IMAGINING GOALS WITH GEOMETRIC HORIZON MODELS FOR POLICY GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Training from offline data has allowed for substantial progress in domains such as robotics, leading to general-purpose policies that can be easily applied zero-shot or efficiently finetuned for downstream tasks. However, these policies can still have poor generalization, both due to the choice of modeling objective and from learning from a static dataset. In this work, we focus on the challenging task of zero-shot goal generalization, where a policy is evaluated on unseen tasks that require combining its existing knowledge (compositional generalization). An avenue for improving a policy's generalization is by generating new experience with world models; however, such generation has proven difficult for longer horizons. Thus, to alleviate this issue, we propose TD-Aug, which samples from a geometric horizon model and allows for directly imagining novel outcomes that can be achieved by composing existing knowledge. We demonstrate that training on these future outcomes as goals for goal-conditioned BC policies significantly improves generalization in stitching-based OGBench tasks.

1 INTRODUCTION

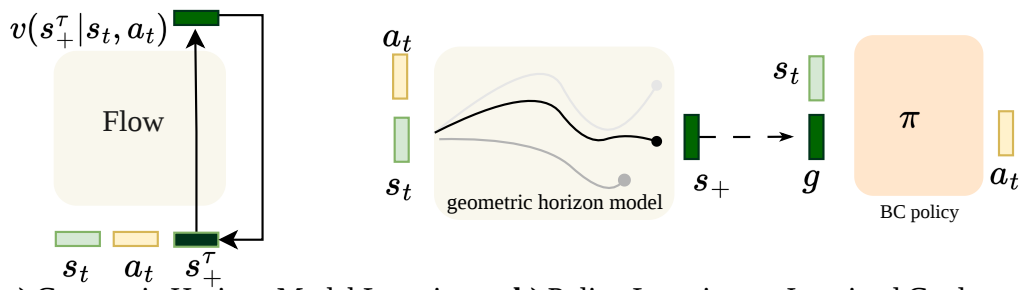
Training policies with goal-conditioned behavior cloning (BC) provides a simple strategy to learn from prior experience. However, generalization remains a challenge, limiting performance outside of the training data. Generally, world models (Ha & Schmidhuber, 2018) present an interesting strategy for improving policies from offline data (Hafner et al., 2025). These models can be used to generate synthetic experience (Sutton, 1991; Bruce et al., 2024), allowing policies to get more out of limited real data, which may be important in domains such as robotics. In this work, we consider generating experience using a type of world model to imagine counterfactual future outcomes to enable improved goal generalization.

We tackle a gap identified for BC policies showing poor performance on tasks involving stitching (Ghugare et al., 2024). In this setting, after policies are trained on a subset of tasks, they are then evaluated zero-shot on harder tasks involving composition. We illustrate an example in Figure 2, where a policy is trained on how to solve short horizon tasks (left), and then tested on a compositional task, such as going across the maze (middle). To solve this problem, we propose to learn a generative model that predicts what states will be encountered after mixing behaviors of the independent data collecting policies (right). Specifically, the generative model we consider is a geometric horizon model (GHM) (Janner et al., 2020; Thakoor et al., 2022) trained using Temporal Difference Flows (TD-Flows) (Farebrother et al., 2025). Then, using samples generated from our GHM as a data augmentation for goals, we teach our policy to perform compositional tasks. We provide an overview in Figure 1, and refer to our method as **TD-Aug**. We further justify this procedure by showing that augmenting BC with goals from the successor measure relates to value weighted regression.

To implement TD-Aug, we study several GHM objectives based on both temporal difference (TD) (Farebrother et al., 2025; 2026) and geometric Monte Carlo (MC) (Eysenbach et al., 2022) sampling, demonstrating that TD methods are crucial for generalization. Using horizon consistency (Farebrother et al., 2026), we propose a new TD-Flow objective which combines both MC and TD sampling during training, leading to improved long-horizon generation.

Empirically, in Section 6.1, we first validate TD-Aug on low-dimensional navigation tasks in OGBench (Park et al., 2025a) that require stitching for success, substantially improving behavior cloning generalization, from 31% to 79% average success. Then, we demonstrate that our approach can also

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.



a) Geometric Horizon Model Learning **b)** Policy Learning on Imagined Goals

Figure 1: **Overview of TD-Aug.** We **a)** train a GHM on an offline dataset to approximate samples from the successor measure of the behavior mixture policy. Then, **b)** we train BC policies on a mix of ground-truth and generated goals, *improving zero-shot generalization on compositional tasks.*

improve generalization on high-dimensional, image-based tasks (29% to 56%) in Section 6.2 and complicated manipulation tasks (13% to 52%) in Section 6.3.

2 Related Work

Data augmentation with temporal structure has been an effective way to re-use experience in reinforcement learning. Models can generate data, for example by sampling transitions for model-free approaches (Sutton, 1991; Ha & Schmidhuber, 2018; Feinberg et al., 2018; Hafner et al., 2020; Lu et al., 2023; Zhou et al., 2024). In the goal-conditioned setting, Hindsight Experience Replay (HER) (Andrychowicz et al., 2017) allows re-using transitions with different achieved outcomes as goals (Kaelbling, 1993). When training policies with BC, sampling hindsight goals is beneficial, but does not cover *compositional generalization* (Ghugare et al., 2024), where a policy is evaluated on state-goal pairs not seen together in the same training trajectories. To mitigate this gap, states are stitched across trajectories using clustering, but this only covers 1-step stitches and relies on heuristics, such as a good distance metric between states. In contrast, we use generative models to capture a future state distribution that directly stitches across trajectories.

Algorithm 1 TD-Aug

Representation Learning (Optional)
Use either states s, or learned representations $z = \phi(s)$, as the space for GHM and policy (Section 6.2).

Geometric Horizon Model Training
Train M_ϕ^β to approximate the successor measure of the behavior mixture policy, using $\mathcal{L}_{\text{MCHC-}\beta}$ (Equation 9).

Policy Training
Train π_θ by BC on a mixture of generated goals from M_ϕ^β with proportion α, as well as real samples (Equation 11).

Long-Horizon Prediction: In the context of predicting future states, instead of generating states sequentially, prediction can be made more accurate by sampling states in parallel, i.e. jointly sampling sub-trajectories (Janner et al., 2022), which could be chained together (Luo et al., 2025; Lee & Choi, 2025), for longer-horizon or compositional predictions. In comparison, we avoid having to denoise intermediate states or utilize heuristic strategies to chain sub-trajectories together. This is achieved while still generating long-horizon samples in $O(1)$ steps through amortization during GHM training.

Representation learning can be employed to improve *compositional generalization* of BC by encoding temporal dependencies in the representations of a policy, such as with objectives that relate to approximating successor measures or temporal distance (Myers et al., 2025; Lawson et al., 2025; Park et al., 2026b). This has the advantage of avoiding state prediction, but still has challenges, such as with stability and long-horizon generalization. In practice, generative modeling and representation learning can be used together, which is useful for predicting high-dimensional states.

Geometric horizon models (GHMs), also known as γ-models (Janner et al., 2020), propose a generative variant of temporal difference learning, allowing for approximate sampling from the *successor measure* (Thakoor et al., 2022; Tomar et al., 2024; Wiltzer et al., 2024; Farebrother et al., 2025), a continuous generalization of *successor representations* (Dayan, 1993) of a policy.

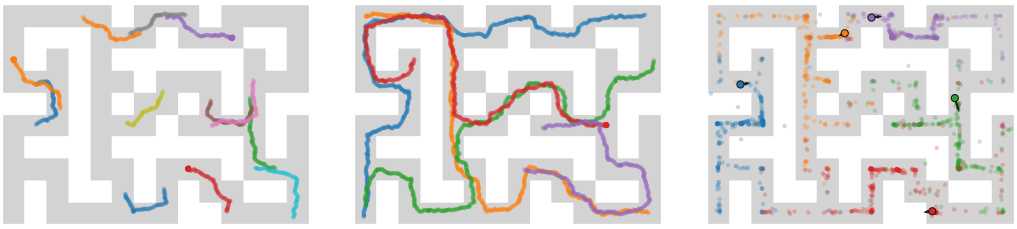


Figure 2: **Ground-truth and generated samples. Left**, on `pointmaze-giant-stitch`, we display ground truth trajectories from the dataset. **Middle**, we display long-horizon evaluation tasks. **Right**, we generate samples from an MCHC TD-Flow, as in Section 4, conditioned on the initial (outlined) state and action. These samples correspond to reachable future states, including those that can only be reached through stitching several trajectories with longer total horizon than short training trajectories. We provide visualizations across more environments in Figure 8.

Specifically, we consider Temporal Difference Flows (Farebrother et al., 2025) (TD-Flows), which use flow-matching objectives for learning such models. Most recent work with TD-Flows (Farebrother et al., 2025; 2026) has considered using GHMs for decision-time planning. GHMs can also be used for policy improvement (Zheng et al., 2026), or model-based value expansion (Feinberg et al., 2018), where future synthetic states can be used to estimate bootstrap targets for value estimation (Janner et al., 2020). Recent work has used MVE for offline RL using GHMs and Universal Horizon Models (UHMs) (Chung et al., 2026), but does not consider the goal-conditioned setting. In contrast, we particularly study GHMs for goal augmentation, and how this alone can induce better zero-shot goal generalization after offline training.

3 Background

3.1 Goal-Conditioned Reinforcement Learning

We study offline learning in goal-conditioned Markov Decision Processes (MDPs), defined as the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{G}, P, r, \gamma, \mu)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{G}$ is the goal space, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition kernel, r is the reward function, $\gamma \in [0, 1)$ is the discount, and $\mu \in \Delta(\mathcal{S})$ is an initial state distribution. We consider that goals coincide with states ($\mathcal{G} = \mathcal{S}$), and the reward is the Dirac delta at the goal: $r_g(s_t, a_t, s_{t+1}) = \delta_g(s_{t+1})$. We aim to learn a policy $\pi : \mathcal{S} \times \mathcal{G} \to \Delta(\mathcal{A})$ using a dataset $\mathcal{D}$, *without online interaction*, to maximize expected return:

$$J(\pi) = \mathbb{E}_{g\sim p(g), s_0\sim\mu, a_t\sim\pi(\cdot|s_t,g), s_{t+1}\sim P(\cdot|s_t,a_t)}\left[\sum_{t=0}^{\infty} \gamma^t r_g(s_t, a_t, s_{t+1})\right]$$

For the subset $X \subseteq \mathcal{S}$, we define the *normalized successor measure* of a policy π as:

$$M^\pi_\gamma(X \mid s, a) \triangleq (1-\gamma)\sum_{k\geq 0} \gamma^k \Pr(s_{k+1} \in X \mid s_0 = s, a_0 = a, \pi) \tag{1}$$

where $\Pr$ denotes the time-specific probability that we are in X starting from (s, a) when following π. We write the density of the successor measure m^π_γ via $M^\pi_\gamma(\mathrm{d}s_+ \mid s, a) = m^\pi_\gamma(s_+ \mid s, a)\,\mathrm{d}s_+$, and the action-marginalized successor measure is defined as $M^\pi_\gamma(\cdot \mid s) \triangleq \mathbb{E}_{a\sim\pi(\cdot|s)}[M^\pi_\gamma(\cdot \mid s, a)]$. We also define the discounted state occupancy density as the marginal of m^π_γ over the initial state distribution: $d^\pi(s) \triangleq \mathbb{E}_{s_0\sim\mu, a_0\sim\pi(\cdot|s_0)}[m^\pi_\gamma(s \mid s_0, a_0)]$. With the goal reaching reward, the successor measure's density relates to the *goal-conditioned value function* via $Q^\pi_\gamma(s, a, g) = (1-\gamma)^{-1} m^\pi_\gamma(g \mid s, a)$.

3.2 Flow Matching

Following success in modeling policies and geometric horizon models (Farebrother et al., 2025), we utilize *flow matching* (Lipman et al., 2023). With a *source* distribution $p_0 \in \Delta(\mathbb{R}^d)$ and a *target*

distribution $p_1 \in \Delta(\mathbb{R}^d)$, flow matching learns a velocity field $v^\tau : \mathbb{R}^d \to \mathbb{R}^d$ such that for $x_0 \sim p_0$, the sample $x_1 = x_0 + \int_0^1 v^\tau(x_\tau)\mathrm{d}\tau$ matches the target distribution ($x_1 \sim p_1$).

In practice, we use *conditional* flow matching (Lipman et al., 2023):

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{x_1\sim\mathcal{D},\tau\sim U(0,1),x_\tau\sim p_{\tau|1}(\cdot|x_1)}\left[\|v_\theta^\tau(x_\tau) - u_\tau(x_\tau \mid x_1)\|_2^2\right].$$

We use the *conditional optimal transport* (CondOT) coupling, drawing a straight line between x_0 and x_1, $x_\tau = (1-\tau)x_0 + \tau x_1$. This gives the true conditional velocity field target as $u_\tau(x_\tau \mid x_1) = x_1 - x_0$. Our source distribution p_0 is the standard Gaussian, $\mathcal{N}(0, I)$.

3.3 Training Policies

We train policies on datasets $\mathcal{D} = \{(s_0^i, a_0^i, \cdots, s_T^i, a_T^i)\}_{i=1}^N$ comprised of trajectories collected by a set of unknown policies $\{\beta_j(a|s)\}$. These policies can be marginalized by a mixture policy $\beta(a \mid s) = \sum_j \beta_j(a \mid s)p(\beta_j \mid s)$, which is a Markovian policy that has occupancy equal to the expectation under the individual policies: $d^\beta(s) = \mathbb{E}_{p(\beta_j)}\left[d^{\beta_j}(s)\right]$ (Ghugare et al., 2024). In general, there is not equality in the conditional occupancy (the SM), $M^\beta(\cdot|s,a) \neq \mathbb{E}_{\beta_j\sim p(\beta_j|s,a)}\left[M^{\beta_j}(\cdot|s,a)\right]$.

We consider the standard *goal-conditioned BC* objective, which samples a current state from the dataset (d^β), and trains the policy to reach goals sampled as future states in the same trajectory, corresponding to samples from the successor measure of an individual policy M^{β_j}:

$$\mathcal{L}_{\text{BC}}(\theta) = -\mathbb{E}_{\substack{s\sim d^\beta, a\sim\beta(\cdot \mid s)\\ \beta_j\sim p(\beta_j|s,a),\ s_+\sim M^{\beta_j}(\cdot|s,a)}}[\log \pi_\theta(a|s, g = s_+)] \tag{2}$$

We perform goal-conditioned behavior cloning using flow matching, representing the policy by a conditional velocity field $v_\theta^\tau(a_\tau \mid s, g)$:

$$\mathcal{L}_{\text{BC}}(\theta) = \mathbb{E}_{(s,a,s_+),\tau\sim U(0,1),a_0\sim p_0,a_\tau=(1-\tau)a_0+\tau a}\left[\|v_\theta^\tau(a_\tau \mid s, g = s_+) - (a - a_0)\|_2^2\right] \tag{3}$$

To sample from $\pi_\theta(\cdot|s,g)$, using $a_0 \sim p_0$, we integrate with v_θ^τ, via $a_1 = a_0 + \int_0^1 v_\theta^\tau(a_\tau|s,g)\mathrm{d}\tau$.

3.4 Temporal Difference Flows

Recently, there has been significant interest in developing generative models of the successor measure, called geometric horizon models (GHMs), that can sample states from the *geometric horizon* of any given policy (Janner et al., 2020; Thakoor et al., 2022). These models can be trained via TD learning, as for all (s, a), the successor measure M_γ^π satisfies:

$$M_\gamma^\pi(\cdot|s,a) = (1-\gamma)P(\cdot|s,a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a),a'\sim\pi(\cdot|s')}[M_\gamma^\pi(\cdot|s',a')] \tag{4}$$

Optimizing this objective is challenging with prior generative modeling techniques such as GANs (Goodfellow et al., 2014) or normalizing flows (Rezende & Mohamed, 2015), especially at large discounts γ. This led to the proposal to use flow matching to learn GHMs (Farebrother et al., 2025), which showed strong success in simulated control domains. TD-Flows parameterize the GHM as a velocity field $v_\phi^\tau : \mathbb{R}^d \times \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ with parameters ϕ, such that with source $x_0 \sim p_0 \in \Delta(\mathbb{R}^d)$, the velocity field induces the flow $\psi^\tau : \mathbb{R}^d \times \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$:

$$\psi_\phi^\tau(x_0|s,a) = x_0 + \int_0^\tau v_\phi^i(\psi_\phi^i(x_0|s,a)|s,a)\mathrm{d}i. \tag{5}$$

We aim to learn v_ϕ^τ, such that for $x_0 \sim p_0$, the samples $\psi_\phi^1(x_0|s,a)$ are distributed according to $M_\gamma^\pi(\cdot|s,a)$. Thus, we will use the notation $M_\phi^\pi(\cdot|s,a)$ for our approximate successor measure parameterized by v_ϕ^τ. Exploiting the Bellman structure of the successor measure, Farebrother et al. (2025) show that a flow that models said measure can be learned by minimizing the TD^2-CFM loss:

$$\mathcal{L}_{\text{TD}^2}(\phi;\bar{\phi}) = (1-\gamma)\mathcal{L}_{\text{td-ss}}(\phi) + \gamma\mathcal{L}_{\text{td-bs}}(\phi;\bar{\phi}), \text{ where:} \tag{6}$$

$$\mathcal{L}_{\text{td-ss}}(\phi) = \mathbb{E}_{\tau\sim U(0,1),(s,a,s')\sim\mathcal{D}^\beta,x_0\sim p_0,x_\tau=(1-\tau)x_0+\tau s'}\left[\|v_\phi^\tau(x_\tau|s,a) - (s' - x_0)\|^2\right]$$

$$\mathcal{L}_{\text{td-bs}}(\phi) = \mathbb{E}_{\tau\sim U(0,1),(s,a,s')\sim\mathcal{D}^\beta,a'\sim\pi(\cdot|s'),x_0\sim p_0,x_\tau=\psi_{\bar{\phi}}^\tau(x_0|s',a')}\left[\|v_\phi^\tau(x_\tau|s,a) - v_{\bar{\phi}}^\tau(x_\tau|s',a')\|^2\right]$$

where $\bar{\phi}$ are target flow parameters, as commonly used in value-based RL (Mnih et al., 2015). Recalling equation 4, $\mathcal{L}_{\text{td-ss}}$ targets the single-step transition kernel, while $\mathcal{L}_{\text{td-bs}}$ bootstraps using the successor measure at the next state. Together, this leads to approximating M^{π}_{γ}.

We can also approximate the successor measure $M^{\beta}(\cdot|s,a)$ of the mixture policy β, rather than an arbitrary policy π. While this requires sampling $a' \sim \beta(\cdot|s')$ in the Bellman update, in practice, we will generally bootstrap with the next action in the dataset, yielding the term:

$$\mathcal{L}_{\text{td-bs}}(\phi) = \mathbb{E}_{\tau\sim U(0,1),(s,a,s',a')\sim\mathcal{D}^{\beta},x_0\sim p_0,x_\tau=\psi^{\tau}_{\bar{\phi}}(x_0|s',a')}\left[\|v^{\tau}_{\phi}(x_\tau|s,a) - v^{\tau}_{\bar{\phi}}(x_\tau|s',a')\|^2\right]$$

We refer to this objective as $\mathcal{L}_{\text{TD}^2\text{-}\beta}$. This is a simpler objective than the fully off-policy objective in equation 6, as we bootstrap over fully in-support actions. If one does not need to stitch across trajectories, we can utilize a simpler approach to model M^{β}_{γ} using empirical Monte Carlo samples of future states. This allows us to learn a generative model directly with supervised learning:

$$\mathcal{L}_{\text{MC}}(\phi) = \mathbb{E}_{\substack{(s,a)\sim\mathcal{D}^{\beta},k\sim\text{geom}(1-\gamma),s_+=s_{\to k}\\ \tau\sim U(0,1),x_0\sim p_0,x_\tau=(1-\tau)x_0+\tau s_+}}\left[\|v^{\tau}_{\phi}(x_\tau \mid s,a) - (s_+ - x_0)\|^2_2\right]. \tag{7}$$

Here, $s_{\to k}$ denotes the state that is k steps further from s along the sampled trajectory in the dataset. If we had infinitely long trajectories, collected directly by the mixture policy, this objective would approximate $M^{\beta}(\cdot|s,a)$. However, in practice, we have datasets with truncated horizon, and collected from a set of data-collecting policies. This leads $\mathcal{L}_{\text{MC}}(\phi)$ to instead approximate a mixture of successor measures $\mathbb{E}_{\beta_j\sim p(\beta_j|s,a)}M^{\beta_j}_{\gamma}(\cdot|s,a)$ (Lawson et al., 2025), preferring the use of $\mathcal{L}_{\text{TD}^2\text{-}\beta}$.

4 Improving Long-Horizon, On-Policy Generation by Combining MC and TD Sampling

In this section, we propose a variant of the TD-Flow training procedure ($\mathcal{L}_{\text{TD}^2\text{-}\beta}$) to address scaling to long horizon prediction, which is affected by the discount γ used during bootstrapping. Intuitively, fitting the GHM is easier at lower discounts, as the loss is more heavily weighted to the ground truth next-state prediction, rather than the bootstrap. Recent work proposes the Temporal Difference Horizon Consistency (TD-HC) objective (Farebrother et al., 2026), which proposes how to use a "local horizon" GHM (e.g. a GHM conditioned on a discount $\omega < \gamma$) to obtain an estimate of the successor measure to approximate a higher discount GHM (γ) (Janner et al., 2020; Farebrother et al., 2026), i.e. $\omega < \gamma$ through a Bellman-like relation:

$$\begin{aligned} M^{\pi}_{\gamma}(\cdot|s,a) \triangleq &(1-\gamma)P(\cdot|s,a) + \gamma\frac{1-\gamma}{1-\omega}\mathbb{E}_{s'\sim P(\cdot|s,a),a'\sim\pi(\cdot|s')}[M^{\pi}_{\omega}(\cdot|s',a')] \\ &+\gamma\frac{\gamma-\omega}{1-\omega}\mathbb{E}_{s'\sim P(\cdot|s,a),a'\sim\pi(\cdot|s'),s_+\sim M^{\pi}_{\omega}(\cdot|s',a'),a_+\sim\pi(\cdot|s_+)}[M^{\pi}_{\gamma}(\cdot|s_+,a_+)] \end{aligned} \tag{8}$$

This can be converted to a TD-Flow loss, $\mathcal{L}_{\text{TDHC}}$, by modifying equation 6, shown in Appendix B.1.

We propose a variant of $\mathcal{L}_{\text{TDHC}}$ adapted for the on-policy setting to approximate M^{β}_{γ}. While equation 7 supervises the flow with ground-truth samples, which we will show leads to high-quality generations, it does not generate samples that can be reached through stitching trajectories. In contrast, equation 6 stitches, but uses sparse ground-truth signal, which decreases as the discount is increased. This may make it harder to have higher-quality samples and to scale to longer horizons. To obtain the best of both objectives, we consider a *horizon-consistent objective utilizing MC samples* from the dataset. While we will learn M^{β}_{γ}, we will not learn M^{β}_{ω}, and instead utilize MC samples from dataset trajectories. This leads to the objective:

$$\mathcal{L}_{\text{MCHC-}\beta}(\phi;\bar{\phi}) = (1-\gamma)\mathcal{L}_{\text{td-ss}}(\phi) + \gamma\frac{1-\gamma}{1-\omega}\mathcal{L}_{\text{mc}}(\phi) + \gamma\frac{\gamma-\omega}{1-\omega}\mathcal{L}_{\text{td-mc-bs}}(\phi;\bar{\phi}), \text{ where:} \tag{9}$$

$$\mathcal{L}_{\text{td-ss}}(\phi) = \mathbb{E}_{\tau\sim U(0,1),(s,a,s')\sim\mathcal{D}^{\beta},x_0\sim p_0,x_\tau=(1-\tau)x_0+\tau s'}\left[\|v^{\tau}_{\phi}(x_\tau|s,a) - (s'-x_0)\|^2\right]$$

$$\mathcal{L}_{\text{mc}}(\phi) = \mathbb{E}_{\substack{\tau\sim U(0,1),(s,a)\sim\mathcal{D}^{\beta},k\sim\text{geom}(1-\omega)\\ s_+=s_{\to k+1},x_0\sim p_0,x_\tau=(1-\tau)x_0+\tau s_+}}\left[\|v^{\tau}_{\phi}(x_\tau|s,a) - (s_+-x_0)\|^2\right]$$

$$\mathcal{L}_{\text{td-mc-bs}}(\phi) = \mathbb{E}_{\substack{\tau\sim U(0,1),(s,a)\sim\mathcal{D}^{\beta},k\sim\text{geom}(1-\omega)\\ s_+,a_+=s_{\to k+1},a_{\to k+1},x_0\sim p_0,x_\tau=\psi^{\tau}_{\bar{\phi}}(x_0|s_+,a_+)}}\left[\|v^{\tau}_{\phi}(x_\tau|s,a) - v^{\tau}_{\bar{\phi}}(x_\tau|s_+,a_+)\|^2\right]$$

This loss is not just supervised by the next ground truth states, but also future MC samples. Additionally, instead of bootstrapping from the next state s_{t+1}, we will bootstrap from this MC sample, which is obtained as a future state after s_{t+1}, i.e. $s_+ = s_{t+k+1}$, where $k \sim \text{geom}(1-\omega)$. As we utilize MC samples, which are both truncated, and are collected by a single policy, they do not correspond to the true M^β_ω. Thus, this objective is a biased approximation of M^β_γ as ω increases, but allows increasing γ while still training on higher proportions of ground-truth data.

5 Imagining Goals for Policy Generalization

5.1 Generalization Problem

We consider a generalization problem for BC policies. Following the setting from Section 3.3, we train on trajectories collected by unknown policies $\{\beta_j(a|s)\}$, with mixture policy $\beta(a|s)$. We will now define training and test distributions for π_θ equation 2, following Ghugare et al. (2024):

$$R^\pi_{\text{id}} = \underbrace{\mathbb{E}_{s\sim d^\beta, \beta_j\sim p(\beta_j|s), s_g\sim M^{\beta_j}(\cdot|s)}\left[R^{\pi_\theta}(s, s_g)\right]}_{\text{in-distribution training tasks}}, R^\pi_{\text{gen}} = \underbrace{\mathbb{E}_{s\sim d^\beta, s_g\sim M^{\beta}(\cdot|s)}\left[R^{\pi_\theta}(s, s_g)\right]}_{\text{tasks requiring compositional generalization}} \tag{10}$$

With $R^\pi_\theta(s, s_g)$ defining the success rate of the policy to reach s_g from s, R^π_{id} averages performance over in-distribution tasks and R^π_{gen} describes average success on a class of out-of-distribution tasks requiring compositional generalization. These include tasks which may not be completed by a single data collecting policy β_j, but still have sufficient data coverage to complete through composing or stitching trajectories collected by different policies, i.e. reachable with β.

While equation 2 corresponds to improving on R^π_{id}, the BC objective does not cover R^π_{gen}, leading to poor empirical performance. We can see that if we had access to goals coming from the successor measure of the mixture policy β during training, rather than that of individual policies, our policy could learn to reach goals required for better R^π_{gen}.

5.2 Method

To address this type of policy generalization, we can directly approximate and sample from the target generalization distribution. We note that because we do not have full rollouts from the mixture policy, we must utilize TD learning to stitch, relying on $\mathcal{L}_{\text{TD}^2\text{-}\beta}$ or $\mathcal{L}_{\text{MCHC-}\beta}$, rather than $\mathcal{L}_{\text{MC}}(\phi)$. Thus, we train **on-policy TD-Flows**, which will yield an approximation $M^\beta_\phi(\cdot|s,a)$ for the successor measure of the mixture policy. After obtaining M^β_ϕ, we perform **goal augmentation** where we train on a mix of original goals plus samples from the generative model, covering the compositional distribution:

$$\max_{\pi_\theta} \mathbb{E}_{s\sim d^\beta, a\sim\beta(a|s),\ s_+\sim P_{\text{aug}}(\cdot|s,a)}\left[\log \pi_\theta(a|s, g = s_+)\right] \tag{11}$$

$$\text{where } P_{\text{aug}}(\cdot|s,a) \triangleq \underbrace{\alpha M^\beta_\phi(\cdot|s,a)}_{\text{imagined compositional samples}} + \underbrace{(1-\alpha)\mathbb{E}_{p(\beta_j|s,a)}M^{\beta_j}(\cdot|s,a)}_{\text{ground truth future states}} \tag{12}$$

We generally refer our goal augmentation procedure as **TD-Aug**, and by default use $\mathcal{L}_{\text{MCHC-}\beta}$. We can also instantiate our method with $\mathcal{L}_{\text{TD}^2\text{-}\beta}$, which we denote later with **TD-Aug**$_2$. We provide an overview of our method in Algorithm 1 and Figure 1. An advantage of this procedure is that we are not limited to a fixed number of stitches, or heuristic sampling strategies (Ghugare et al., 2024). We also perform $O(1)$ sampling steps per sample to cover an expected horizon $H = (1-\gamma)^{-1}$, while one-step models need $O(H)$ steps, potentially resulting in compounding error and longer time for rollouts.

We relate this procedure to offline goal-conditioned RL, specifically 1-step policy improvement over the mixture policy with value-weighted regression (Peters & Schaal, 2007; Wang et al., 2018; Peng et al., 2019; Wang et al., 2020), for a brief proof, see Appendix C:

Theorem 1 (Equivalence to 1-step improvement)**.** *With $\alpha = 1$, and exact samples from $M^\beta(\cdot|s,a)$, our augmentation strategy in equation 11 is equivalent to value-weighted regression used for one-step improvement over behavior policy β:*

$$\theta^* \in \arg\max_\theta \mathbb{E}_{\substack{s\sim d^\beta, s_+\sim d^\beta \\ a\sim\beta(\cdot|s)}}\left[\frac{Q^\beta(s,a,g=s_+)}{d^\beta(s_+)}\log \pi_\theta(a \mid s, g = s_+)\right] \tag{13}$$

6 EXPERIMENTS

In this section, we show that **TD-Aug** can improve combinatorial generalization across a variety of `stitch` tasks in the OGBench (Park et al., 2025a) benchmark suite. In Section 6.1, we (a) evaluate the quality of generated samples as horizon scales for different GHM objectives on low-dimensional navigation tasks, and (b) show that TD-Aug is useful for improving BC generalization, even on long-horizon tasks requiring many stitches. Then, (c) we show that our method can scale to visual environments by learning GHMs in latent space in Section 6.2, and Section 6.3, we demonstrate that we can also improve stitching on manipulation tasks.

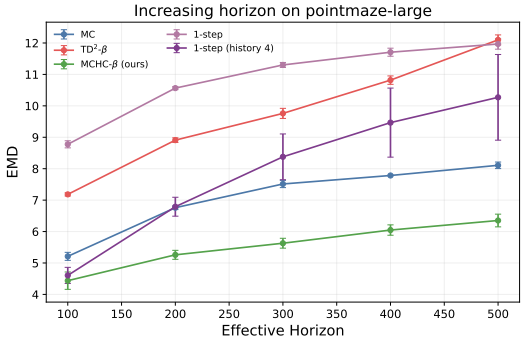


Figure 3: **Evaluating sampling with increasing horizon**. We compute earth mover's distance (EMD) between empirical samples and generated samples on held-out evaluation data for `pointmaze-large-stitch`.

6.1 STATE-BASED ENVIRONMENTS

We first evaluate our method on `pointmaze-stitch` navigation tasks from OGBench, which allow us to isolate the challenge of long-horizon stitching from high-dimensional dynamics. In these tasks, agents are trained on short trajectories, which travel up to 4 maze cells. Then, agents are evaluated on longer compositional tasks such as those that span the whole width of the maze. We find that the original stitching datasets, especially for point, are not terminated upon goal-reaching, resulting in many transitions that idle at the goal. Thus, we re-collect the stitch datasets that terminate upon reaching a sub-goal. On new datasets, baselines perform similarly, or slightly better than before.

To begin our study, we first **compare the quality of generative models**. *Qualitatively*, we provide visualizations on `pointmaze-{medium,large,teleport,giant}-stitch` in Figure 8. We can see that MC Flows can accurately capture training trajectories and multi-modality in the dataset. We can see this captures both multi-modality in the policy, and environment stochasticity, as shown in `teleport`, where upon entering a black hole, the agent is randomly sent to a random white hole. However, the MC Flow does not implicitly stitch across trajectories. On the other hand, the TD-based objectives (TD^2-β, MCHC-β), can generate samples that require stitching, capturing states reachable beyond than the training trajectory horizon. However, the TD^2-β samples are more diffuse, or may go backwards from the direction implied by the ground truth samples. In contrast, the MCHC-β samples balance implicit stitching with the quality of MC samples.

Next, we perform a *quantitative* comparison between model samples on a held-out evaluation dataset on `pointmaze-large-stitch`. From an initial state, we can see how models (trained on short trajectories) generate samples that are closer in distribution to longer, held-out trajectories, measuring stitching. This is performed by sampling initial (state,action) tuples from a held-out dataset. Then, from these tuples we sample future states geometrically, obtaining empirical samples. From these same initial states, we sample from our model, as shown in Figure 7. Then, we compute the earth mover's distance (EMD) between these two sets of points, and then average across different initial positions. We repeat this for models trained at various effective horizons $H = (1-\gamma)^{-1}$, measuring against ground truth samples at the corresponding horizon, as shown in Figure 3. In addition to the GHM based models, we also generate samples from autoregressive one-step prediction, which we further discuss in Appendix B.3. While some models have similar EMD at the initial horizon (100), we can see that the errors of TD^2-β, and 1-step baselines, quickly increases with horizon. Additionally, the EMD of MC increases because it is trained on short trajectories, and thus has no mechanism to stitch. In contrast, MCHC-β has the lowest EMD as horizon increases, showing the benefit of mixing both MC targets and TD when training on-policy GHMs.

Next, we study how geometric horizon models can **empirically** improve a policy's **combinatorial generalization** capabilities in Table 1. Without any augmentation, a goal-conditioned flow-based behavior cloning (GCFBC) baseline (Park et al., 2025b) obtains moderate zero-shot generalization in smaller mazes, but struggles in larger mazes as the horizon is increased. To improve generalization, we apply our generative goal augmentation procedure, **TD-Aug**. In addition to TD-Aug using $\mathcal{L}_{\text{MCHC-}\beta}$, we compare to TD-Aug$_2$ using $\mathcal{L}_{\text{TD}^2\text{-}\beta}$, as well as MC-Aug using $\mathcal{L}_{\text{MC}}$. We compare to

Table 1: **Generalization through compositional data augmentation.** With goal conditioned flow behavior cloning (GCFBC), we compare training on the original dataset to various data augmentation strategies. We calculate success rate averaged over the last 3 checkpoints, and then report the mean, std over 3 seeds. We **bold** the top performing methods.

pointmaze	GCFBC	+Cluster-Aug	+Dyn-Aug	+MC-Aug	+TD-Aug$_2$	+TD-Aug	GCIQL	QRL	CRL	HIQL
medium-stitch	55 ± 10	89 ± 5	94 ± 6	70 ± 2	96 ± 1	**99** ± 0	74 ± 1	88 ± 5	4 ± 2	71 ± 8
large-stitch	26 ± 6	62 ± 8	51 ± 10	28 ± 17	40 ± 3	**97** ± 3	40 ± 1	89 ± 15	0 ± 0	18 ± 4
giant-stitch	0 ± 0	3 ± 6	0 ± 0	0 ± 0	0 ± 0	**54** ± 17	0 ± 0	**54** ± 9	0 ± 0	0 ± 0
teleport-stitch	44 ± 7	47 ± 2	40 ± 7	48 ± 3	45 ± 4	**66** ± 3	27 ± 1	2 ± 4	10 ± 5	30 ± 3
average	31	50	46	37	45	**79**	35	58	4	30

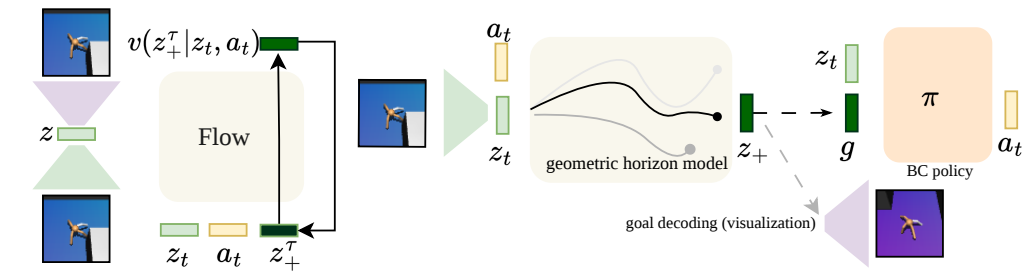
clustering-based goal augmentation (Cluster-Aug) (Ghugare et al., 2024), which can create stitches between two trajectories, and against autoregressive 1-step prediction with history to sample from the SM (Dyn-Aug), as utilized in Figure 3. For these methods, we list additional training details in Appendix A, including hyperparameters for BC in Table 5, and for TD-Flows in Table 4. Although the task horizon varies, we use the same hyperparameters for each environment in `pointmaze` for TD-Aug. This includes the effective horizon, which we set to 500 ($\gamma = 0.998$), while for MC-Aug and TD-Aug$_2$ we use 100 ($\gamma = 0.99$), as they perform worse at higher horizons. After training GHMs, all policies are trained for 10^6 steps, and each method is trained with batches using goals that are of a proportion $1 - \alpha$ of future states in the same trajectory (i.e. hindsight goal sampling), and with proportion α with generated goals. The BC baseline uses $\alpha = 0$, and augmentation uses $\alpha = 0.5$. We can see that TD-Aug significantly improves generalization, including on long-horizon tasks, such as `pointmaze-giant-stitch`. In contrast, MC-Aug maintains close to original performance, demonstrating that generating additional synthetic short-horizon samples does not improve generalization. We also see that with TD-Aug, GCFBC outperforms prior offline GCRL methods in `pointmaze`, including GCIQL (Kostrikov et al., 2022), QRL (Wang et al., 2023), Contrastive RL (CRL) (Eysenbach et al., 2022), and HIQL (Park et al., 2023). Observing the weaker performance of CRL on `stitch` tasks, which also uses MC samples like GCBC, we show contrastive learning on generated samples leads to improved performance in Appendix E.

Table 2: **Visual generalization through compositional data augmentation.** With autoencoder (AE) GCFBC, we compare training on the original visual datasets to various data augmentation strategies. We calculate success rate averaged over the last 3 checkpoints, and then report the mean, std over 3 seeds. We **bold** the highest performing method overall, and highlight in **brown** the top BC method.

visual-antmaze	AE-GCFBC	+MC-Aug	+TD-Aug$_2$	+TD-Aug	GCBC	GCIQL	QRL	CRL	HIQL
medium-stitch	45 ± 4	60 ± 2	56 ± 1	**84** ± 7	69 ± 4	9 ± 0	0 ± 0	82 ± 2	**92** ± 1
large-stitch	12 ± 1	19 ± 1	9 ± 0	**28** ± 1	21 ± 6	1 ± 0	1 ± 0	29 ± 6	**46** ± 2
average	29	40	33	**56**	45	5	1	56	**69**

6.2 Composition in Visual Navigation

In this section, we extend **TD-Aug** to *visual environments* with `visual-antmaze-stitch`. Instead of learning the GHM in pixel space, we first train an autoencoder (AE), obtaining a low-dimensional latent space $\mathcal{Z}$. We provide details for training the AE in Table 6. Then, we utilize the



a) Train AE **b)** Latent GHM Learning **c)** Policy Learning on Imagined Goals in Latent Space

Figure 4: **Extension of TD-Aug to visual environments.**

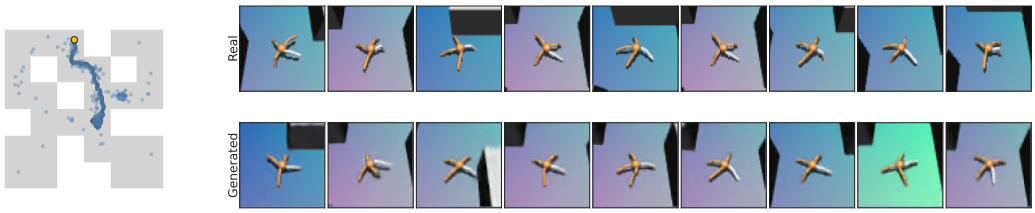


Figure 5: **Samples for `visual-antmaze-medium`.** We visualize samples generated with MCHC-β at $\gamma = 0.995$. We display both real, geometrically sampled, states, as well as reconstructions from the GHM's predicted latents using the AE decoder.

same architectures for the GHM, and policy, as performed in the non-visual setting (Section 6.1), but now operate over latent space, as shown in Figure 4. As a baseline, we report both standard BC in OGBench (**GCBC**), using early state-goal fusion, and an autoencoder BC baseline, labeled **AE-GCFBC**. The standard baseline concatenates state and goal images, and passes them through a small IMPALA-based (Espeholt et al., 2018) CNN. In contrast, the autoencoder BC baseline separately encodes goals and states into $\mathcal{Z}$, and then concatenates them to predict actions. This architecture is preferable with goal augmentation, as we generate goals in latent space (Dao et al., 2023) and feed them to the policy without decoding. With this setup, in Table 2, we report adding augmentation to AE-GCFBC. We can see that without augmentation, this architecture under-performs GCBC. While we improve with MC-Aug, we can see that $\mathcal{L}_{\text{TD}^2\text{-}\beta}$, used in TD-Aug$_2$ leads to less performance, which is in contrast to the non-visual setting. We attribute this to the difficulty of bootstrapping in higher-dimensional approximate ($\mathcal{Z}$) space. Thus, we find the combination of MC and TD to be the most effective, where TD-Aug improves AE-GCFBC and outperforms the GCBC baseline.

Table 3: **Manipulation with compositional data augmentation.** We evaluate GCFBC with action chunking on OGBench manipulation tasks, using the `oracle` goal representation. We use the original `play` dataset, and then introduce a new `stitch` dataset, consisting of only short trajectories. Then, we add TD-Aug to close the generalization gap from `stitch`. We calculate average success on the last 3 checkpoints, and report the mean, std over 3 seeds. We **highlight** the top method on `stitch`.

dataset	GCFBC		+TD-Aug$_2$	+TD-Aug	GCIQL	HIQL
	`play`	`stitch`	`stitch`	`stitch`	`stitch`	`stitch`
`cube-double`	75 ± 1	20 ± 0	78 ± 2	**84 ± 0**	63 ± 1	6 ± 2
`cube-triple`	35 ± 5	20 ± 0	**36 ± 5**	31 ± 2	8 ± 2	0 ± 0
`puzzle-4x4`	23 ± 2	0 ± 0	18 ± 0	**40 ± 2**	16 ± 5	0 ± 0
average	44	13	44	**52**	29	2

6.3 Composition in Manipulation

In this section, we evaluate TD-Aug on OGBench manipulation tasks requiring stitching, which is a challenging setting for modeling the successor measure. We first consider `cube-{double,triple}-play`, `puzzle-4x4-play` tasks with `oracle` goal representation (Park et al., 2025a; 2026a). In addition to the original `play` datasets, we construct a new challenging `stitch` dataset, where each trajectory contains one atomic task (i.e. pick-and-place, or button press), but the policy is evaluated on the original, long-horizon, multi-step tasks. While GCFBC using action chunking, $\pi(a_{t:t+h}|s, g)$, obtains good average success (44%) on the `play` tasks, it struggles when training on the `stitch` (13%). Thus, we apply TD-Aug to close the gap. We also train GHMs, with action chunking, $M^{\beta}_{\phi}(\cdot|s, a_{t:t+h})$, and predict h-sized jumps in the full state space, with additional details in Appendix B.2. On `stitch`, with TD-Aug, GCFBC's performance substantially increases from $13 \rightarrow 52$. We provide visualizations of the GHM's samples in Figure 9.

7 Conclusion

In this work, we have shown that TD-based geometric horizon models allow for enhanced generalization for behavior cloning through goal augmentation. There are a variety of directions that could build on this work. For example, behavior cloning is a strong pre-training algorithm that has been shown to scale well in real-world robotics settings (O'Neill et al., 2024; Black et al., 2024; Intelligence et al., 2025). Therefore, a key open question is how to efficiently and stably scale TD-based GHM training to high-dimensional, partially-observable robotics tasks.

AI USE STATEMENT

In this work, we used generative AI tools for coding, and to provide feedback on the paper drafts. We have not used generative AI tools for generating synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, propose or refine hypotheses, assist with translation, clean and reformat datasets, support qualitative and thematic data analysis, or interpret results. Additionally, we used generative AI tools to create or modify scientific figures or images, and helping with LaTeX editing. We have reviewed the AI-assisted work.

REPRODUCIBILITY STATEMENT

In the main text and Appendix, we provide details for reproducing our work. This includes architecture details, and hyperparameters in Appendix A, additional algorithm details in Appendix B, proof in Appendix C for Theorem 1, and visualizations for dataset structure in Appendix D. Upon the publication of our work, we will open-source our code and datasets to reproduce our experiments.

REFERENCES

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

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024. URL https://arxiv.org/abs/2410.24164.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

Hojun Chung, Junseo Lee, and Songhwai Oh. Offline reinforcement learning with universal horizon models, 2026. URL https://arxiv.org/abs/2605.15603.

Quan Dao, Hao Phung, Binh Nguyen, and Anh Tran. Flow matching in latent space. *arXiv preprint arXiv:2307.08698*, 2023.

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

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pp. 1407–1416. PMLR, 2018.

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, Remi Munos, Alessandro Lazaric, and Ahmed Touati. Temporal difference flows. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=j6H7c3aQyb.

Jesse Farebrother, Matteo Pirotta, Andrea Tirinzoni, Marc G. Bellemare, Alessandro Lazaric, and Ahmed Touati. Compositional planning with jumpy world models, 2026. URL https://arxiv.org/abs/2602.19634.

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.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Raj Ghugare, Matthieu Geist, Glen Berseth, and Benjamin Eysenbach. Closing the gap between TD learning and supervised learning - a generalisation point of view. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2401.11237.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/f033ed80deb0234979a61f95710dbe25-Paper.pdf.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination, 2020. URL https://arxiv.org/abs/1912.01603.

Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527*, 2025.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alex Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a vla that learns from experience, 2025. URL https://arxiv.org/abs/2511.14759.

Michael Janner, Igor Mordatch, and Sergey Levine. γ-models: Generative temporal difference learning for infinite-horizon prediction. In *Advances in Neural Information Processing Systems*, 2020.

Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis, 2022. URL https://arxiv.org/abs/2205.09991.

Leslie Pack Kaelbling. Learning to achieve goals. In *International Joint Conference on Artificial Intelligence*, 1993.

Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *arXiv preprint arXiv:2004.13649*, 2020.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://arxiv.org/abs/2110.06169.

Daniel Lawson, Adriana Hugessen, Charlotte Cloutier, Glen Berseth, and Khimya Khetarpal. Self-predictive representations for combinatorial generalization in behavioral cloning, 2025. URL https://arxiv.org/abs/2506.10137.

Kyowoon Lee and Jaesik Choi. State-covering trajectory stitching for diffusion planners, 2025. URL https://arxiv.org/abs/2506.00895.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2): 129–137, 1982.

Cong Lu, Philip J. Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=6jNQ1AY1Uf.

Yunhao Luo, Utkarsh A. Mishra, Yilun Du, and Danfei Xu. Generative trajectory stitching through diffusion composition, 2025. URL https://arxiv.org/abs/2503.05153.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

Vivek Myers, Bill Chunyuan Zheng, Anca Dragan, Kuan Fang, and Sergey Levine. Temporal representation alignment: Successor features enable emergent compositionality in robot instruction following, 2025. URL https://arxiv.org/abs/2502.05454.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

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, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations (ICLR)*, 2025a. URL https://arxiv.org/abs/2410.20092.

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

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, 2026a.

Seohong Park, Deepinder Mann, and Sergey Levine. Dual goal representations, 2026b. URL https://arxiv.org/abs/2510.06714.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pp. 745–750, 2007.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1530–1538, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/rezende15.html.

Richard S Sutton. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin*, 2(4):160–163, 1991.

Shantanu Thakoor, Mark Rowland, Diana Borsa, Will Dabney, Remi Munos, and Andre Barreto. Generalised policy improvement with geometric policy composition. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 21272–21307. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/thakoor22a.html.

Manan Tomar, Philippe Hansen-Estruch, Philip Bachman, Alex Lamb, John Langford, Matthew E. Taylor, and Sergey Levine. Video occupancy models, 2024. URL https://arxiv.org/abs/2407.09533.

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.

Qing Wang, Jiechao Xiong, Lei Han, Han Liu, Tong Zhang, et al. Exponentially weighted imitation learning for batched historical data. *Advances in Neural Information Processing Systems*, 31, 2018.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*. PMLR, 2023. URL https://arxiv.org/abs/2304.01203.

Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33:7768–7778, 2020.

Harley Wiltzer, Jesse Farebrother, Arthur Gretton, Yunhao Tang, André Barreto, Will Dabney, Marc G. Bellemare, and Mark Rowland. A distributional analogue to the successor representation, 2024. URL https://arxiv.org/abs/2402.08530.

Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the aaai conference on artificial intelligence*, volume 35, pp. 10674–10681, 2021.

Chongyi Zheng, Seohong Park, Sergey Levine, and Benjamin Eysenbach. Intention-conditioned flow occupancy models, 2026. URL https://arxiv.org/abs/2506.08902.

Zhaoyi Zhou, Chuning Zhu, Runlong Zhou, Qiwen Cui, Abhishek Gupta, and Simon Shaolei Du. Free from bellman completeness: Trajectory stitching via model-based return-conditioned supervised learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.19308.

A TRAINING DETAILS

	Hyperparameter	
Flow Matching	ODE Solver	Midpoint
	Train ODE $\mathrm{d}\tau$	0.1
	Sample ODE $\mathrm{d}\tau$	0.1
Network (U-Net)	t-Positional Embedding Dim	256
	t-Positional Embedding MLP	$(256, 256)$
	Block Dimension	$(512, 512, 512)$
	Encoder MLP	$(512, 512, 512)$
	activation	Mish
Optimizer (AdamW)	AdamW β_1	0.9
	AdamW β_2	0.999
	AdamW ϵ	10^{-4}
	Learning Rate	10^{-4}
	Weight Decay	10^{-3}
Training	Target Network EMA	10^{-3}
	Gradient Steps	
	TD	
	State,Visual	3M
	Manipulation	1M
	MC, MCHC	1M
	Batch Size	1024
	Discount (γ)	
	MC, TD	0.99
	MCHC	
	State	0.998
	Visual	0.995
	Manipulation	0.99
	Discount (ω)	
	State,Visual	0.99
	Manipulation	0.98

Table 4: Hyperparameters for TD-Flow training, following similar architecture details to Farebrother et al. (2025).

Table 5: Hyperparameters for GCFBC training.

	Hyperparameter	
Flow Matching	ODE Solver	Euler
	Sample ODE $\mathrm{d}\tau$	0.1
Network (MLP)	Hidden Layers	$(1024, 1024, 1024, 1024)$
	activation	GELU
Optimizer (Adam)	Adam β_1	0.9
	Adam β_2	0.999
	Learning Rate	
	State, Visual	3×10^{-4}
	Manipulation	3×10^{-5}
	Weight Decay	0
Training	Gradient Steps	
	State	1M
	Manipulation	2M
	Visual	500k
	Batch Size	
	State	1024
	Manipulation	1024
	Visual	256
Augmentation (only)	Batch Proportion (α)	0.5

Table 6: Hyperparameters for visual autoencoder training. For our autoencoder, we use a similar architecture to (Kostrikov et al., 2020; Yarats et al., 2021). For the encoder, we utilize two convolutional layers, with ReLU activations after each layer. The output of the convolutional layers is flattened, and linearly mapped to the latent dimension $|z| = 50$, followed by LayerNorm and tanh activation. The decoder utilizes a similar architecture, utilizing deconvolution layers, with the same kernel size, followed by ReLU (except for the final layer).

	Hyperparameter	
Architecture	Convolution Layers	2
	Channels	32
	Kernel	(3×3)
	Activation	ReLU
	Latent Dimension, $\lvert z \rvert$	50
Optimizer (Adam)	Adam β_1	0.9
	Adam β_2	0.999
	Learning Rate	10^{-4}
	Weight Decay	0
Training	Gradient Steps	100k
	Batch Size	1024

Table 7: **Hyperparameters for clustering-based goal-augmentation.** We follow (Ghugare et al., 2024) and use k-means clustering (Lloyd, 1982), with k set as below.

Hyperparameter	
Clusters	
`medium`	40
`large`	40
`teleport`	40
`giant`	80

B Additional Objectives

B.1 TD-HC

For reference, we provide the original Temporal Difference Horizon Consistency (TDHC) objective (Farebrother et al., 2026), which we modify to use MC samples in Section 4:

$$\mathcal{L}_{\text{TDHC}}(\phi;\bar{\phi}) = (1-\gamma)\mathcal{L}_{\text{td-ss}}(\phi;\gamma) + \gamma\frac{1-\gamma}{1-\omega}\mathcal{L}_{\text{td-bs}}(\phi;\bar{\phi},\gamma,\omega) + \gamma\frac{\gamma-\omega}{1-\omega}\mathcal{L}_{\text{td-hc-bs}}(\phi;\bar{\phi},\gamma,\omega) \tag{14}$$

$$\mathcal{L}_{\text{td-ss}} = \mathbb{E}_{\substack{\tau\sim U(0,1),(s,a,s')\sim\mathcal{D}^\beta \\ x_0\sim p_0, x_\tau=(1-\tau)x_0+\tau s'}}\left[\|v_\phi^\tau(x_\tau|s,a,\gamma) - (s'-x_0)\|^2\right]$$

$$\mathcal{L}_{\text{td-bs}} = \mathbb{E}_{\substack{\tau\sim U(0,1),(s,a,s')\sim\mathcal{D}^\beta, a'\sim\pi(s') \\ x_0\sim p_0, x_\tau=\psi_{\bar{\phi}}^\tau(x_0|s',a',\omega)}}\left[\|v_\phi^\tau(x_\tau|s,a,\gamma) - v_{\bar{\phi}}^\tau(x_\tau|s',a',\omega)\|^2\right]$$

$$\mathcal{L}_{\text{td-hc-bs}} = \mathbb{E}_{\substack{\tau\sim U(0,1),(s,a,s')\sim\mathcal{D}^\beta, a'\sim\pi(s') \\ x_0^1,x_0^2\sim p_0, s_+=\psi_{\bar{\phi}}^1(x_0^1|s',a',\omega), a_+\sim\pi(s_+) \\ x_\tau=\psi_{\bar{\phi}}^\tau(x_0^2|s_+,a_+,\gamma)}}\left[\|v_\phi^\tau(x_\tau|s,a,\gamma) - v_{\bar{\phi}}^\tau(x_\tau|s_+,a_+,\gamma)\|^2\right]$$

With $\gamma=\omega$, the original TD^2 objective equation 6 is recovered. To train with the TDHC objective, the flow is trained with discount conditioning, and in each batch, the discount is sampled uniformly $\gamma\in[\gamma_{\min},\gamma_{\max}]$ and $\omega\in[\gamma_{\min},\gamma]$. Instead, the MCHC objective equation 9 we use fixes $\gamma=\gamma_{\max}$, and samples $(s_+,a_+)\sim D^\beta$, where $s_+=s_{t+k}$, $k\sim\text{geom}(1-\omega)$, replacing having to learn and sample with $v_\phi^\tau(\cdot|\cdot,\omega)$.

B.2 GHMs for Manipulation

For manipulation tasks, we modify both GCFBC and the GHM to utilize action chunks of size H as shown in equation 15. For the GHM we also predict future states starting from after the action chunk, at increments matching the chunk size. We also proportionally reduce γ,ω. For example, with $h=10$, we reduce γ from 0.99 to 0.9, to match the same expected horizon. Additionally, we bootstrap with samples from a learned prior, $\beta_\theta(\cdot|s_+)$, predicting the next action chunk, rather than using a_+. This is useful, as it can allow for bootstrapping from the end states in a trajectory, where an action chunk is not available.

$$\mathcal{L}_{\text{MCHC-}\beta}(\phi;\bar{\phi}) = (1-\gamma)\mathcal{L}_{\text{td-ss}}(\phi) + \gamma\frac{1-\gamma}{1-\omega}\mathcal{L}_{\text{mc}}(\phi) + \gamma\frac{\gamma-\omega}{1-\omega}\mathcal{L}_{\text{td-mc-bs}}(\phi;\bar{\phi}) \tag{15}$$

$$\mathcal{L}_{\text{td-ss}}(\phi) = \mathbb{E}_{\substack{\tau\sim U(0,1),(s_t,a_{t:t+h},s_{t+h})\sim\mathcal{D}^\beta \\ x_0\sim p_0, x_\tau=(1-\tau)x_0+\tau s_{t+h}}}\left[\|v_\phi^\tau(x_\tau|s,a_{t:t+h}) - (s_{t+h}-x_0)\|^2\right]$$

$$\mathcal{L}_{\text{mc}}(\phi) = \mathbb{E}_{\substack{\tau\sim U(0,1),(s_t,a_{t:t+h},s_+)\sim\mathcal{D}^\beta \\ x_0\sim p_0, x_\tau=(1-\tau)x_0+\tau s_+}}\left[\|v_\phi^\tau(x_\tau|s,a_{t:t+h}) - (s_+-x_0)\|^2\right]$$

$$\mathcal{L}_{\text{td-mc-bs}}(\phi) = \mathbb{E}_{\substack{\tau\sim U(0,1),(s_t,a_{t:t+h},s_+)\sim\mathcal{D}^\beta \\ x_0\sim p_0, a_+\sim\beta_\theta(\cdot|s_+), x_\tau=\psi_{\bar{\phi}}^\tau(x_0|s_+,a_+)}}\left[\|v_\phi^\tau(x_\tau|s_t,a_{t:t+h}) - v_{\bar{\phi}}^\tau(x_\tau|s_+,a_+)\|^2\right]$$

where: $s_+ = s_{t+h(1+k)}, k \sim \text{geom}(1-\omega),$ and a_+ also uses chunk size $h = 10$.

B.3 One-step Dynamics Model

As a baseline to geometric horizon models, we utilize auto-regressive prediction with one-step models to sample from the successor measure. This follows from first sampling a random offset $k \sim \text{geom}(1-\gamma)$. We then perform autoregressive prediction for k steps, first sampling $s_{t+1} \sim p(\cdot|s,a)$, and then sampling $k-1$ times from $s_{t+i+1} \sim p^\beta(\cdot|s_{t+i})$, for $i = 1$ to $k-1$. We parameterize $p(\cdot|s,a)$ and $p^\beta(\cdot|s)$ via velocity field $v_\phi^\tau(x_\tau|s,a)$, trained via flow matching:

$$\mathcal{L}_{\text{1-step}}(\phi) = \mathbb{E}_{\substack{\tau\sim\mathcal{U}(0,1),\,(s,a,s')\sim\mathcal{D}^\beta,\,a_m\sim\text{mask}(a) \\ x_0\sim p_0,\,x_\tau=(1-\tau)x_0+\tau s'}}\left[\left\|v_\phi^\tau(x_\tau \mid s, a_m) - (s'-x_0)\right\|_2^2\right] \tag{16}$$

We utilize a standard 1-step flow matching objective, however we condition on $a_m \sim \text{mask}(a)$, where with probability 0.5, the action is replaced by a learned embedding (shared for all actions). Thus, this objective trains both forms of p. We utilize the same architecture as GHM as described in Table 6, and train for 1 million steps.

While in theory, this objective can perfectly capture the successor measure, we found issues with compounding error during autoregressive prediction as shown in Figure 3. This is particularly due to $p^\beta(\cdot|s)$ being memory-less, i.e. after the first action, there is no conditioning information in $p^\beta(\cdot|s)$ that captures an intention (such as running to the right). Thus, we find a stronger performing baseline incorporates history, leading to:

$$\mathcal{L}_{\text{history 1-step}}(\phi) = \mathbb{E}_{\substack{\tau\sim\mathcal{U}(0,1),\,(s_{t-h+1:t},a,s_{t+1})\sim\mathcal{D}^\beta \\ x_0\sim p_0,\,x_\tau=(1-\tau)x_0+\tau s_{t+1} \\ a_m\sim\text{mask}(a)}}\left[\left\|v_\phi^\tau(x_\tau \mid s_{t-h+1:t}, a_m) - (s_{t+1}-x_0)\right\|_2^2\right] \tag{17}$$

Where we utilize a history length of $h = 4$, as used in Figures 3, 7. When sampling with a specific discount γ, with expected horizon $H = \frac{1}{1-\gamma}$, we cap k for rollouts at $2H$. For example, if $\gamma = 0.99$, we cap $k = \min(\text{geom}(1-0.99), 200)$.

As GHMs can sample in O(1) steps, when training a policy, we can efficiently sample a fresh batch of goals, $s_+^i \sim M_\gamma^\beta(\cdot|s_t,a_t)$, for all batch indices, every time we form a batch for the policy update. However, this is not realistic with one-step models at $\gamma = 0.99$. Thus, we instead cache a $2H = 200$ sized trajectory from each state from the dataset, storing $(N, 200)$ states, where N is the original number of transitions (e.g. $1,000,000$). Then, while training the policy, we geometrically sample future states from this window. We utilize this procedure when performing augmentation with one-step dynamics models (Dyn-Aug) as in Table 1.

C RELATION TO 1-STEP POLICY IMPROVEMENT

Proof.

$$\mathbb{E}_{\substack{s\sim d^\beta(s)\\ a\sim\beta(a|s)\\ s_+\sim d^\beta(s_+)}}\left[\frac{Q^\beta(s,a,s_+)}{d^\beta(s_+)}\log\pi_\theta(a\mid s,g=s_+)\right] \tag{18}$$

$$=\frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^\beta(s)\\ a\sim\beta(a|s)\\ s_+\sim d^\beta(s_+)}}\left[\frac{m^\beta(s_+\mid s,a)}{d^\beta(s_+)}\log\pi_\theta(a\mid s,g=s_+)\right] \tag{19}$$

$$=\frac{1}{1-\gamma}\mathbb{E}_{\substack{s\sim d^\beta(s)\\ a\sim\beta(a|s)\\ s_+\sim M^\beta(s_+|s,a)}}\left[\log\pi_\theta(a\mid s,g=s_+)\right]. \tag{20}$$

The second equality uses the identity $Q^\beta(s,a,s_+) = (1-\gamma)^{-1}m^\beta(s_+ \mid s,a)$ in the goal-conditioned setting, and the final change corresponds to importance sampling.

D SAMPLES

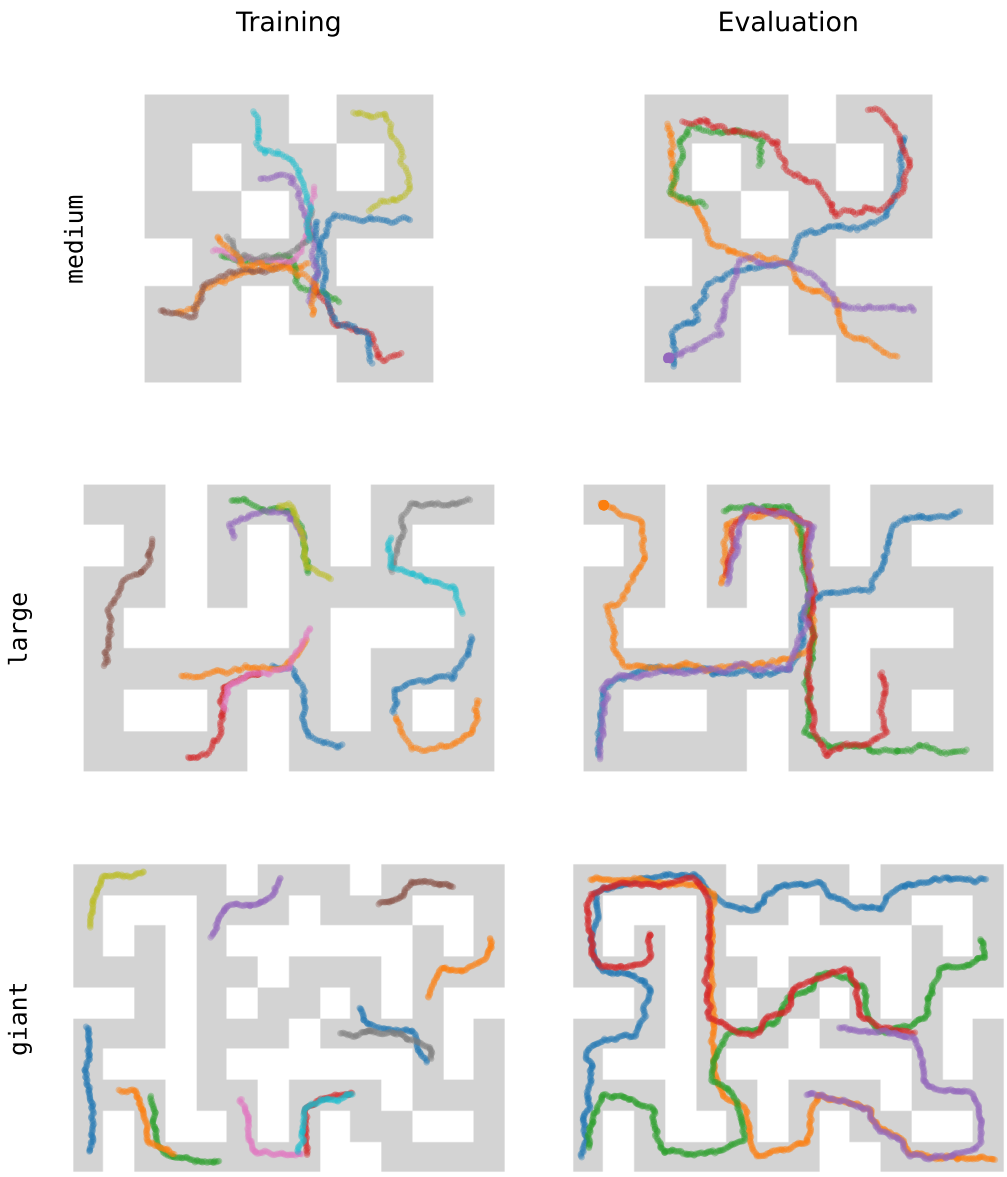
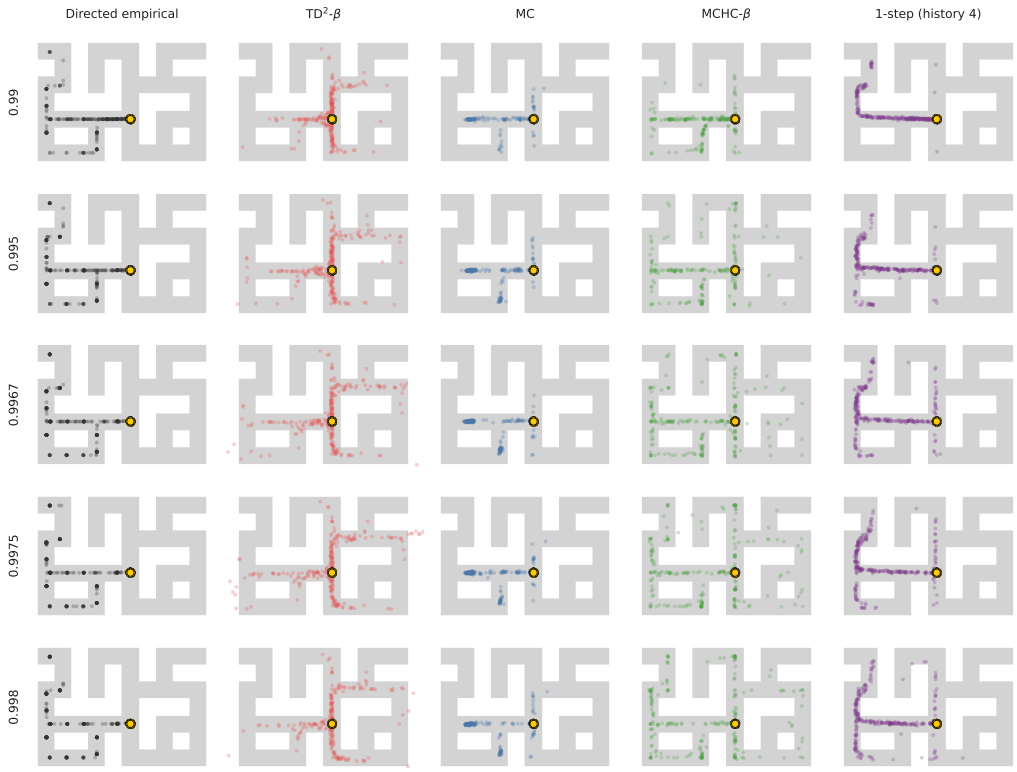
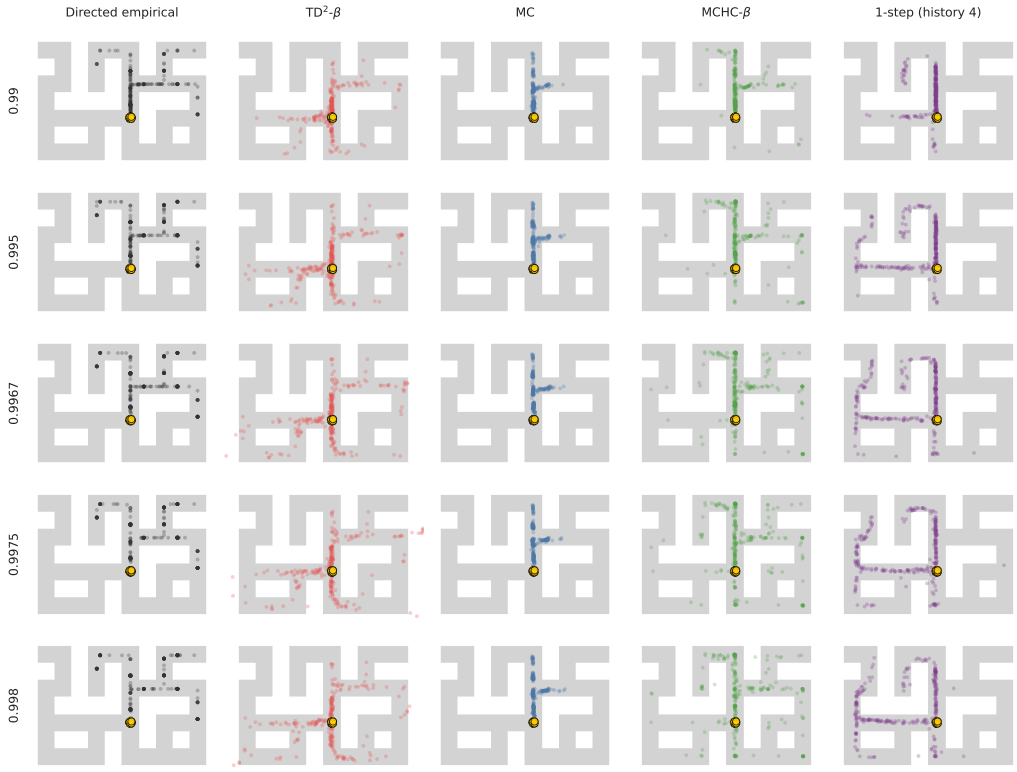


Figure 6: Example ground-truth training and evaluation trajectories on `pointmaze`.

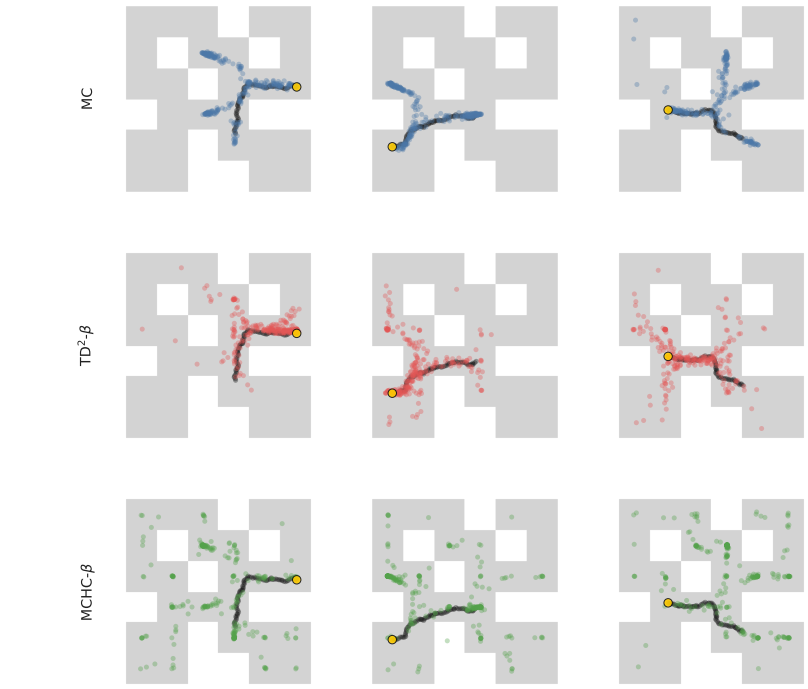


(a)



(b)

Figure 7: **Generated samples in `pointmaze-large` with varying horizon.** Samples are conditioned on different sources displayed with ●, and ground-truth action, whose effect can be inferred by the ground-truth trajectory shown in **black**. Across each row, we vary discount γ, with $H = (1-\gamma)^{-1}$.

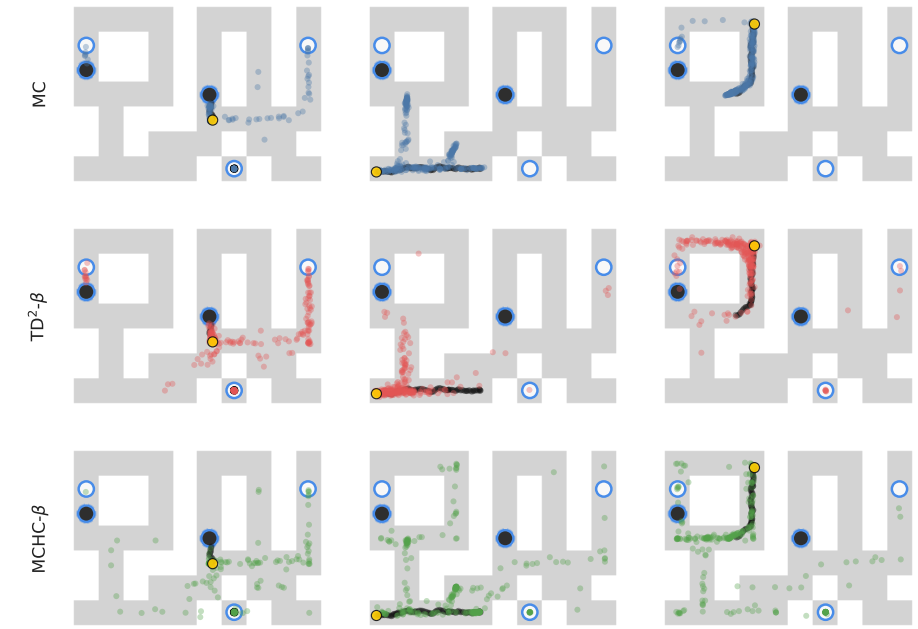


(a) `pointmaze-medium`

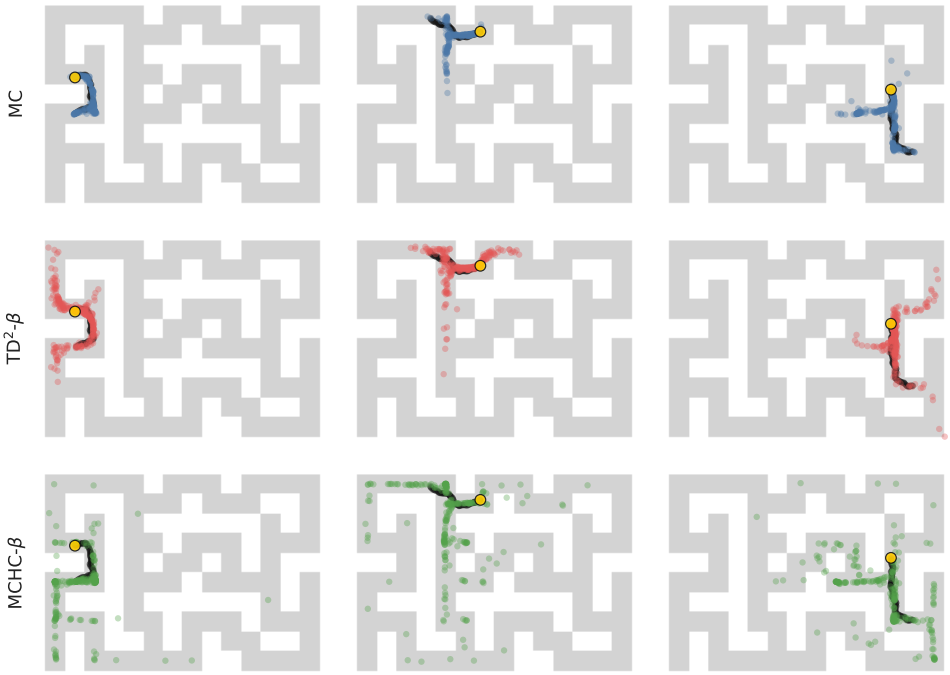


(b) `pointmaze-large`

Figure 8: **Generated samples in `pointmaze` environments.** Samples are conditioned on the source displayed with ○, and ground-truth action, whose effect can be inferred by the ground-truth trajectory shown in **black**. These samples reflect the settings used in Table 1, where MCHC-β uses $\gamma = 0.998$ while TD^2-β and MC use $\gamma = 0.99$.



(c) `pointmaze-teleport`



(d) `pointmaze-giant`

Figure 8: **Generated samples in `pointmaze` environments, continued.**



Figure 9: Samples on `cube-triple-stitch`. The first row shows initial states, and the second shows overlaid empirical futures from the same trajectory. The third row shows overlaid generated samples, while the following rows show individual generated samples.

Table 8: **TD-Aug with Contrastive RL**. We utilize TD-Aug for improving value learning with CRL. We calculate success rate averaged over the last 3 checkpoints, and then report the mean and standard deviation over 3 seeds.

pointmaze	CRL	+MC-Aug	+TD-Aug$_2$	+TD-Aug
medium-stitch	4 ± 2	2 ± 0	24 ± 6	$\mathbf{70} \pm 6$
large-stitch	0 ± 0	0 ± 0	4 ± 5	$\mathbf{46} \pm 17$
giant-stitch	0 ± 0	0 ± 0	0 ± 0	$\mathbf{1} \pm 0$
teleport-stitch	10 ± 5	6 ± 3	6 ± 6	$\mathbf{26} \pm 10$
average	4	2	9	**36**

E Extension to Offline RL with Contrastive RL

In this section, we apply our data augmentation strategy to Contrastive RL (CRL) (Eysenbach et al., 2022). In particular, instead of augmenting goals for training the policy, we use augmentation to learn better goal-conditioned (on-policy) value functions. CRL in OGBench Park et al. (2025a) uses the binary NCE objective to train a critic $f(s, a, g)$:

$$\mathcal{L}_{\text{CRL}} = -\mathbb{E}_{\substack{(s,a)\sim\mathcal{D}^\beta, s_-\sim\mathcal{D}^\beta \\ s_+\sim P_{\text{goal}}(\cdot|s,a)}} \left[\log \sigma(f(s, a, s_+) - \log(1 - \sigma(f(s, a, s_-))\right] \tag{21}$$

Here, $s_+ = s_{\to k}, k \sim \text{geom}(1 - \gamma)$, an MC sample of future state, σ is the sigmoid function, and the critic is parameterized by $f(s, a, g) = \phi(s, a)^T \psi(g)/\sqrt{d}$. With full rollouts from the mixture policy, the optimal critic estimates a log density ratio with the successor measure $f^*(s, a, g) = \log(m^\beta_\gamma(g|s, a)/p(g))$, and thus, $Q^\beta_\gamma(s, a, g) \propto \exp(f^*(s, a, g))$ for a fixed g. However, in practice, standard CRL is subject to the same compositional generalization gap as BC, where we only have truncated trajectories from individual policies, not the mixture policy. Thus, if we can approximately sample goals from the successor measure $M^\beta_\gamma(\cdot|s, a)$ with **TD-Aug**, we can improve the performance of value learning with CRL by replacing $P_{\text{goal}}(\cdot|s, a)$, with $P_{\text{aug}}(\cdot|s, a)$ as in equation 12. With $\alpha = 0$, we maintain the original CRL objective, and with $0 < \alpha \leq 1$, we mix in generated goals with proportion α.

Isolating the contribution of TD-Aug solely to *value learning*, we train the policy on goals sampled without augmentation for DDPG+BC (Fujimoto & Gu, 2021):

$$\arg\max_\theta \mathbb{E}_{(s,a)\sim\mathcal{D}^\beta, g\sim P_{\text{mixed}}(\cdot|s,a)} \left[\lambda \log \pi_\theta(a|s, g) + f(s, \pi^\mu_\theta(s, g), g)\right] \tag{22}$$

Where $P_{\text{mixed}}(\cdot|s, a)$ denotes a proportion of 0.5 for random states in the dataset, and 0.5 for future MC samples, thus only using ground-truth samples. In Table 8, we utilize the standard OGBench (Park et al., 2025a) configuration for CRL, with $\alpha = 0.25$ for the augmentation probability when training the contrastive critic. We can see that augmenting goals with TD-Aug leads to improved overall performance ($4 \to 36$). We also apply our visual setup as in Section 6.2 also utilizing $\alpha = 0.25$, except we reconstruct generated frames with the decoder so that we can perform CRL without modification. Here, we see smaller gains ($56 \to 61$). Further work could study if synthetic samples or other data augmentation may allow for improved scaling of CRL in the offline setting, which has been successful previously in the online setting (Wang et al., 2026).

Table 9: **TD-Aug with Contrastive RL in visual environments.**

visual-antmaze	CRL	+TD-Aug
medium-stitch	82 ± 2	$\mathbf{86} \pm 1$
large-stitch	29 ± 6	$\mathbf{36} \pm 4$
average	56	**61**