\big(\phi(s), a\big), \phi(g)\big) + f\big(\phi(s), \psi_b(\phi(g))\big)\right].$$

  Compared to Lawson et al. (2026) which also trains ϕ, we found that using randomly initialized features and freezing ϕ gave better performance.
- **Dual GR** (Park et al., 2026) trains state and goal representations $\phi(s)$ and $\psi(g)$ to estimate temporal distances by parameterizing them as components of a goal-conditioned value function $V(s, g)$ trained along with a Q function $Q(s, a, g)$ using Implicit Q Learning (Kostrikov et al., 2021). In our implementation, we use the learned $Q(s, a, g)$ for action ranking.
- **TMD** (Myers et al., 2026): Temporal Metric Distillation learns state-action $\phi(s, a)$ and goal $\psi(g)$ representations trained such that the quasimetric distance $d(\phi(s, a), \psi(g))$ is predictive of the temporal connectivity of g from (s, a) under d^{π_D}.
- **SMORE** (Sikchi et al., 2023a): By posing offline GCRL as an occupancy matching problem, SMORE solves for its dual formulation by learning a score function, called the dual value function using a constrained optimization objective.
- **TD-flows** (Farebrother et al., 2025): Temporal-difference flows learns a flow-matching occupancy model using a TD-based loss:

$$\mathcal{L}_{\text{flow}}(\theta) = (1-\gamma)\,\mathcal{L}_{\text{next}}(\theta) + \gamma\,\mathcal{L}_{\text{future}}(\theta); \qquad \forall\,(s, a, s', a') \in \mathcal{D} \tag{24}$$

$$\mathcal{L}_{\text{next}}(\theta) = \mathbb{E}_{\substack{s'=x_1\sim\mathcal{D}\\ x_0\sim\mathcal{N}(0, I_d)\\ k\sim\text{Unif}([0,1])}}\left[\|v_\theta(k, s, a, x_k) - (x_1 - x_0)\|_2^2\right];$$

$$\mathcal{L}_{\text{future}}(\theta) = \mathbb{E}_{\substack{s^+=x_1\sim d_{\theta^-}^{\pi_D}(.|s',a')\\ x_0\sim\mathcal{N}(0, I_d)\\ k\sim\text{Unif}([0,1])}}\left[\left\|v_\theta(k, s, a, x_k) - \mathbf{sg}[v_{\theta^-}(k, s', a', x_k)]\right\|_2^2\right].$$

  where v_θ is the velocity field. Once the velocity field is trained, we test out two estimators to recover $d^{\pi_D}(g \mid s, a)$:

  1. Monte-Carlo estimator: We generate samples from the trained TD-flow model and estimate a proxy for $d^{\pi_D}(g \mid s, a)$ as:

$$\hat{d}^{\pi_D}(g \mid s, a) \;\propto\; \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\Big[\big\|s_f^{(i)} - g\big\| < \epsilon\Big], \qquad s_f^{(i)} \sim \hat{p}_{\text{TD-flow}}(\cdot \mid s, a),$$

     where $N = 32$ and $\epsilon = 10^{-3}$.
  2. Hutchinson trace estimator (Hutchinson, 1989; Grathwohl et al., 2018):

$$\log p_1(g_1 \mid s, a) = \log p_0(g_0) - \int_0^1 \nabla \cdot v_\theta(k, s, a, g_k)\, dk, \tag{25}$$

$$\nabla \cdot v_\theta(k, s, a, g_k) = \frac{1}{m}\sum_{j=1}^{m} \epsilon^{(j)\top} \frac{\partial v_\theta}{\partial g}(k, s, a, g_k)\, \epsilon^{(j)}, \qquad \epsilon^{(j)} \sim \mathcal{N}(0, I_d). \tag{26}$$

     We use $m = 32$ and do not see significant gains from using a higher m.

In general, we find that the Hutchinson trace estimator is better at estimating occupancy density than the MC estimator and therefore, we use it in our experiments.

- **NF** (Ghugare & Eysenbach, 2025): We use the same Normalizing Flows architecture from (Ghugare & Eysenbach, 2025) and use it to estimate occupancy density, as detailed in Sec. 4.3 and Sec. 6.4 of (Ghugare & Eysenbach, 2025).
- For all baselines, we use the best hyperparameters provided in their respective papers. If these are not available, we perform hyper-parameter tuning.
- **Goal sampling for all algorithms**: We sample goals/future-states in three different ways, according to categorical distribution:
  - $p^{\mathcal{D}}_{\text{cur}}(g|s)$: Dirac delta distribution centered at the current state s.
  - $p^{\mathcal{D}}_{\text{traj}}(g|s)$: goal is sampled from future states of s from the same trajectory using geometric sampling.
  - $p^{\mathcal{D}}_{\text{rand}}(g|s)$: goal is uniformly sampled from $\mathcal{D}$.

D.3 Training Iterations

Table 4: Training iterations per task. All algorithms are trained for the same number of iterations.

Dataset	Training Iterations
toy MDP	250k
puzzle-4x4-play	1M
scene-play	1M
cube-triple-play	3M
antmaze-large-navigate	3M
pointmaze-large-diverse	3M
puzzle-4x4-noisy	1M
scene-noisy	1M
cube-triple-noisy	3M

D.4 Hyperparameters

Common hyperparameters are the same across all methods and are shown in Table. 5.

Table 5: Common hyperparameters.

Hyperparameter	Value
Optimizer	Adam (Kingma & Ba, 2014)
Learning rate	0.0003
Batch size	1024
Architecture	MLP
MLP size	{512, 512, 512, 512}
Representation dimensions (GCRL expts)	512
Representation dimensions (frozen encoder expts)	40
Nonlinearity	GELU (Hendrycks & Gimpel, 2016) for flow-matching policy
Layer normalization	True
Actions sampled for Best-of-M selection	16
FQL α (Park et al., 2025b)	300

Hyperparameters specific to GTSE are shown in Table 6 and Table 7. Here, $p^{\mathcal{D}}$ denotes the tuple $\left(p^{\mathcal{D}}_{\mathrm{cur}}, p^{\mathcal{D}}_{\mathrm{traj}}, p^{\mathcal{D}}_{\mathrm{rand}}\right)$.

Table 6: GTSE hyperparameters.

Hyperparameter	Value
Nonlinearity	ELU (Clevert et al., 2015) for GTSE, GELU for GTSE-r
Discount factor γ	0.995 for cube-triple and cube-triple-noisy, 0.99 otherwise
Time embedding size	64 (toy-MDP and pointmaze) & 96 (other experiments)
Goal noise std.	0.0001

Table 7: Task-specific hyperparameters for GTSE training

Dataset	τ (target update rate)	GTSE ($p^{\mathcal{D}}$)	N (Eq. 6)	policy ($p^{\mathcal{D}}_{\text{cur}}, p^{\mathcal{D}}_{\text{traj}}, p^{\mathcal{D}}_{\text{rand}}$)	K_{max}
toy MDP	0.005	(0, 1, 0)	30	(0, 1, 0)	0.9999
puzzle-4x4-play	0.005	(0, 1, 0)	30	(0, 1, 0)	0.99
puzzle-4x4-noisy	0.005	(0, 1, 0)	30	(0, 1, 0)	0.95
cube-triple-play	0.003	(0, 1, 0)	30	(0, 0.5, 0.5)	0.925
cube-triple-noisy	0.005	(0, 1, 0)	30	(0, 0.5, 0.5)	0.925
scene-play	0.003	(0, 1, 0)	30	(0, 0.5, 0.5)	0.925
scene-noisy	0.005	(0, 1, 0)	30	(0, 0.5, 0.5)	0.925
antmaze-large-navigate	0.005	(0, 1, 0)	20	(0, 1, 0)	0.95
pointmaze-large-diverse	0.005	(0, 1, 0)	30	(0, 1, 0)	0.9999

D.5 Wall-clock time

We report wall-clock time during both training and inference in Table 8. The training time is the median time required for each training step reported over 1000 steps. The inference time is the median time required for computing the occupancy density estimate, reported over 1000 steps as well. We see from Table 8 that GTSE's training and inference times are comparable to most state-of-the-art methods, despite its additional representational capacity and substantially better downstream performance. In fact, it is the fastest method during training.

Table 8: Wall-clock runtime comparison reported in milliseconds.

	CRL	**FB**	**TD-JEPA**	**BYOL-γ**	**Dual GR**	**TMD**	**SMORE**	**TD-flows**	**NF**	**GTSE-b**	**GTSE-r**	GTSE
Training (ms) ↓	3.12	3.66	5.12	3.77	4.51	44.55	2.53	2.26	4.62	4.49	3.59	**2.17**
Inference (ms) ↓	0.24	0.29	0.45	0.29	0.26	0.23	**0.19**	0.70	0.64	0.55	0.36	0.31

E Additional Experiments

E.1 Occupancy measure structure in antmaze and GTSE results

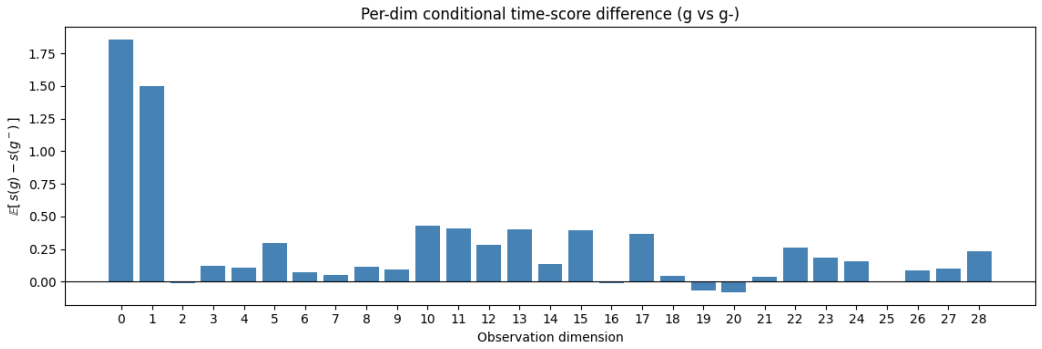


Figure 5: Difference in time score per dimension, averaged over k, between the in-trajectory goal and randomly sampled goals on **antmaze-large-navigate**.

As mentioned in Sec. 7, the GTSE-r, the bilinear variant, performs better on **antmaze-large-navigate**. We hypothesize that this is due to the occupancy measure being inherently low-dimensional. To verify this, we first train GTSE on **antmaze-large-navigate** and then measure the dimension-wise difference in time score, averaged over k, between 10k in-trajectory goals and 10k randomly sampled goals. We plot the mean differences in Fig. 5. The results show that the

difference occurs primarily in dimensions 0 and 1 of the state, corresponding to the x and y positions of the agent. This supports our hypothesis that the occupancy measure is low-dimensional, and suggests that goal likelihood is almost exclusively determined by the (x, y) position. Furthermore, we find that by training GTSE by masking all dimensions except 0 and 1, performance on **antmaze-large-navigate** improves from $53 \pm 4\%$ to $68 \pm 5\%$, reaching a level comparable to GTSE-r.

E.2 When does additional representational capacity translate to better control?

Table 9: Performance comparison of GTSE-b_L and GTSE-b on OGBench GCRL tasks. While performance improves with truncation order on **antmaze**, it degrades with truncation order on **puzzle**.

Dataset	GTSE-b_0	GTSE-b_2	GTSE-b_{10}	GTSE-b
puzzle-4x4-play	36 ± 3	30 ± 2	29 ± 7	30 ± 3
antmaze-large-navigate	63 ± 5	77 ± 4	77 ± 3	77 ± 1

Here, we investigate when the additional representational capacity induced by occupancy density estimation along probability paths translates to better downstream policy performance. To do so, we ablate the GCRL performance of GTSE-b and GTSE-b_L, the post-hoc truncated variants, for different L on a locomotion task (**antmaze-large-navigate**) as well as a manipulation task(**puzzle-4x4-play**). Table 9 summarizes these results. On **puzzle-4x4-play**, GTSE-b_0 performs best out of all variants whereas on **antmaze-large-navigate**, the trend is reversed. We hypothesize that this difference arises due to the structure of the occupancy measure in each dataset.

Specifically, we see that performance improves with truncation order L and later saturates on **antmaze-large-navigate** whereas on **puzzle-4x4-play**, GTSE-b_0 performs best and performance decreases with L. We attribute these trends to differences in how the time score evolves along the probability path on each task. In Antmaze, higher-order Legendre modes progressively recover finer time score structure relevant to control. On the other hand, on Puzzle, additional modes do not recover information that improves control and may instead fit noise, slightly degrading performance.

To verify our hypothesis, we first measure whether truncation order improves goal discrimination. For each of the two tasks, we sample 10k (s, a, g^+, g^-) tuples, where g^+ corresponds to the last state of the trajectory corresponding to (s, a) and g^- is a random state. We then measure the gap: $\Delta_L = \mathbb{E}\left[\log \frac{\hat{d}_L^{\pi_D}(g^+|s,a)}{p_0(g^+)} - \log \frac{\hat{d}_L^{\pi_D}(g^-|s,a)}{p_0(g^-)}\right]$; where L corresponds to the truncation order. Table 10 shows that Δ_L increases substantially by 19.6% between GTSE-b_0 and GTSE-b on Antmaze whereas the increment in Puzzle is much more modest at 2.4%.

To determine whether this increase in Δ_L translates to better control, we now measure how well each variant ranks actions. Since the ground truth occupancy measure is not available, we measure the Spearman rank correlation coefficient ρ of each variant compared to the best-performing algorithm on each task out of GTSE, GTSE-b, and truncated variants: GTSE-b for Antmaze and GTSE for Puzzle. Specifically, we randomly sample 10k (s, g) pairs and sample 32 actions per pair. Table 11 shows that ρ improves with L from 0.94 to 0.98 on Antmaze, suggesting that goal-discriminative information recovered at higher L translates to better downstream performance. On Puzzle, however, ρ falls substantially from GTSE-b_0 (0.95) to GTSE-b_{10} (0.78). Thus, although Δ_L increases slightly with L on Puzzle, the increase in truncation order does not translate downstream, consistent with our hypothesis that higher L fits noise, degrading performance.

Taken together, the results in this section paint a nuanced relationship between occupancy estimation, representational capacity of the estimator and downstream policy performance when learning along probability paths.

Table 10: Δ_L for different L. Higher Δ_L indicates better discrimination between goals. The increments in ability with L are greater on **antmaze** than on **puzzle**.

Dataset	GTSE-b_0	GTSE-b_2	GTSE-b_{10}	GTSE-b
puzzle-4x4-play	1.27	1.28	1.28	1.30
antmaze-large-navigate	2.70	2.95	2.98	3.23

Table 11: Spearman rank correlation (ρ) of each GTSE variant compared to the best-performing algorithm on each task. While ρ increases with L on **antmaze**, it decreases with L on **puzzle**.

Dataset	GTSE-b_0	GTSE-b_2	GTSE-b_{10}
puzzle-4x4-play	0.95	0.87	0.78
antmaze-large-navigate	0.94	0.97	0.98