CROSS-TRAJECTORY VALUE COMPOSITION FOR OFFLINE GOAL-CONDITIONED REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Offline goal-conditioned reinforcement learning (GCRL) learns goal-reaching policies from fixed datasets. In long-horizon tasks, the data may contain useful parts of a solution across different trajectories without containing a complete path from the start to the goal. Standard temporal-difference learning propagates value information through local one-step backups, so supervision from a distant goal must pass through many updates, making long-horizon values slow and difficult to learn. Existing structured methods address this limitation but often require specialized value models or compose only segments from the same trajectory. We propose **Cross-Trajectory Value Composition (CVC)**, which samples intermediate waypoints across trajectories and uses a discounted triangle relation to combine two shorter goal-reaching problems. This divide-and-conquer approach propagates value information over long distances without imposing a specialized structure on the value function. We incorporate CVC into a hierarchical offline GCRL framework and evaluate it on 24 long-horizon OGBench datasets. CVC achieves the best or tied-best performance on 20 tasks, learns substantially faster than prior methods, and performs especially well on stitching and noisy datasets. These results show that cross-trajectory composition can efficiently learn long-horizon behaviors from fragmented offline experience.

1 INTRODUCTION

Offline goal-conditioned reinforcement learning (GCRL) aims to learn policies that reach specified goals using a fixed dataset of state–action trajectories, without additional training interactions (Park et al., 2023; 2025a). In practice, however, real-world datasets may contain suboptimal behavior, exploratory actions, and partial task executions rather than complete demonstrations. Useful segments of a solution may therefore be distributed across separate trajectories. For example, one trajectory may reach an intermediate region, while another continues from that region toward the goal. Exploiting such experience requires learning how these segments can be combined to solve tasks that were not demonstrated end to end.

A central challenge is propagating value information over long horizons. One-step temporal-difference (TD) learning provides supervision from observed transitions, but information about distant goals must pass through many successive backups, allowing estimation errors to accumulate (Park et al., 2025b). Hierarchical policies introduce intermediate subgoals to simplify action selection, yet selecting useful subgoals still requires accurate estimates of progress toward the final goal (Park et al., 2023). Recent methods improve value propagation by using longer segments of recorded trajectories. OTA (Ahn et al., 2025) learns an additional value function for training the high-level policy, treating short trajectory segments as single steps to reduce the effective planning horizon. TRL (Park et al., 2026) instead composes two value estimates through an intermediate waypoint sampled within the same trajectory. Quasimetric methods also support trajectory stitching by encoding triangle-inequality constraints in their value models (Myers et al., 2025; Wang et al., 2023; Ke et al., 2025). These approaches motivate a complementary question: can explicit value composition through independently sampled dataset waypoints provide useful long-range supervision without requiring a quasimetric value architecture?

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.

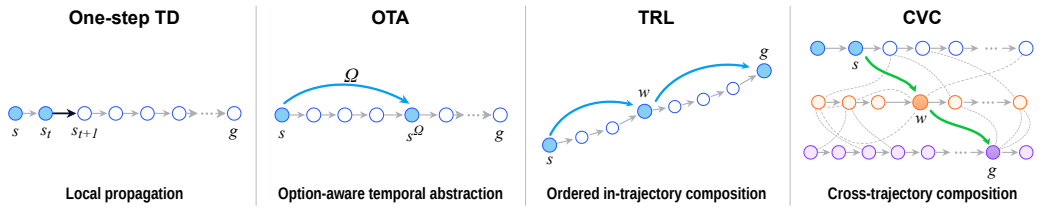


Figure 1: Value propagation strategies in offline GCRL. One-step TD uses local transition backups. OTA (Ahn et al., 2025) learns a high-level value function that treats short trajectory segments as single steps, while TRL (Park et al., 2026) composes values through ordered in-trajectory waypoints. CVC samples the start, waypoint, and goal independently from dataset states, allowing composition across trajectories.

We propose **Cross-Trajectory Value Composition (CVC)**, an auxiliary objective that combines two goal-reaching value estimates through an independently sampled dataset waypoint. The start state, waypoint, and goal need not belong to the same trajectory, allowing the learner to consider routes assembled from dispersed experience. CVC uses the discounted return of a route through the waypoint to supervise the direct state–goal value. However, arbitrary waypoints can introduce long detours whose returns underestimate the quality of the best direct route. We therefore filter candidate compositions according to the estimated values of their constituent subproblems and control their influence through target-clipped expectile regression. Figure 1 contrasts this sampling strategy with local TD updates, temporal abstraction, and in-trajectory composition.

We incorporate CVC into Hierarchical Implicit Q-Learning (HIQL) (Park et al., 2023), which trains its goal-conditioned value function using the one-step Goal-Conditioned Implicit Value Learning (GCIVL) objective. CVC augments this objective with cross-trajectory composition while retaining direct supervision from observed transitions. To stabilize value learning, we additionally combine a demonstrated-return lower bound on TD targets with target-clipped expectile regression. The resulting value function supplies advantages for HIQL's high-level subgoal policy and low-level action policy. CVC retains these hierarchical policy objectives and uses a flexible neural value function without imposing architectural triangle-inequality constraints.

We evaluate CVC on 24 state-based OGBench datasets (Park et al., 2025a), comparing against HIQL, OTA, and LAVL. CVC achieves the highest or tied-highest reported mean success rate on 20 datasets, with strong performance on `giant-stitch` tasks and exploratory data. It also performs well on sequential manipulation tasks. Learning curves show that strong policies can emerge early in training, while ablations support the benefit of cross-trajectory sampling within the composition framework.

Our contributions are:

1. We propose CVC, an offline GCRL method that integrates cross-trajectory value composition with standard GCIVL learning. Independently sampled dataset waypoints and a value-based filter provide compositional supervision without requiring a quasimetric architecture.
2. We develop a stabilized value-learning objective that combines one-step TD learning with a demonstrated-return target lower bound and target-clipped expectile regression to control large updates and the influence of suboptimal compositions.
3. We evaluate CVC across 24 navigation and manipulation datasets in OGBench, demonstrating strong performance against competitive hierarchical methods. Ablations examine cross-trajectory sampling, value-learning stabilization, and sensitivity to the composition weight.

2 Related Work

2.1 Offline goal-conditioned and hierarchical RL

Goal-conditioned reinforcement learning learns policies that reach different goals (Schaul et al., 2015; Liu et al., 2022), but sparse goal-reaching rewards provide limited supervision when successful behavior is rare. Hindsight Experience Replay addresses this difficulty by relabeling experience with achieved goals (Andrychowicz et al., 2017). Related approaches learn from visual goals (Nair

et al., 2018), goal-conditioned imitation (Ghosh et al., 2021; Ding et al., 2019), and classification or contrastive objectives that connect observed future states to goal-reaching values (Eysenbach et al., 2022; 2021; Zheng et al., 2024). Recent online methods also investigate exploration and network scaling for self-supervised goal-reaching (Liu et al., 2025; Wang et al., 2025).

Our work focuses on offline GCRL, which learns from fixed datasets (Park et al., 2023; 2025a; Ma et al., 2022; Yang et al., 2022; Chebotar et al., 2021; Hejna et al., 2023; Sikchi et al., 2024; Venugopal et al., 2026). Suboptimal trajectories may contain useful segments without complete task demonstrations. Weighted imitation (Yang et al., 2022; Hejna et al., 2023) and occupancy-based objectives (Ma et al., 2022; Sikchi et al., 2024) provide approaches to learning from such data. Actionable Models chains experience across episodes through goal-conditioned TD learning (Chebotar et al., 2021), with conservative regularization for unseen actions (Kumar et al., 2020). Goal-conditioned values also support passive-data pretraining (Ma et al., 2023; Ghosh et al., 2023).

Over long horizons, reward information must propagate through many transitions, allowing value estimation errors to accumulate (Park et al., 2025b). Hierarchical methods introduce temporal abstraction (Sutton et al., 1999) and intermediate subgoals (Nachum et al., 2018; Chane-Sane et al., 2021; Gürtler et al., 2021). HIQL extracts both policy levels from one implicit value function (Park et al., 2023), whereas OTA learns an additional high-level value that treats trajectory segments as single steps (Ahn et al., 2025). LAVL uses separate state and goal representations and penalizes excessive value differences across transitions (Kang et al., 2026). CVC retains HIQL's policy objectives and augments value learning with composition.

2.2 Structured values and value composition

Trajectory stitching requires estimating progress along routes that may never appear as complete trajectories in the dataset. Structured value models exploit relationships between goal-reaching distances to support this reasoning. Quasimetrics capture the triangle inequality while allowing asymmetric distances. Interval Quasimetric Embeddings enforce this structure by construction (Wang & Isola, 2022), and Metric Residual Networks combine metric and asymmetric components (Liu et al., 2023). QRL learns quasimetric distances under local transition-cost constraints (Wang et al., 2023). HILP uses latent Euclidean distances for skill learning (Park et al., 2024), while contrastive successor features provide another route to temporal distance structure (Myers et al., 2024).

Temporal Metric Distillation combines contrastive learning with action and temporal consistency constraints (Myers et al., 2025). Multistep Quasimetric Estimation combines observed trajectory offsets with bootstrapped waypoint values (Zheng et al., 2026), while Eikonal-based methods constrain value gradients (Giammarino et al., 2025; Giammarino & Qureshi, 2026). Conservative GCIVL addresses overestimation for dataset-unconnected state–goal pairs by coupling unrestricted TD and quasimetric networks through distillation and conservative regularization (Ke et al., 2025).

Explicit composition constructs supervision through intermediate states. Early work uses triangle-based relaxation (Kaelbling, 1993), and Sub-Goal Trees extends recursive subgoal decomposition to batch learning (Jurgenson et al., 2020). TRL composes action values through ordered in-trajectory waypoints using expectile-weighted binary cross-entropy (Park et al., 2026), providing demonstrated connections between the segments. CVC instead samples the start state, waypoint, and goal independently. This broadens candidate routes but introduces potentially harmful detours, motivating its value-based filter and clipped expectile updates. CVC retains one-step GCIVL supervision without requiring a quasimetric architecture.

3 Preliminaries

3.1 Goal-conditioned reinforcement learning

Offline goal-conditioned reinforcement learning (GCRL) learns a policy $\pi(a \mid s, g)$ from a fixed dataset $\mathcal{D} = \{\tau_i\}_{i=1}^{N}$, without additional environment interaction. Each trajectory contains states and actions, $\tau = (s_0, a_0, s_1, a_1, \ldots, s_T)$. We denote the state and action spaces by $\mathcal{S}$ and $\mathcal{A}$, the initial-state distribution by μ, the transition dynamics by $p(s' \mid s, a)$, and the discount factor by

$\gamma \in (0, 1)$. We consider state-reaching tasks with goal space $\mathcal{G} = \mathcal{S}$ and sparse reward

$$r(s, g) = \begin{cases} 0, & s = g, \\ -1, & s \neq g. \end{cases}$$

Let $V^\pi(s, g)$ denote the expected discounted goal-reaching return from state s under policy π, and let $V^*(s, g) = \sup_\pi V^\pi(s, g)$. Given an evaluation-goal distribution q, the objective is to maximize

$$J(\pi) = \mathbb{E}_{s_0 \sim \mu,\, g \sim q} \left[V^\pi(s_0, g) \right].$$

We construct training goals by relabeling dataset states and compute their rewards using $r(s, g)$ (Andrychowicz et al., 2017). Following HIQL (Park et al., 2023; 2025a), the training-goal distribution $p^{\mathcal{D}}_{\text{mixed}}(\cdot \mid s)$ mixes the current state, a future state from the same trajectory, and a random state from the dataset. This training distribution differs from the evaluation-goal distribution q. The sampling probabilities are reported in Appendix C.

3.2 Goal-Conditioned Implicit Value Learning

We build on Goal-Conditioned Implicit Value Learning (GCIVL), the action-free value objective used by HIQL (Park et al., 2023; 2025a). GCIVL adapts IQL's expectile-based learning (Kostrikov et al., 2022) to goal-conditioned state values using observed transitions. For a residual x, the asymmetric expectile loss is

$$\ell^\tau_2(x) = |\tau - \mathbb{1}[x < 0]|\, x^2, \qquad \tau \in (0, 1).$$

Following the standard presentation in prior GCRL work (Kang et al., 2026; Park et al., 2023), the one-step value objective is

$$\mathcal{L}_{\text{TD}}(V) = \mathbb{E}_{\substack{(s,s') \sim \mathcal{D}, \\ g \sim p^{\mathcal{D}}_{\text{mixed}}(\cdot \mid s)}} \left[\ell^{\tau_{\text{TD}}}_2 \left(r(s, g) + \gamma \bar{V}(s', g) - V(s, g) \right) \right],$$

where $\bar{V}$ is a target value function held fixed during each gradient update, and τ_{TD} is the TD expectile parameter. For $\tau_{\text{TD}} > 1/2$, positive residuals receive greater weight than negative residuals, so targets above the current value estimate have a stronger influence on learning than equally distant targets below it. This one-step objective provides the base value supervision that CVC augments.

4 Cross-Trajectory Value Composition

CVC augments GCIVL with supervision obtained by composing two goal-reaching value estimates through an intermediate waypoint. The waypoint need not lie on a recorded path between the start and goal. A value-based filter selects candidate decompositions, and a clipped expectile loss trains the direct value toward their composed estimates. We retain the one-step GCIVL objective to anchor value learning to observed transitions.

4.1 Cross-trajectory value-composition loss

We sample a start state s, a waypoint w, and a composition goal g randomly from the offline dataset, without requiring them to belong to the same trajectory. Unlike the mixed goals used for TD learning, composition goals and waypoints are sampled independently from dataset states. CVC then estimates the return of reaching g through w using the learned values of the two subproblems.

In a deterministic goal-reaching problem, a path of length n has discounted return $(\gamma^n - 1)/(1 - \gamma)$. Returns for two consecutive path segments therefore compose according to

$$u \oplus_\gamma v := u + \big(1 + (1 - \gamma)u\big)v.$$

Here, u is the return accumulated while reaching the waypoint, and v is the return from the waypoint to the goal. If the first segment lasts n steps, then $1 + (1 - \gamma)u = \gamma^n$. The second segment's return is therefore discounted by the time required to reach the waypoint.

Composing optimal paths through w gives a candidate route from s to g. The optimal direct value is at least as high as the return of this route:

$$V^*(s, w) \oplus_\gamma V^*(w, g) \leq V^*(s, g).$$

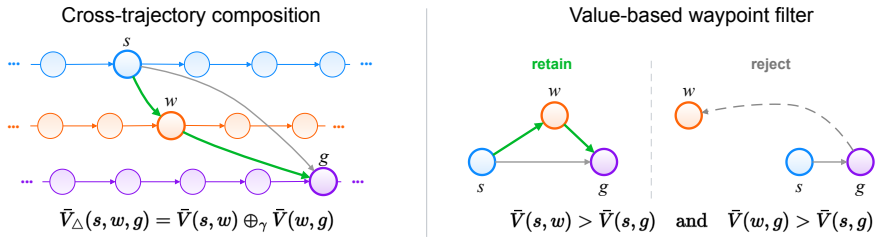


Figure 2: Cross-trajectory value composition and value-based waypoint filtering. CVC combines target values for two goal-reaching subproblems whose endpoints may come from different recorded trajectories. A composition is retained only when both constituent subproblems have higher target values than the direct state–goal problem.

Equality holds when w lies on an optimal route. This relation motivates using compositions to improve direct value estimates: when two well-estimated subproblems yield a higher return than the current direct estimate, their composition provides evidence that the direct value is too low.

Using the target value function $\bar{V}$, we construct

$$\bar{V}_\triangle(s,w,g) := \bar{V}(s,w) + \mathrm{clip}\big(1 + (1-\gamma)\bar{V}(s,w), 0, 1\big)\,\bar{V}(w,g).$$

The clipping operation keeps the estimated discount multiplier within $[0,1]$. We define the compositional advantage as

$$A_\triangle(s,w,g) := \bar{V}_\triangle(s,w,g) - V(s,g).$$

Regressing through arbitrary waypoints can bias direct values downward because many composed routes are suboptimal. For example, when $s = g$, the optimal direct value is zero, whereas a detour through a distinct waypoint has a negative return. Regressing toward such detour targets would decrease an already-correct direct estimate whenever downward compositional updates are allowed. Expectile regression down-weights these negative residuals, but their accumulated influence can still bias the estimate downward.

To reduce this effect, we retain a tuple only when both constituent subproblems have higher target values than the direct problem:

$$M(s,w,g) = \mathbb{1}[\bar{V}(s,w) > \bar{V}(s,g)]\;\mathbb{1}[\bar{V}(w,g) > \bar{V}(s,g)].$$

Because higher values indicate easier goal-reaching problems, this criterion selects waypoints that divide the task into two apparently easier subproblems. At $s = g$, correctly estimated values cause the filter to reject all nontrivial detours. More generally, two individually easier subproblems can still compose into a suboptimal route. The filter therefore reduces detour bias without guaranteeing an optimal decomposition.

Following TRL (Park et al., 2026), we use expectile regression for value composition. We additionally apply target clipping to control the regression residuals on retained compositions. For a prediction x and target y, define

$$\widetilde{y} := \mathrm{sg}\big(\mathrm{clip}(y, x + c^-, x + c^+)\big), \qquad c^- \le 0 \le c^+,$$

and

$$\ell^\tau_{\mathrm{clip}}(y - x, c^-, c^+) := \ell^\tau_2(\widetilde{y} - x).$$

Here, sg denotes stop-gradient. Finite bounds c^- and c^+ limit downward and upward target deviations from the current prediction, respectively. Residuals within these bounds remain unchanged. This follows the error-clipping principle used in DQN (Mnih et al., 2015).

The CVC loss is

$$\mathcal{L}_\triangle = \mathbb{E}_{s,w,g}\Big[M(s,w,g)\,\ell^{\tau_\triangle}_{\mathrm{clip}}\Big(A_\triangle(s,w,g), c^-_\triangle, c^+_\triangle\Big)\Big].$$

Expectile regression favors higher-valued compositions, while the lower clipping bound controls the influence of negative residuals. Setting $c^-_\triangle = 0$ removes negative-residual updates from the CVC loss. Choosing $c^-_\triangle < 0$ permits downward updates with bounded residual magnitude. We select these bounds empirically and report the task-specific settings in Appendix C. We use $c^+_\triangle = \infty$, leaving positive compositional residuals unclipped.

4.2 STABILIZED GCIVL TD OBJECTIVE

CVC relies on learned estimates for both constituent subproblems. We therefore train it alongside the one-step GCIVL objective, which provides supervision from observed transitions. We stabilize this objective using demonstrated trajectory returns and the target-clipped regression loss defined above.

A recorded path provides additional value supervision: under deterministic dynamics, the optimal value should be at least as high as the return of that path. For a future goal sampled from the same trajectory, let $\Delta(s, g)$ denote the number of recorded transitions from s to g; for a current-state goal, we set $\Delta(s, s) = 0$. Following methods that combine bootstrapped estimates with demonstrated returns (Wright et al., 2013; Wilcox et al., 2022), we lower-bound the TD target as

$$\bar{V}_{\mathrm{TD}}(s, s', g) = \max\left(r(s, g) + \gamma \bar{V}(s', g), \frac{\gamma^{\Delta(s,g)} - 1}{1 - \gamma}\right).$$

The first term provides the one-step backup. The second prevents the target from falling below the return demonstrated by the recorded path. For a random goal without a known within-trajectory path, we set $\Delta(s, g) = \infty$. The second term then becomes $-1/(1 - \gamma)$, the minimum possible discounted return, and does not tighten the target within the valid value range.

We apply target-clipped expectile regression to this bounded target:

$$\mathcal{L}^{\mathrm{clip}}_{\mathrm{TD}}(V) = \mathbb{E}_{\substack{(s,s')\sim\mathcal{D}, \\ g\sim p^{\mathcal{D}}_{\mathrm{mixed}}(\cdot|s)}}\left[\ell^{\tau_{\mathrm{TD}}}_{\mathrm{clip}}\left(\bar{V}_{\mathrm{TD}}(s, s', g) - V(s, g), c^{-}_{\mathrm{TD}}, c^{+}_{\mathrm{TD}}\right)\right].$$

The demonstrated-return bound constrains the target's value, whereas residual clipping controls its deviation from the current prediction. Together with CVC, this gives the overall value objective

$$\mathcal{L}_V = \mathcal{L}^{\mathrm{clip}}_{\mathrm{TD}} + \lambda_{\triangle}\mathcal{L}_{\triangle}.$$

The TD term learns from observed transitions, and the CVC term adds supervision from selected compositions across dataset states. All coefficients, clipping bounds, and expectile settings are reported in Appendix C.

We parameterize $V_\theta(s, g)$ with a neural network whose output is constrained to the range of discounted goal-reaching returns. The architecture does not enforce a triangle inequality. Appendix B provides the parameterization.

4.3 HIERARCHICAL POLICY LEARNING

We apply CVC within the hierarchical policy framework of HIQL (Park et al., 2023). The high-level policy $\pi^h(w \mid s, g)$ selects a subgoal conditioned on the current state and final goal, while the low-level policy $\pi^\ell(a \mid s, w)$ produces an environment action for reaching that subgoal. Common policy-extraction approaches include value maximization regularized by behavior cloning (Fujimoto & Gu, 2021) and advantage-weighted regression (Peng et al., 2019). Following HIQL, we use advantage-weighted regression for both policies (Park et al., 2023), with advantages computed from the CVC-trained value function.

Let s_{t+k} denote a future state from the same dataset trajectory, where k is the subgoal horizon. Writing the standard HIQL objectives in state-space notation, the high-level policy loss is

$$\mathcal{L}_{\pi^h} = -\mathbb{E}_{\substack{(s_t, s_{t+k})\sim\mathcal{D}, \\ g\sim p^{\mathcal{D}}_{\mathrm{actor}}(\cdot|s_t)}}\left[\exp(\alpha_h A_h(s_t, s_{t+k}, g)) \log \pi^h(s_{t+k} \mid s_t, g)\right],$$

and the low-level policy loss is

$$\mathcal{L}_{\pi^\ell} = -\mathbb{E}_{(s_t, a_t, s_{t+1}, s_{t+k})\sim\mathcal{D}}\left[\exp(\alpha_\ell A_\ell(s_t, s_{t+1}, s_{t+k})) \log \pi^\ell(a_t \mid s_t, s_{t+k})\right].$$

The corresponding advantage estimates are

$$A_h(s_t, s_{t+k}, g) = V(s_{t+k}, g) - V(s_t, g),$$

$$A_\ell(s_t, s_{t+1}, s_{t+k}) = V(s_{t+1}, s_{t+k}) - V(s_t, s_{t+k}),$$

where $\alpha_h, \alpha_\ell > 0$ control the advantage weighting and $p^{\mathcal{D}}_{\text{actor}}$ denotes the actor-goal sampling distribution. The high-level objective favors dataset subgoals that improve progress toward the final goal, while the low-level objective favors actions that improve progress toward the selected subgoal. In our implementation, the policies operate on latent subgoal representations. Appendix B details this parameterization and the high-level advantage normalization; the actor-goal sampling probabilities are reported in Appendix C.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

We evaluate CVC on 24 state-based datasets from OGBench (Park et al., 2025a), including 20 maze datasets spanning `PointMaze`, `AntMaze`, and `HumanoidMaze`, and four manipulation datasets: `cube-single`, `cube-double`, `cube-triple`, and `Scene`. The maze datasets cover navigation, stitching, and exploration behaviors. `stitch` datasets contain short trajectories that must be combined to solve longer evaluation tasks, while `explore` datasets provide broad coverage through less directed behavior. The manipulation datasets test sequential object manipulation and goal generalization. Appendix A provides environment and dataset details.

We compare against three competitive hierarchical methods: HIQL (Park et al., 2023), OTA (Ahn et al., 2025), and LAVL (Kang et al., 2026). To account for performance degradation during prolonged training, we select the training length separately for each method and dataset using preliminary runs, then retrain each configuration with five random seeds. Appendix C.3 describes the training-step selection procedure and lists the selected training steps; other hyperparameters are provided in Appendix C.

For each final run, we evaluate 50 episodes on each of five fixed tasks at 80%, 90%, and 100% of the selected training steps, giving 750 evaluation episodes per run. We average success across tasks and checkpoints to obtain a score for each run and report the mean and standard deviation across the five seeds.

5.2 BENCHMARK PERFORMANCE

Table 1 compares the complete CVC method with the hierarchical baselines. CVC performs strongly on the `giant-stitch` datasets, achieving success rates of 98% on `PointMaze`, 86% on `AntMaze`, and 88% on `HumanoidMaze`. These tasks require combining short trajectory segments to reach distant goals, making them particularly relevant to our cross-trajectory composition design. CVC also achieves 99% and 98% success on `antmaze-medium-explore` and `antmaze-large-explore`, respectively. These results demonstrate CVC's effectiveness on stitching and exploration datasets, where useful experience is distributed across trajectories.

These results are consistent with CVC's ability to provide supervision for start–goal pairs through waypoints beyond individual recorded trajectories. One-step TD learning can also stitch experience through repeated backups; CVC supplies additional compositional targets that can propagate information between distant states. The sampling ablation below provides more direct evidence for the benefit of allowing cross-trajectory combinations.

On manipulation tasks, CVC achieves 99%, 63%, and 30% success on `cube-single`, `cube-double`, and `cube-triple`, respectively, and 84% on `Scene`. OGBench identifies the `Cube` tasks as requiring some generalization to unseen goals (Park et al., 2025a). The results on `cube-double` and `cube-triple` are therefore consistent with the intended benefit of composing learned reachability relations to support more complex goal configurations.

5.3 LEARNING SPEED AND STABILITY

Figure 3 compares the learning curves of CVC, OTA, and LAVL. CVC learns quickly on the `giant-stitch` datasets: it reaches 85.6% success on `AntMaze` after 100K steps and 85.1% on `HumanoidMaze` after 200K steps. It also reaches 98.2% on `cube-single` after 100K steps. These curves show that strong policies can emerge well before one million training steps.

Table 1: Success rates (%) on 24 OGBench datasets. Results are means and standard deviations across five random seeds. Each run averages 50 evaluation episodes per task over five fixed tasks at 80%, 90%, and 100% of the selected training steps, totaling 750 episodes. All methods follow the training-step selection protocol described in Appendix C.3, which also lists the selected training steps. The Average row reports the unweighted mean across all 24 datasets.

Environment	Type	Size	HIQL	OTA	LAVL	CVC
PointMaze	navigate	medium	93 ± 2	90 ± 2	94 ± 1	$\mathbf{99 \pm 1}$
	navigate	large	58 ± 5	98 ± 2	98 ± 1	$\mathbf{99 \pm 1}$
	navigate	giant	46 ± 7	88 ± 3	91 ± 4	$\mathbf{99 \pm 2}$
	stitch	medium	76 ± 9	80 ± 2	$\mathbf{99 \pm 1}$	94 ± 3
	stitch	large	17 ± 6	66 ± 8	96 ± 2	$\mathbf{99 \pm 1}$
	stitch	giant	0 ± 0	52 ± 7	95 ± 2	$\mathbf{98 \pm 1}$
AntMaze	navigate	medium	96 ± 1	96 ± 1	$\mathbf{97 \pm 1}$	$\mathbf{97 \pm 1}$
	navigate	large	91 ± 2	92 ± 1	$\mathbf{93 \pm 1}$	$\mathbf{93 \pm 1}$
	navigate	giant	65 ± 5	77 ± 4	$\mathbf{85 \pm 2}$	83 ± 1
	stitch	medium	94 ± 1	93 ± 1	$\mathbf{97 \pm 1}$	$\mathbf{97 \pm 1}$
	stitch	large	67 ± 5	84 ± 3	$\mathbf{92 \pm 1}$	$\mathbf{92 \pm 1}$
	stitch	giant	2 ± 2	37 ± 6	82 ± 2	$\mathbf{86 \pm 2}$
	explore	medium	37 ± 10	94 ± 3	$\mathbf{99 \pm 1}$	$\mathbf{99 \pm 1}$
	explore	large	4 ± 5	75 ± 16	93 ± 3	$\mathbf{98 \pm 2}$
HumanoidMaze	navigate	medium	89 ± 2	$\mathbf{94 \pm 1}$	$\mathbf{94 \pm 1}$	$\mathbf{94 \pm 1}$
	navigate	large	49 ± 4	$\mathbf{83 \pm 2}$	74 ± 3	81 ± 2
	navigate	giant	12 ± 4	$\mathbf{92 \pm 1}$	83 ± 2	$\mathbf{92 \pm 2}$
	stitch	medium	88 ± 2	88 ± 2	$\mathbf{93 \pm 2}$	90 ± 3
	stitch	large	28 ± 3	57 ± 3	72 ± 3	$\mathbf{73 \pm 2}$
	stitch	giant	3 ± 2	79 ± 3	80 ± 1	$\mathbf{88 \pm 2}$
Cube	noisy	single	41 ± 6	68 ± 2	95 ± 2	$\mathbf{99 \pm 1}$
	noisy	double	2 ± 1	15 ± 4	22 ± 3	$\mathbf{63 \pm 4}$
	noisy	triple	2 ± 1	15 ± 1	12 ± 2	$\mathbf{30 \pm 6}$
Scene	noisy	–	25 ± 4	35 ± 6	25 ± 3	$\mathbf{84 \pm 4}$
Average			45.2	72.8	81.7	**88.6**

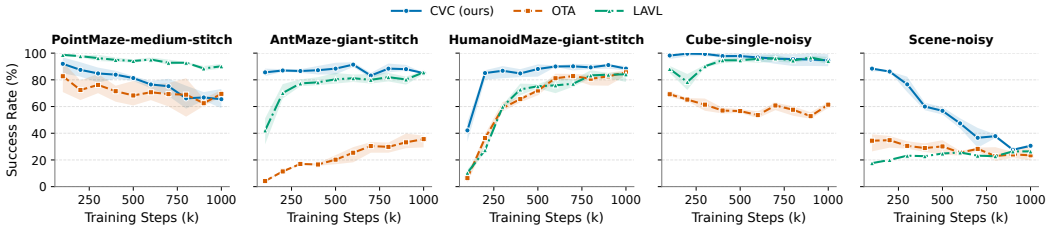


Figure 3: Learning curves of CVC, OTA, and LAVL. Shading indicates the range across runs.

In our experiments, we observe that prolonged training can reduce success rates across methods. On `pointmaze-medium-stitch`, CVC, OTA, and LAVL all decline with continued training, although to different degrees. By contrast, several methods continue improving on the giant-stitch datasets. These observations motivate selecting training steps separately for each method and dataset. We apply the same selection procedure to all compared methods, as detailed in Appendix C.3.

5.4 Ablation Studies

Cross-trajectory sampling. Our central ablation examines the sampling strategy within the CVC composition framework. Figure 4 compares CVC with an in-trajectory variant that samples the starting observation, waypoint, and goal from the same trajectory. Restricting composition in this way reduces success from 98% to 20% on `pointmaze-giant-stitch`, from 98% to 58% on `antmaze-large-explore`, and from 88% to 71% on `humanoidmaze-giant-stitch`. Success also decreases from 99% to 50% on `cube-single` and from 84% to 32% on `Scene`. These results support the benefit of cross-trajectory sampling within the composition framework.

Demonstrated-return lower bound. Removing the demonstrated-return lower bound from the TD target has little effect on four of the five tested datasets, but reduces `Scene` success from 84% to 69%. Its contribution is therefore task dependent, with the clearest benefit appearing on `Scene`.

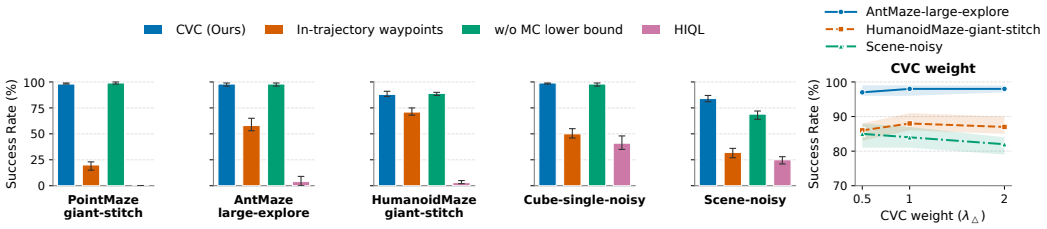


Figure 4: Sensitivity to the composition weight and component ablations. The in-trajectory variant samples the starting observation, waypoint, and goal from the same trajectory. HIQL is included as a reference baseline whose value function is trained using a one-step TD objective without a value-composition loss. Error bars show the range across runs.

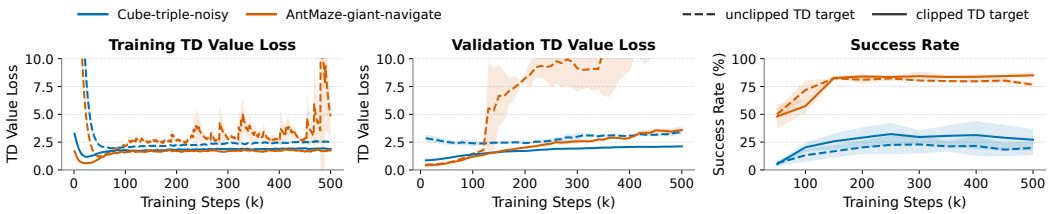


Figure 5: TD target clipping stabilizes value learning and improves goal-reaching performance.

Composition weight. CVC is relatively insensitive to the composition weight $\lambda_\triangle$ within the tested range. Varying $\lambda_\triangle$ over $\{0.5, 1, 2\}$ changes mean success by at most three percentage points on the tested `AntMaze`, `HumanoidMaze`, and `Scene` datasets. We therefore use $\lambda_\triangle = 1$ for all tasks.

TD target clipping. We investigate TD target clipping as a means of stabilizing value learning and limiting potential value overfitting during prolonged training. Figure 5 compares clipped and unclipped TD targets on `antmaze-giant-navigate` and `cube-triple`. Without clipping, the logged losses exhibit larger fluctuations and increasing validation loss, particularly on `AntMaze`, indicating unstable value learning and raising concerns about value overfitting. Clipping produces smoother loss curves and improves policy performance: at 500K training steps, mean success increases from 76.7% to 85.1% on `AntMaze` and from 19.9% to 27.2% on `cube-triple`. Because clipping changes the regression objective, the absolute loss magnitudes are not directly comparable across variants. The accompanying success-rate improvements nevertheless support its practical benefit.

6 Limitations and Future Directions

Our evaluation is limited to state-based OGBench tasks; CVC has not yet been evaluated on other benchmarks or real-world datasets. Testing these settings would clarify how broadly the observed benefits extend. Moreover, composing values across trajectories does not itself provide a mechanism for composing states or object configurations. Combining CVC with entity-centric methods (Haramati et al., 2026) is a promising direction for addressing this complementary challenge.

7 Conclusion

We presented Cross-Trajectory Value Composition (CVC), which augments one-step TD learning with value composition through waypoints sampled across trajectories while retaining HIQL's hierarchical policy objectives. Across 24 OGBench datasets, CVC performs strongly on stitching, exploration, and manipulation tasks. Ablations support the benefit of cross-trajectory sampling and show limited sensitivity to the composition weight. These results support CVC as a practical approach to learning goal-reaching policies from dispersed offline experience.

Reproducibility Statement

We will provide our implementation as supplementary material. Section 4 describes the learning objectives, and Section 5 specifies the experimental evaluation protocol. Appendix A details the environments and datasets, Appendix B describes the value-function architecture and hierarchical policy implementation, and Appendix C reports the hyperparameters and training-step selection procedure, including the training lengths used for each method and dataset.

AI Use Statement

We used generative AI tools to assist with drafting and polishing the manuscript, including improving clarity, wording, and organization. We also used these tools to support literature retrieval and discovery, including identifying relevant studies and locating references. The authors designed and conducted the experiments. Generative AI tools provided feedback on the description of the evaluation protocol and assisted with interpreting the results. The authors reviewed and revised all AI-assisted material and take full responsibility for the accuracy of the references and the final content of the paper, including its claims and interpretations.

References

H. Ahn, H. Choi, J. Han, and T. Moon. Option-aware Temporally Abstracted Value for Offline Goal-Conditioned Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba. Hindsight Experience Replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

E. Chane-Sane, C. Schmid, and I. Laptev. Goal-Conditioned Reinforcement Learning with Imagined Subgoals. In *International Conference on Machine Learning (ICML)*, 2021.

Y. Chebotar, K. Hausman, Y. Lu, T. Xiao, D. Kalashnikov, J. Varley, A. Irpan, B. Eysenbach, R. C. Julian, C. Finn, and S. Levine. Actionable Models: Unsupervised Offline Reinforcement Learning of Robotic Skills. In *International Conference on Machine Learning (ICML)*, 2021.

Y. Ding, C. Florensa, P. Abbeel, and M. Phielipp. Goal-conditioned Imitation Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

B. Eysenbach, R. Salakhutdinov, and S. Levine. C-Learning: Learning to Achieve Goals via Recursive Classification. In *International Conference on Learning Representations (ICLR)*, 2021.

B. Eysenbach, T. Zhang, S. Levine, and R. Salakhutdinov. Contrastive Learning as Goal-Conditioned Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

S. Fujimoto and S. S. Gu. A Minimalist Approach to Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

D. Ghosh, A. Gupta, A. D. Reddy, J. Fu, C. M. Devin, B. Eysenbach, and S. Levine. Learning to Reach Goals via Iterated Supervised Learning. In *International Conference on Learning Representations (ICLR)*, 2021.

D. Ghosh, C. A. Bhateja, and S. Levine. Reinforcement Learning from Passive Data via Latent Intentions. In *International Conference on Machine Learning (ICML)*, 2023.

V. Giammarino and A. H. Qureshi. Goal Reaching with Eikonal-Constrained Hierarchical Quasimetric Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2026.

V. Giammarino, R. Ni, and A. H. Qureshi. Physics-informed Value Learner for Offline Goal-Conditioned Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

N. Gürtler, D. Büchler, and G. Martius. Hierarchical Reinforcement Learning with Timed Subgoals. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

D. Haramati, C. Qi, T. Daniel, A. Zhang, A. Tamar, and G. D. Konidaris. Hierarchical Entity-centric Reinforcement Learning with Factored Subgoal Diffusion. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://openreview.net/forum?id=TimC6hxVHj.

J. Hejna, J. Gao, and D. Sadigh. Distance Weighted Supervised Learning for Offline Interaction Data. In *International Conference on Machine Learning (ICML)*, 2023.

D. Hendrycks and K. Gimpel. Gaussian Error Linear Units (GELUs). arXiv preprint, 2016.

T. Jurgenson, O. Avner, E. Groshev, and A. Tamar. Sub-Goal Trees: A Framework for Goal-Based Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2020.

L. P. Kaelbling. Learning to Achieve Goals. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 1993.

H. Kang, B. Kim, and M. h. Oh. Latent Representation Alignment for Offline Goal-Conditioned Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2026.

K. Ke, Q. Lin, Z. Liu, S. He, and C. Yu. Conservative Offline Goal-Conditioned Implicit V-Learning. In *International Conference on Machine Learning (ICML)*, 2025.

D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

I. Kostrikov, A. Nair, and S. Levine. Offline Reinforcement Learning with Implicit Q-Learning. In *International Conference on Learning Representations (ICLR)*, 2022.

A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative Q-Learning for Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

B. Liu, Y. Feng, Q. Liu, and P. Stone. Metric Residual Network for Sample Efficient Goal-Conditioned Reinforcement Learning. In *AAAI Conference on Artificial Intelligence*, 2023.

G. Liu, M. Tang, and B. 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 (ICLR)*, 2025.

M. Liu, M. Zhu, and W. Zhang. Goal-Conditioned Reinforcement Learning: Problems and Solutions. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2022.

Y. J. Ma, J. Yan, D. Jayaraman, and O. Bastani. Offline Goal-Conditioned Reinforcement Learning via f-Advantage Regression. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. VIP: Towards Universal Visual Reward and Representation via Value-Implicit Pre-Training. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=YJ7o2wetJ2.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level Control through Deep Reinforcement Learning. *Nature*, 2015.

V. Myers, C. Zheng, A. Dragan, S. Levine, and B. Eysenbach. Learning Temporal Distances: Contrastive Successor Features Can Provide a Metric Structure for Decision-Making. In *International Conference on Machine Learning (ICML)*, 2024.

V. Myers, B. C. Zheng, B. Eysenbach, and S. Levine. Offline Goal-Conditioned Reinforcement Learning with Quasimetric Representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

O. Nachum, S. S. Gu, H. Lee, and S. Levine. Data-Efficient Hierarchical Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine. Visual Reinforcement Learning with Imagined Goals. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

S. Park, D. Ghosh, B. Eysenbach, and S. Levine. HIQL: Offline Goal-Conditioned RL with Latent States as Actions. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

S. Park, T. Kreiman, and S. Levine. Foundation Policies with Hilbert Representations. In *International Conference on Machine Learning (ICML)*, 2024.

S. Park, K. Frans, B. Eysenbach, and S. Levine. OGBench: Benchmarking Offline Goal-Conditioned RL. In *International Conference on Learning Representations (ICLR)*, 2025a.

S. Park, K. Frans, D. Mann, B. Eysenbach, A. Kumar, and S. Levine. Horizon Reduction Makes RL Scalable. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025b.

S. Park, A. Oberai, P. Atreya, and S. Levine. Transitive RL: Value Learning via Divide and Conquer. In *International Conference on Learning Representations (ICLR)*, 2026.

X. B. Peng, A. Kumar, G. Zhang, and S. Levine. Advantage-Weighted Regression: Simple and Scalable Off-Policy Reinforcement Learning. arXiv preprint, 2019.

T. Schaul, D. Horgan, K. Gregor, and D. Silver. Universal Value Function Approximators. In *International Conference on Machine Learning (ICML)*, 2015.

H. Sikchi, R. Chitnis, A. Touati, A. Geramifard, A. Zhang, and S. Niekum. Score Models for Offline Goal-Conditioned Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2024.

R. S. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning. *Artificial Intelligence*, 112(1–2):181–211, 1999.

A. Venugopal, J. Chen, X. Wu, C. Zheng, B. Eysenbach, and J. Schneider. Occupancy Reward Shaping: Improving Credit Assignment for Offline Goal-Conditioned Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2026.

K. Wang, I. Javali, M. Bortkiewicz, T. Trzciński, and B. Eysenbach. 1000 Layer Networks for Self-Supervised RL: Scaling Depth Can Enable New Goal-Reaching Capabilities. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

T. Wang and P. Isola. Improved Representation of Asymmetrical Distances with Interval Quasimetric Embeddings. In *NeurIPS Workshop on Symmetry and Geometry in Neural Representations*, 2022.

T. Wang, A. Torralba, P. Isola, and A. Zhang. Optimal Goal-Reaching Reinforcement Learning via Quasimetric Learning. In *International Conference on Machine Learning (ICML)*, 2023.

A. Wilcox, A. Balakrishna, J. Dedieu, W. Benslimane, D. S. Brown, and K. Goldberg. Monte Carlo Augmented Actor-Critic for Sparse Reward Deep Reinforcement Learning from Suboptimal Demonstrations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

R. Wright, S. Loscalzo, P. Dexter, and L. Yu. Exploiting Multi-Step Sample Trajectories for Approximate Value Iteration. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2013.

R. Yang, Y. Lu, W. Li, H. Sun, M. Fang, Y. Du, X. Li, L. Han, and C. Zhang. Rethinking Goal-Conditioned Supervised Learning and Its Connection to Offline RL. In *International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=KJztlfGPdwW.

B. C. Zheng, V. Myers, B. Eysenbach, and S. Levine. Multistep Quasimetric Learning for Scalable Goal-conditioned Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2026.

C. Zheng, R. Salakhutdinov, and B. Eysenbach. Contrastive Difference Predictive Coding. In *International Conference on Learning Representations (ICLR)*, 2024.

A ENVIRONMENTS AND DATASETS

We evaluate on the 24 state-based OGBench datasets listed in Table 1 (Park et al., 2025a): 20 maze datasets and four manipulation datasets. Figure 6 illustrates the environments and data collection settings. The descriptions and specifications below follow OGBench; we use its standard goal-conditioned tasks and success criteria.

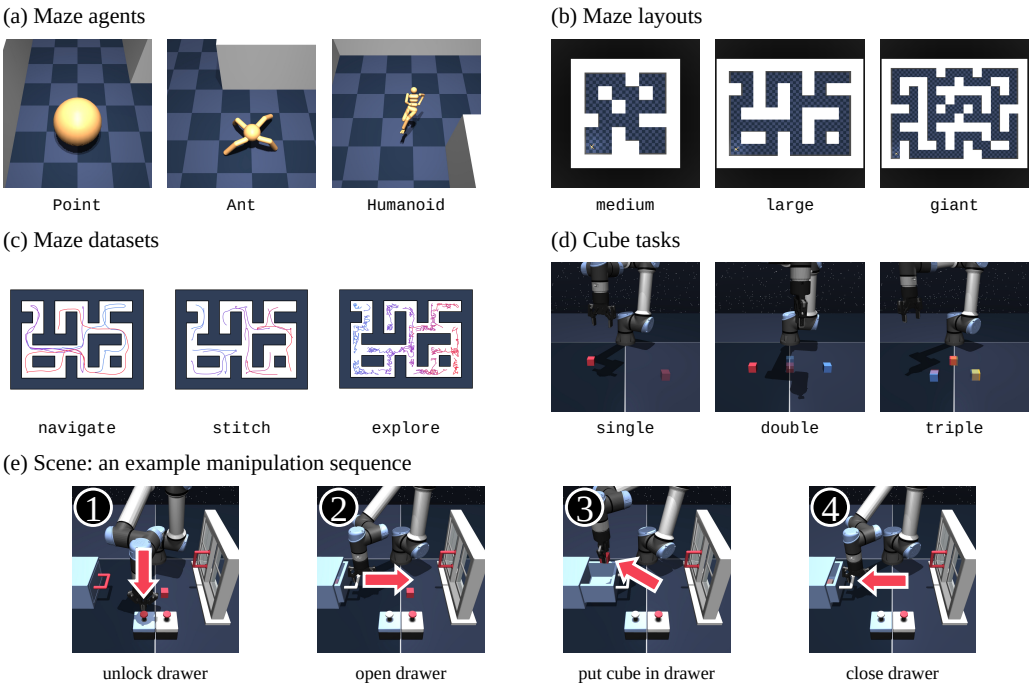


Figure 6: Environments and datasets used in our experiments. (a) Maze agents, (b) maze layouts, (c) example trajectories from the three maze dataset types, (d) `Cube` variants, and (e) an example `Scene` task requiring an ordered sequence of interactions. Images are reproduced from OGBench (Park et al., 2025a, Sections 7.1–7.2, pp. 6–9), with panels rearranged and labels adapted. All experiments use state observations.

A.1 MAZE NAVIGATION

Agents and layouts. `PointMaze` controls a two-dimensional point mass. `AntMaze` and `HumanoidMaze` control an eight-actuator quadruped and a 21-actuator humanoid, respectively, requiring the agent to learn locomotion together with navigation. We use the `medium`, `large`, and `giant` layouts for each agent. Larger mazes introduce longer routes and more intermediate decisions; `giant` is the most demanding layout in our evaluation. Success depends on reaching the target position, without requiring a particular joint configuration.

Dataset types. We use `navigate` and `stitch` datasets for all three agents and all three maze sizes, together with `antmaze-medium-explore` and `antmaze-large-explore`. This gives 18 navigation/stitching datasets and two exploration datasets.

- **`navigate`:** A noisy expert controller repeatedly reaches randomly selected destinations, producing goal-directed trajectories through the maze.
- **`stitch`:** Trajectories connect nearby locations separated by at most four maze cells. Distant evaluation goals therefore require combining experience from multiple short trajectories. The dataset episodes contain 200 steps for `PointMaze` and `AntMaze` and 400 steps for `HumanoidMaze`.

- **explore:** A low-level controller follows random directions resampled every ten steps, with substantial action noise. These trajectories provide broad coverage but contain inefficient movement, testing learning from suboptimal experience.

Each PointMaze dataset and each AntMaze navigate or stitch dataset contains one million transitions; each AntMaze explore dataset contains five million. HumanoidMaze datasets contain two million transitions for medium and large and four million for giant.

A.2 Robotic manipulation

Cube. We use the noisy datasets for cube-single, cube-double, and cube-triple, involving one, two, or three cubes manipulated by a UR5e arm and gripper. Depending on the variant, evaluation goals involve relocating, stacking, swapping, or permuting cubes. Multi-cube tasks require sequencing pick-and-place operations to reach a joint object configuration. OGBench identifies Cube as requiring some generalization to unseen goals (Park et al., 2025a). Success is determined by cube positions; cube orientations and the robot arm pose are not part of the success criterion.

Scene. We use scene-noisy, which contains a cube, a drawer, a window, and two buttons controlling the drawer and window locks. Tasks can require ordered interactions, such as unlocking and opening a drawer, placing the cube inside, and closing it. The longest evaluation task combines eight atomic behaviors. Success depends on the target object configuration rather than the arm pose.

Noisy data. All four manipulation datasets use OGBench's noisy collection protocol: closed-loop scripted controllers execute diverse object interactions with temporally uncorrelated Gaussian action noise. The noise magnitude is sampled for each episode. The cube-single-noisy, cube-double-noisy, and scene-noisy datasets each contain one million transitions, while cube-triple-noisy contains three million.

A.3 Observations and evaluation tasks

Table 2 summarizes the state and action dimensions and maximum evaluation episode lengths. Maze observations contain the agent's position and, for articulated agents, its physical state; manipulation observations describe the arm and objects. The manipulation action space controls three-dimensional end-effector displacement, gripper yaw, and gripper opening. Goals are supplied as state observations. We evaluate on OGBench's five predefined tasks per environment, with its prescribed randomization of initial and goal states. An episode ends when the environment's success criterion is met or the time limit is reached. Evaluation episode lengths can differ from the lengths of trajectories in the offline dataset. The aggregation across tasks, checkpoints, and training seeds follows the experimental setup in the main text.

Table 2: Specifications of the evaluated state-based environments, from OGBench (Park et al., 2025a, Table 7). Episode limits refer to evaluation, not dataset trajectories.

Environment	State dim.	Action dim.	Max. eval. steps
PointMaze (all three sizes)	2	2	1000
AntMaze (all three sizes)	29	8	1000
HumanoidMaze (medium, large)	69	21	2000
HumanoidMaze (giant)	69	21	4000
cube-single	28	5	200
cube-double	37	5	500
cube-triple	46	5	1000
Scene	40	5	750

B ARCHITECTURE AND POLICY IMPLEMENTATION

B.1 VALUE-FUNCTION PARAMETERIZATION

Let $\phi_\theta : \mathcal{S} \to \mathbb{R}^d$ be a shared state encoder, implemented as a neural network followed by output layer normalization. We separately embed the current state and goal,

$$z_s = \phi_\theta(s), \qquad z_g = \phi_\theta(g),$$

and evaluate a value predictor on their concatenation. The resulting state–goal value function is

$$V_\theta(s, g) := -\frac{\sigma(v_\theta(\phi_\theta(s), \phi_\theta(g)))}{1-\gamma},$$

where v_θ is a neural network and σ is the logistic sigmoid. This transformation restricts the output to $(-1/(1-\gamma), 0)$, matching the natural range induced by the sparse goal-reaching reward: zero is the upper bound upon reaching the goal, while $-1/(1-\gamma)$ is the lower bound associated with receiving -1 indefinitely. The predictor changes the HIQL-style conditioning pattern $V(s, \phi(s, g))$ to $v(\phi(s), \phi(g))$ (Park et al., 2023). This is not a quasimetric architecture and does not guarantee the triangle relation globally; CVC instead encourages the relation on selected training tuples.

B.2 HIERARCHICAL POLICY IMPLEMENTATION

Using the subgoal horizon k and dataset subgoal s_{t+k} defined in the main text, we define the relative subgoal representation as

$$\delta z_{t,k} := \phi(s_{t+k}) - \phi(s_t),$$

rather than directly using the absolute embedding $\phi(s_{t+k})$.

The high-level advantage measures whether the subgoal improves progress toward the final goal g:

$$A_h(s_t, s_{t+k}, g) = V(s_{t+k}, g) - V(s_t, g).$$

The high-level Gaussian policy is conditioned on $(\phi(s_t), \phi(g))$ and is trained to predict the relative subgoal representation. We implement the main-text high-level objective with this latent parameterization and advantage normalization:

$$\mathcal{L}_{\pi^h} = -\mathbb{E}_{\substack{(s_t, s_{t+k})\sim\mathcal{D}, \\ g\sim p^{\mathcal{D}}_{\text{actor}}(\cdot|s_t)}} \left[\exp\left(\alpha_h \frac{A_h(s_t, s_{t+k}, g)}{\hat{\sigma}_h} \right) \log \pi^h(\delta z_{t,k} \mid \phi(s_t), \phi(g)) \right],$$

where $\hat{\sigma}_h$ is the running standard deviation of the high-level advantage. Rather than tuning α_h separately for each task, we keep it fixed and normalize the advantage by $\hat{\sigma}_h$, making the weighting less sensitive to differences in value scale across tasks. During training, we add small Gaussian noise to the high-level state embedding as input regularization, making subgoal prediction more robust to local perturbations of the learned representation.

The low-level policy is conditioned on the raw state and the same relative subgoal representation. Its advantage is

$$A_\ell(s_t, s_{t+1}, s_{t+k}) = V(s_{t+1}, s_{t+k}) - V(s_t, s_{t+k}),$$

and its objective is

$$\mathcal{L}_{\pi^\ell} = -\mathbb{E}_{(s_t, a_t, s_{t+1}, s_{t+k})\sim\mathcal{D}} \left[\exp(\alpha_\ell A_\ell(s_t, s_{t+1}, s_{t+k})) \log \pi^\ell(a_t \mid s_t, \delta z_{t,k}) \right].$$

Value and actor losses are optimized jointly. During evaluation, we convert the value to an estimated remaining step count using the inverse discounted-return transform. If the goal is estimated to be within k steps, the low-level policy directly receives $\phi(g) - \phi(s_t)$; otherwise, it receives the subgoal proposed by the high-level policy. On `HumanoidMaze` and manipulation tasks, we use a small high-level policy temperature of 0.1 during evaluation.

C HYPERPARAMETERS

C.1 COMMON HYPERPARAMETERS

Unless stated otherwise, we use the common hyperparameters below. Environment-specific overrides are listed separately.

Hyperparameter	Value
Optimizer	Adam (Kingma & Ba, 2015)
Learning rate	0.0003
Minibatch size	1024
Discount factor γ	0.999
Target-network smoothing coefficient	0.005
Encoder dimensions	(512, 512, 512)
Value network dimensions	(512, 512, 512)
High-level actor dimensions	(512, 512, 512)
Activation function	GELU (Hendrycks & Gimpel, 2016)
State/goal embedding dimension d_z	32 (default), 64 (`HumanoidMaze`)
CVC expectile $\tau_\triangle$	0.7
CVC target lower bound $c^-_\triangle$	0 (default), -2 (`stitch`), -5 (`explore`)
CVC target upper bound $c^+_\triangle$	∞
CVC loss coefficient $\lambda_\triangle$	1.0
High-level advantage weighting α_h (after normalization)	5.0
Low-level advantage weighting α_ℓ	3.0
Value-goal sampling probabilities (current, trajectory, random)	(0.2, 0.5, 0.3)
Actor-goal distribution $p^{\mathcal{D}}_{\text{actor}}$ (current, trajectory, random)	(0.0, 0.5, 0.5)

C.2 HYPERPARAMETER TUNING AND TASK-SPECIFIC SETTINGS

The CVC upper target bound $c^+_\triangle$ remains ∞ for every task. The manipulation tasks use a (256, 256) low-level actor. For `antmaze-explore`, $p^{\mathcal{D}}_{\text{actor}}$ samples all actor goals randomly from the dataset instead of using the default actor-goal mixture.

For each environment, the TD target bounds $(c^-_{\text{TD}}, c^+_{\text{TD}})$ are tuned from $\{(-\infty, \infty), (-10, 20), (-10, 50)\}$, and the TD expectile τ_{TD} is tuned from $\{0.65, 0.75\}$. We use a default subgoal time offset of $0.5\,\text{s}$ and convert it to the corresponding number of environment steps. In OGBench, `PointMaze`, `AntMaze`, `Cube`, and `Scene` use a timestep of $0.02\,\text{s}$, giving $k = 25$, whereas `HumanoidMaze` uses a timestep of $0.005\,\text{s}$, giving $k = 100$. For `Scene`, we instead shorten the horizon to $k = 5$ because the task requires higher-frequency manipulation.

Environment	Subgoal horizon k (steps)	c^-_{TD}	c^+_{TD}	τ_{TD}	Low-level actor dimensions
`PointMaze`	25	$-\infty$	∞	0.75	(128, 128)
`AntMaze`	25	-10	50	0.75	(512, 512)
`HumanoidMaze`	100	$-\infty$	∞	0.65	(512, 512, 512)
`Cube`	25	-10	20	0.75	(256, 256)
`Scene`	5	-10	50	0.65	(256, 256)

C.3 TRAINING STEPS

For each method and dataset, we conduct preliminary runs of one million gradient updates, evaluating every 50,000 updates. We score each checkpoint using the mean success rate over the latest three evaluations. At the first checkpoint, we use its score alone; at the second, we use the mean of the first two scores. We select the checkpoint with the highest resulting average and use its training-step count for the final runs. For example, if the mean over 300K, 350K, and 400K is highest, the selected training length is 400K updates.

The table below lists the selected training lengths for HIQL, OTA, LAVL, and CVC in the same row order as Table 1. The suffix k denotes 1,000 gradient updates. Each training length applies to all five random seeds for that method and dataset. Final performance is evaluated at 80%, 90%, and 100% of the listed training steps and aggregated as described in the experimental setup.

Environment	Size	Dataset type	HIQL	OTA	LAVL	CVC
PointMaze	medium	navigate	350k	200k	300k	100k
	large	navigate	650k	300k	250k	200k
	giant	navigate	600k	200k	750k	300k
	medium	stitch	300k	200k	100k	50k
	large	stitch	450k	300k	250k	200k
	giant	stitch	1000k	200k	800k	300k
AntMaze	medium	navigate	1000k	200k	300k	200k
	large	navigate	1000k	1000k	300k	300k
	giant	navigate	1000k	1000k	1000k	500k
	medium	stitch	1000k	200k	300k	200k
	large	stitch	1000k	1000k	300k	300k
	giant	stitch	1000k	1000k	1000k	600k
	medium	explore	1000k	1000k	500k	200k
	large	explore	1000k	1000k	1000k	300k
HumanoidMaze	medium	navigate	1000k	1000k	800k	150k
	large	navigate	1000k	1000k	1000k	400k
	giant	navigate	1000k	1000k	1000k	1000k
	medium	stitch	1000k	1000k	750k	150k
	large	stitch	1000k	1000k	1000k	400k
	giant	stitch	1000k	1000k	1000k	1000k
Cube	single	noisy	100k	100k	100k	50k
	double	noisy	100k	100k	650k	200k
	triple	noisy	1000k	300k	1000k	500k
Scene	–	noisy	150k	100k	1000k	200k