GTRL: GROUNDING DIVIDE-AND-CONQUER VALUE LEARNING WITH TEMPORAL DIFFERENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

In offline goal-conditioned reinforcement learning (GCRL), divide-and-conquer scales to long horizons by joining two shorter segments at a subgoal. However, under stochastic dynamics, the base case of this rule values the luckiest trajectories through the data. The subgoal must also lie on a shared trajectory, so a state-goal pair that no trajectory connects gets no value update at all. To address both, we present **Grounded Transitive RL (GTRL)**, an offline GCRL value learning algorithm that grounds the divide-and-conquer update with a one-step TD target. Over a single step, TD is correct, as its target averages over the successors and needs no subgoal. GTRL adds this target to the composition rather than replacing it, so every pair receives an update, and the composition still carries the long horizon. GTRL also corrects the bias from hindsight relabeling by reweighting each goal against how reachable it was from other successors. We evaluate our algorithm on nineteen OGBench tasks spanning stochastic, deterministic, and stitching environments, where it achieves the highest average success rate.

1 INTRODUCTION

In offline goal-conditioned reinforcement learning (GCRL) (Kaelbling, 1993; Park et al., 2025a), the value function is often fit by temporal difference (TD) learning, which regresses each value toward a target built from the next state. The problem is that the target is itself an estimate, so each backup inherits the error of the last (Park et al., 2025b). A goal T steps away needs $O(T)$ backups. Divide and conquer removes this dependence on the horizon by composing two shorter estimates through a subgoal. So, the same goal now needs only $O(\log T)$ updates (Park et al., 2026). This composition is possible because temporal distances obey the triangle inequality (Kaelbling, 1993; Dhiman et al., 2018; Wang et al., 2023; Piekos et al., 2023).

Transitive RL (TRL) (Park et al., 2026) turns this triangle inequality into a practical update rule. However, the rule assumes deterministic dynamics, and the assumption sits at its base case. Under stochastic dynamics, the same action can lead to several successors, and a logged transition shows only one of them. The update rule chains these base cases together. So, it values the best case, where every random transition goes the agent's way. This overestimates the value (Myers et al., 2025; Bao et al., 2026), and the gap does not shrink with more data. On top of this, the subgoal has to lie between the state and the goal within a trajectory. Without one, there is nothing to compose, and the state-goal pair never receives a target.

Our key insight is that both faults have the same fix, and that fix is TD learning, which divide and conquer originally set out to replace. TD learning is slow and faulty over a long horizon, but over a single step it is correct. Its target is an expectation over the next state, so the randomness is averaged. It also needs only a transition and a goal, so every state-goal pair gets a target. The triangle inequality still holds under stochastic dynamics (Myers et al., 2024). So, we keep the value composition and ground it in a one-step target. Figure 1 illustrates the argument.

In this work, we introduce the resulting algorithm **Grounded Transitive RL (GTRL)**. It has two components, and the first is counterfactual goals. We sample goals from the whole dataset as well as from the state's own trajectory (Andrychowicz et al., 2017), and give every state-goal pair a one-step target. Where a subgoal also exists, the divide-and-conquer target and the TD target are both valid, so we take the larger. The second is hindsight weighting. A goal from the state's own trajectory lies downstream of the successor that happened. So, under stochastic transitions, the critic credits the

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.

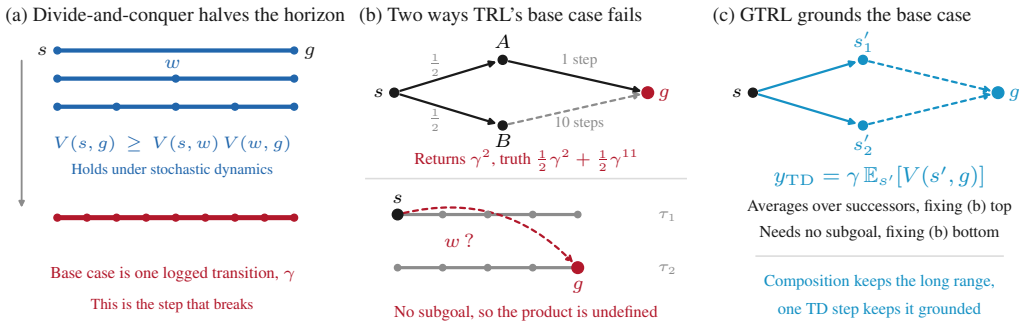


Figure 1: **Where divide and conquer breaks.** (a) The transitive rule halves the horizon down to one logged transition, and base case is the step that breaks. (b) It breaks twice. Under stochastic dynamics, it reads the lucky successor as the agent's own choice (top), and it has no subgoal when the state and the goal lie on different trajectories (bottom). (c) A one-step TD target averages over the successors and needs no subgoal, so it repairs both, while the composition keeps the long horizon.

action with a goal only one successor could reach. We remove this bias by weighting each such goal with a ratio of state occupancies (Blier et al., 2021; Schramm et al., 2023). Every term in the ratio comes from the critic, so it costs one forward pass and no new network.

Contributions. Our contributions are a diagnosis of where divide and conquer breaks and an algorithm that repairs it. The transitive update converges to the value of the luckiest route through the data, and it never trains the state-goal pairs that no trajectory connects. GTRL grounds that rule in a one-step expectation, and the whole change costs one extra forward pass. We evaluate our algorithm on OGBench's (Park et al., 2025a) long-horizon stochastic, deterministic, and stitch tasks, where it achieves the highest average success rate without trading one setting for another.

2 Related Work

Offline goal-conditioned RL. Offline RL learns a policy from a fixed dataset of tasks, and one of its fundamental problems is that it overestimates the value of actions the dataset never shows (Fujimoto et al., 2019; Kumar et al., 2019; Levine et al., 2020). Existing solutions involve making the value function conservative (Kumar et al., 2020) or keeping the policy close to the behavior policy (Wu et al., 2019; Fujimoto & Gu, 2021; Tarasov et al., 2023; Park et al., 2025c). Goal-conditioned RL asks for one more thing: a single policy must reach any state from any other, and its training goals usually come from hindsight relabeling (Kaelbling, 1993; Andrychowicz et al., 2017; Park et al., 2025a). Many offline goal-conditioned methods adapt offline RL or supervised learning to goals (Ghosh et al., 2021; Yang et al., 2022; Ma et al., 2022; Park et al., 2023; Ke et al., 2025). Others instead use a structure that only goal-reaching has, such as quasimetric value functions (Wang et al., 2023), hierarchical policies (Park et al., 2023), or a contrastive, probabilistic view of the value (Eysenbach et al., 2021; 2022; Zheng et al., 2024). Our proposed method belongs to this second group, and the structure it uses is the triangle inequality that goal-reaching values get from shortest paths.

Composition through subgoals. In deterministic environments, the best route to a goal is at least as good as any route that must pass through a given intermediate state. So the optimal goal-conditioned value satisfies $V^*(s,g) \geq V^*(s,w)\,V^*(w,g)$ for every intermediate state w, an idea that goes back to the earliest work on goal-conditioned RL (Kaelbling, 1993). Later methods have built this inequality into the architecture of the value function (Pitis et al., 2020; Wang & Isola, 2022; Liu et al., 2023; Wang et al., 2023; Myers et al., 2024), or used it to plan through subgoals or suggest the subgoals altogether. (Eysenbach et al., 2019; Nasiriany et al., 2019; Parascandolo et al., 2020; Jurgenson et al., 2020; Chane-Sane et al., 2021; Li et al., 2021). We utilize the value composition property of this triangle inequality (Kaelbling, 1993; Dhiman et al., 2018; Jurgenson et al., 2020; Piekos et al., 2023). Joining two shorter estimates through a subgoal allows us to im-

plement divide-and-conquer, which, in the best case, needs only logarithmic Bellman recursions in the horizon. Other methods such as n-step returns and chunked critics shorten the recursion only by a constant factor, and Monte Carlo targets remove it but add variance on top (Park et al., 2025b; Li et al., 2025; 2026; Park et al., 2026). Yet, without planning, value backup methods had mostly been demonstrated to work in two-dimensional mazes and grid worlds. Transitive RL (Park et al., 2026) scaled them up with two changes. It replaced the maximum over subgoals with an in-sample expectile (Kostrikov et al., 2022), and it restricts subgoals to states on the same trajectory as the state and the goal. Both of these changes assume deterministic dynamics.

Stochastic dynamics. When the dynamics are stochastic, prior work changes one of two things: the distance being estimated, or the relabeled data it is trained on. In terms of distance, a temporal distance built from contrastive successor features meets the triangle inequality even under stochastic dynamics (Myers et al., 2024), while temporal metric distillation learns such distances offline with a quasimetric parameterization (Myers et al., 2025). Distributional distance classifiers estimate the entire distribution of distances to the goal rather than one single value (Akella et al., 2023). In terms of data, the problem is that hindsight relabeling is unbiased only when the dynamics are deterministic. Otherwise, it overvalues the lucky outcomes (Plappert et al., 2018; Blier & Ollivier, 2021). Outcome-conditioned supervised methods fail for the same reason, unless they condition on quantities that the environment's randomness does not determine (Paster et al., 2022; Yang et al., 2023). USHER (Schramm et al., 2023) removes the relabeling bias with an importance weight on the Bellman error. However, its weight comes from a separately trained density of future goals, which is a successor representation (Dayan, 1993), and is derived for a one-step backup in the online setting. None of these methods changes the divide-and-conquer transitive update (Park et al., 2026). We fix the divide-and-conquer value composition with a one-step TD update, so the value stays grounded when the dynamics branch, and we take its relabeling correction from the critic, which only costs one forward pass and needs no extra network.

3 Preliminaries

Problem setting. We consider a controlled Markov decision process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, p)$ with state space $\mathcal{S}$, action space $\mathcal{A}$ and transition dynamics $p(s' \mid s, a)$. For offline goal-conditioned reinforcement learning (GCRL), we assume that we are given an unlabeled dataset $\mathcal{D} = \{\tau^{(i)}\}_i$ of trajectories $\tau = (s_0, a_0, s_1, \ldots, s_T)$ with no further access to the environment. In GCRL a task is defined by a state to reach rather than by a reward (Kaelbling, 1993; Schaul et al., 2015). Here, goals lie in state space $\mathcal{S}$, the rewaard is the sparse indicator $0[s \neq g], 1[s = g]$, and one policy function $\pi(a \mid s, g)$ serves a whole family of tasks. For our case, we use the hitting-time form of the objective (Wang et al., 2023), in which the goal is absorbing and the reward is collected upon arrival.

Let $T^\pi(s, g) = \min\{t \geq 0 : s_t = g\}$ be the arrival time of policy $\pi(\cdot \mid \cdot, g)$ started at $s_0 = s$, and $T^\pi(s, g) = \infty$ if it never arrives. The value function and its corresponding Q function $Q(s, a, g) : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ are:

$$V^\pi(s, g) = \mathbb{E}\big[\gamma^{T^\pi(s,g)}\big], \qquad Q^\pi(s, a, g) = \mathbb{E}\big[\gamma^{T^\pi(s,g)} \mid a_0 = a\big], \tag{1}$$

where $\gamma \in (0, 1)$ is the discount factor and $\gamma^\infty = 0$. A larger value means a shorter expected trip. If the episode terminates at each step with probability $1 - \gamma$, then $V^\pi(s, g)$ is the probability of reaching g before the termination. We define optimal V^* and Q^* as the maxima over policy π. The problem is to estimate Q^* from offline dataset $\mathcal{D}$ and extract a policy from it.

Temporal distances. Traditionally in offline GCRL (Eysenbach et al., 2019; Janner et al., 2022; Park et al., 2023; Wang et al., 2023; Park et al., 2026), the dynamics are assumed to be deterministic unless stated otherwise. Therefore, arrival time is not random and the optimal policy takes the shortest route. The temporal distance $d^*(s, g)$ is the smallest number of steps from state s to goal g. So, equation 1 then collapses to $V^*(s, g) = \gamma^{d^*(s,g)}$. Optimal goal-conditioned values are a monotone reparameterization of a shortest-path distance (Kaelbling, 1993; Wang et al., 2023), and it inherits the triangle inequality (Myers et al., 2025) that distances obey. This is the structure the value learning methods below exploit.

3.1 Offline Goal-Conditioned Value Learning

Temporal difference learning. Since the task ends on arrival, Equation 1 satisfies a one-step recursion: $Q^*(s, a, g)$ is 1 when $s = g$, and $\gamma\, \mathbb{E}_{s' \sim p(\cdot|s,a)}[V^*(s', g)]$ otherwise, with $V^*(s, g) = \max_a Q^*(s, a, g)$. Temporal difference (TD) learning fits a critic to this recursion by regression (Mnih et al., 2013), computing the target from a target network $\bar{Q}$. Each backup carries value information one step, so a goal T steps away needs $O(T)$ value backups. Every one is a regression whose error the next inherits. This dependence on the horizon is one of the bottlenecks to scaling offline RL to long-horizon tasks (Park et al., 2025b).

Fitting this recursion requires $\max_{a'} Q(s', a', g)$, which is unsafe offline because the maximum can pick an out of distribution action, and the critic has no data to correct its value (Levine et al., 2020). Implicit Q-learning (Kostrikov et al., 2022; Park et al., 2023) approximates it by an asymmetric regression over in-sample actions instead. For a loss L, a prediction x, a target y and a level $\kappa \in [0.5, 1)$,

$$L_\kappa(x, y) = |\kappa - \mathbb{I}(x > y)| \cdot L(x, y) \tag{2}$$

penalizes underestimating the target by κ and overestimating it by $1 - \kappa$ (Newey & Powell, 1987). Changing κ changes what the critic learns. It fits the average of the targets at $\kappa = 0.5$ (Brandfonbrener et al., 2021) and moves closer to the largest target as κ grows toward 1.

Divide-and-conquer value learning. Divide and conquer replaces the one-step recursion with two composable value estimates. Let w be any intermediate state, which we define as a subgoal. Going from s to g through w is one of many available routes. So, it cannot be shorter than the optimal route. Distances therefore satisfy the triangle inequality $d^*(s, g) \leq d^*(s, w) + d^*(w, g)$ (Kaelbling, 1993; Dhiman et al., 2018; Wang et al., 2023; Piekos et al., 2023). Since $V^*(s, g) = \gamma^{d^*(s,g)}$, the value decomposes into a product, i.e., $V^*(s, g) \geq V^*(s, w)\, V^*(w, g)$, and a subgoal on a shortest route makes it an equality. Taking the best subgoal therefore gives an update rule:

$$Q(s, a, g) \longleftarrow \max_{w \in \mathcal{S},\, a_w \in \mathcal{A}} Q(s, a, w)\, Q(w, a_w, g), \tag{3}$$

where $Q(s, a, g)$ is 1 when $s = g$ and γ when agent reaches goal g in one step. Iterating this rule converges to Q^* in the tabular case (Dhiman et al., 2018; Piekos et al., 2023). Each update joins two estimates instead of adding a step, so at best a horizon of T needs $O(\log T)$ updates (Park et al., 2026) instead of $O(T)$. The cost is the maximum over intermediate state w. It searches the whole state space, so a learned Q picks whichever subgoal it overvalues, and the product inherits the error of both estimators (Levine et al., 2020). This is why methods built on this rule have largely been shown on small domains (Dhiman et al., 2018; Jurgenson et al., 2020; Piekos et al., 2023).

Policy extraction. Since $\arg\max_a Q(s, a, g)$ has no closed form, a critic does not by itself give a policy when the action space is continuous. We extract policy with reparameterized gradients (Fujimoto & Gu, 2021; Park et al., 2024), which train policy π_ϕ to maximize the critic at its own sampled action under a behavior cloning term:

$$J(\phi) = \mathbb{E}_{(s,g) \sim \mathcal{D},\, a^\pi \sim \pi_\phi(\cdot|s,g)}\big[Q(s, a^\pi, g)\big] \;+\; \alpha_{\text{BC}}\, \mathbb{E}_{(s,a,g) \sim \mathcal{D}}\big[\log \pi_\phi(a \mid s, g)\big], \tag{4}$$

where α_{BC} sets how tightly the policy is tied to the data and can be tuned per environment.

4 Methodology

In this work, we extend upon the prior work Transitive RL (TRL) (Park et al., 2026) and develop an effective value learning algorithm for offline GCRL. For deterministic environments, Transitive RL makes Equation 3 practical in two ways. It replaces the maximum over subgoals with the asymmetric regression of Equation 2 and it draws in-trajectory subgoals. So, for a trajectory $\tau \in \mathcal{D}$ and indices $i < k < j$, the critic loss becomes

$$\mathcal{L}_{\text{TRL}}(Q) = \mathbb{E}_{\substack{\tau \sim \mathcal{D},\, i < j \\ k \sim \text{Unif}\{i, \ldots, j-1\}}} \Big[\rho(s_i, s_j)\; L_{0.7}\big(Q(s_i, a_i, s_j),\; \bar{Q}(s_i, a_i, s_k)\, \bar{Q}(s_k, a_k, s_j)\big) \Big], \tag{5}$$

where $\rho(s_i, s_j) = (1 + \log_\gamma \bar{Q}(s_i, a_i, s_j))^{-\lambda}$ puts more weight on short segments. We change two parts of this loss. The first, counterfactual goals, draws goals from outside the trajectory and gives them a one-step target. The second, hindsight weighting, corrects the hindsight bias caused by stochastic state transitions. The next sections explain both choices in detail.

4.1 The Problem

The base case of Equation 3 says that a transition in the dataset can be taken whenever the agent wants. Under stochastic dynamics it is not, because the agent chooses the action but the environment dynamics chooses the successor.

What does this cost? Let G be the directed graph on state space $\mathcal{S}$ with an edge from s to s' whenever $p(s' \mid s, a) > 0$ for some action a. Let $d_G(s, g)$ be the shortest path length from s to g in G. So, $d_G(s, g)$ is the fewest steps needed to reach goal g from state s if the randomness always goes the agent's way. For simplicity, we state the result in the state value form of Equation 3, $V(s, g) \leftarrow \max_{w \in \mathcal{S}} V(s, w)\, V(w, g)$, which has the same base cases.

Proposition 1 (Fixed point of the transitive update)**.** *Assume every edge of graph G appears in dataset $\mathcal{D}$, and let V_n be the result of applying the update n times, starting from the base cases and 0 elsewhere. Let $V_{\mathrm{DC}}(s, g) = \gamma^{d_G(s,g)}$ be the divide-and-conquer value, and let $\mathrm{diam}(G)$ be the largest finite $d_G(s, g)$. Then*

(i) V_n increases pointwise to V_{DC}, and reaches V_{DC} after $\lceil \log_2 \mathrm{diam}(G) \rceil$ steps.

(ii) $V_{\mathrm{DC}} \geq V^*$, with equality at (s, g) if and only if some policy reaches g from s in exactly $d_G(s, g)$ steps almost surely.

(iii) If p is deterministic, $V_{\mathrm{DC}} = V^*$.

The proof is in Appendix A. Intuitively, the update rule chains its base cases together, so what it converges to is the value of the shortest route the agent could conceivably take. Part (iii) recovers the guarantee the rule has in the deterministic case (Dhiman et al., 2018; Piekos et al., 2023). Part (ii) says that once the two come apart the error has one sign, so the rule overestimates. This gap belongs to the fixed point, and more data does not close it.

Example 1 (Teleporter)**.** Let s have a single action that leads to state A or to state B with probability $1/2$ for each. One step reaches g from A, while from B the only route to g is a corridor of length 10 steps. Nothing returns to s. Then $d_G(s, g) = 2$, so $V_{\mathrm{DC}}(s, g) = \gamma^2$, while $V^*(s, g) = \frac{1}{2}\gamma^2 + \frac{1}{2}\gamma^{11}$. At $\gamma = 0.99$ that is $V_{\mathrm{DC}} = 0.980$ against $V^* = 0.938$, or a reported distance of 2 steps against a true 6.4.

The composition step does not fail here. The inequality $V^*(s, g) \geq V^*(s, w)\, V^*(w, g)$ holds under stochastic dynamics as well (Myers et al., 2024), so composing two correct values cannot produce an overestimate on its own. The base case fails, and it fails at exactly stochastic transitions. Its natural replacement is the one-step expectation of the temporal difference backup in Section 3.1.

The second problem is about coverage rather than bias. A subgoal has to lie in between s and g on one trajectory, and so g must appear after s on that trajectory in the first place.

Proposition 2 (Which pairs receive a target)**.** *Let $\mathcal{R}$ be the set of pairs (s_i, s_j) with $i \leq j$ that lie on a common trajectory $\tau \in \mathcal{D}$. Every target formed by Equation 5 is for a pair in $\mathcal{R}$. For $(s, g) \notin \mathcal{R}$ the loss never produces a target, so the value there stays at whatever it was initialized to.*

The two propositions now point at the same missing piece. One says the base case needs a one-step expectation, and the other says the pairs off the trajectory need a target at all. A single one-step term supplies both.

4.2 Counterfactual Goals

In Example 1, the pair (s, g) starts showing up on trajectories that went to B as well. Offline GCRL datasets carry no goals, and so, goals are assigned in hindsight (Andrychowicz et al., 2017). For a state s_i on a trajectory τ, the standard sampler (Park et al., 2023; 2025a) returns one of three things.

It gives back s_i itself, or a later state s_{i+k} on τ with offset k drawn from a $\text{Geometric}(1-\gamma)$ distribution, or a uniformly drawn state from the entire dataset. Transitive RL never uses the third case, because a random goal from the dataset has no subgoal between it and s_i. So, the product in Equation 5 is undefined. However, we use it. Such goals are picked without looking at the trajectory, so the targets for the same input now come from every outcome of a stochastic transition.

The rest of this section is what that choice enables. Let (s, a, s') be a transition from $\mathcal{D}$ and let a' be the action logged at s'. We define $m = 1$ as decomposable goals, which lie ahead of state s on the same trajectory, and $m = 0$ otherwise. TRL trains on the decomposable goals, and their target is $y_{\text{DC}} = \bar{Q}(s, a, w)\,\bar{Q}(w, a_w, g)$ from Equation 5. Non-decomposable goals have no subgoal, so nothing composes. Nevertheless, they still have a one-step target,

$$y_{\text{TD}} \;=\; \begin{cases} 1 & \text{if } g = s, \\ \gamma & \text{if } g = s', \\ \gamma\,\bar{Q}(s', a', g) & \text{otherwise.} \end{cases} \tag{6}$$

Equation 6 is Equation 5 at subgoal offset one, with the first factor at its base case γ and the second at the subsequent logged action. On a decomposable goal the target varies with the subgoal draw. So, setting the expectile κ above 0.5 in Equation 2 maximizes over decompositions. On a non-decomposable goal the target varies with the environment's transition, and there we want the average over outcomes. So,

$$\kappa(s, g) \;=\; 0.7 \;\text{ if } m = 1, \qquad \kappa(s, g) \;=\; 0.5 \;\text{ if } m = 0. \tag{7}$$

Since Equation 6 is defined wherever a transition exists, decomposable goals now carry two targets. Equation 3 is a maximum over subgoals and the one-step case is one of them, so we take the larger,

$$y_{\text{A}} \;=\; \begin{cases} \max\{y_{\text{DC}},\, y_{\text{TD}}\} & \text{if } m = 1, \\ y_{\text{TD}} & \text{if } m = 0. \end{cases} \tag{8}$$

On a non-decomposable goal the subgoal set is empty, so the maximum runs over the one-step case alone.

4.3 Hindsight Weighting

$Q(s, a, g)$ averages over every successor that action a could produce. Hindsight relabeling (Andrychowicz et al., 2017) breaks this, because it picks the goal after the successor is already known. In Example 1, a trajectory that went to A only gives back goals near A, and one that went to B only goals near B. So, for a given goal g, the critic mostly sees the transitions that got closer to it.

Let $\mu(\cdot \mid s)$ be the discounted state occupancy[1] of the behavior policy from state s. The trajectory offset is drawn from $\text{Geometric}(1-\gamma)$ distribution with the same γ the critic uses. So, a decomposable goal is a sample from $\mu(\cdot \mid s')$. We want the average occupancy $\mu(\cdot \mid \tilde{s})$ over the successors $\tilde{s}$ that a could produce. So, we weight each decomposable goal by the ratio of the two. The identity $\mu(g \mid s) = V(s, g)\,\mu(g \mid g)$ (Blier et al., 2021) splits the occupancy into a value and a factor that depends on goal g alone, and that factor cancels in the ratio,

$$h(s, a, s', g) \;=\; \frac{\mathbb{E}_{\tilde{s} \sim p(\cdot \mid s, a)}\big[\mu(g \mid \tilde{s})\big]}{\mu(g \mid s')} \;=\; \frac{\mathbb{E}_{\tilde{s}}\big[V(\tilde{s}, g)\big]}{V(s', g)} \;=\; \frac{Q(s, a, g)}{\gamma\, V(s', g)}. \tag{9}$$

The numerator is the average over outcomes and the denominator is the outcome that occurred. So, at a stochastic transition, a successor state that landed close to goal g is weighted down, and one that landed far from it is weighted up. In the implementation the denominator is $\bar{Q}(s', a', g)$, which Equation 6 already computes. So, hindsight weighting h costs one extra forward pass and no new network.

Two cases need no correction. First, under deterministic dynamics $p(\cdot \mid s, a)$ is a point mass, so $Q(s, a, g) = \gamma V(s', g)$ and $h = 1$ everywhere. Second, a non-decomposable goal is sampled from the whole dataset. So, it is picked without knowing the successor and keeps $h = 1$. In practice, we clip h to $[1/(1+c),\, 1+c]$ following (Schramm et al., 2023), with $c = 1$, and normalize it over the decomposable goals in the batch. This leaves the scale of the loss unchanged.

[1] **Discounted State Occupancy, $\mu(g \mid s)$:** If the episode terminates at each step with probability $1-\gamma$, then $\mu(g \mid s)$ is the probability of being at goal g when it terminates.

4.4 The Full Objective

GTRL samples a transition (s, a, s') from dataset $\mathcal{D}$ with the action a' logged at s', and a goal g from the sampler of Section 4.2. It also samples a subgoal w between s and g when the goal is decomposable. Ultimately, the critic loss becomes

$$\mathcal{L}_{\mathrm{GTRL}}(Q) \;=\; \mathbb{E}\Big[\; h \;\cdot\; \rho \;\cdot\; L_{\kappa(s,g)}\big(Q(s,a,g),\; y_{\mathrm{A}}\big)\;\Big]. \tag{10}$$

The target y_{A} is Equation 8, the expectile $\kappa(s, g)$ is Equation 7, the hindsight weight h is Equation 9, and ρ is the distance weight of Equation 5. The loss $\mathcal{L}_{\mathrm{GTRL}}$ addresses both problems introduced in Section 4.1. It fits a composition where the data supports one, and a one-step expectation everywhere else. For policy extraction, we use Equation 4 from Section 3.1. Pseudo-code Algorithm for GTRL stating the full value update is presented in Appendix C.

5 Experiments

We evaluate GTRL on nineteen OGBench (Park et al., 2025a) tasks, in three groups. Teleport tasks randomly send the agent to one of several destinations, so their transitions are stochastic and the base case of Section 4.1 fails on them. The stitch tasks have only short trajectories, and the agent must join them together to reach a goal. So, most state-goal pairs never lie on a common trajectory, and the transitive loss never trains them. Section 5.1 reports both groups. The remaining ten are standard deterministic navigation and manipulation tasks, and Section 5.2 reports those. Section 5.3 then ablates each part of GTRL to see what it contributes. Each task uses its *-oraclerep-v0* variant, and we report the success rate over its five evaluation goals, averaged over four seeds and the last three checkpoints, as in (Park et al., 2026).

Baselines. Our baselines are prior offline GCRL algorithms from four families, listed below:

- **Behavioral cloning**: BC and FBC (Park et al., 2025a)
- **Temporal difference**: IVL and IQL (Kostrikov et al., 2022; Park et al., 2023), TD and TD-n (Sutton & Barto, 2018)
- **Monte Carlo**: CRL (Eysenbach et al., 2022) and MC (Tian et al., 2021; Shah et al., 2021)
- **Triangle inequality**: QRL (Wang et al., 2023), TDP (Kaelbling, 1993; Dhiman et al., 2018; Jurgenson et al., 2020), COE (Piekos et al., 2023) and TRL (Park et al., 2026)

BC, FBC, IVL and IQL are the goal-conditioned variants (Park et al., 2025a). We present our results with standard deviations, and the best performances are marked in **RED**. Implementation details, full results and hyperparameters are explained in Appendix B.

5.1 Does GTRL repair the faults of divide and conquer?

Tasks. We experiment on nine tasks from OGBench (Park et al., 2025a). The agent is a point mass, an 8-DoF quadruped, a 21-DoF humanoid, or the same quadruped dribbling a soccer ball. In the teleport maze, a teleporter randomly moves the agent to one of three destinations, and one of them is a dead end. In the stitch datasets, the maze trajectories are cut into four-cell pieces, and the task needs up to eight of them joined together to reach its goal.

The top block of Table 1 reports the teleport tasks. GTRL averages 42.2% and is ahead of the second-best GC-IVL (IVL) by 4.2 points (**+11% relative**). Against the divide-and-conquer method TRL, GTRL is ahead on every task by 9.7 points on average. However, the one exception is antmaze-teleport-navigate, where the Monte Carlo method CRL (Eysenbach et al., 2022) finishes a point ahead of GTRL. The bottom block of Table 1 reports the stitch tasks. GTRL averages 35.2% and is ahead of the second-best QRL by 10.4 points (**+42% relative**). Against TRL, the margin is 25.4 points, and TRL itself finishes next to last at 9.8%. The one exception is pointmaze-large-stitch, where the quasimetric method QRL (Wang et al., 2023) reaches 84%, and nothing else passes 34%.

Table 1: **Evaluation on teleport and stitch tasks.** We report success rate (%) and standard deviation over four seeds on OGBench (Park et al., 2025a) tasks. Experiments used *oracle-rep* variants. Best results are marked **RED**, and the second best are underlined.

Environment	Dataset	BC	FBC	IVL	IQL	TD	CRL	MC	QRL	TRL	GTRL
pointmaze-teleport-navigate	Teleport	25 ±3	28 ±2	35 ±3	29 ±4	24 ±7	24 ±9	29 ±4	4 ±4	26 ±5	37 ±7
pointmaze-teleport-stitch		31 ±9	34 ±6	43 ±2	29 ±2	36 ±2	0 ±0	26 ±1	12 ±5	40 ±4	47 ±2
antmaze-teleport-navigate		26 ±3	29 ±3	40 ±4	29 ±3	27 ±3	50 ±2	27 ±2	28 ±4	32 ±2	49 ±2
antmaze-teleport-stitch		31 ±6	31 ±4	34 ±3	18 ±4	24 ±2	12 ±4	22 ±0	13 ±4	32 ±4	36 ±2
Average		28.2	30.5	38.0	26.2	27.7	21.5	26.0	14.2	32.5	42.2
pointmaze-large-stitch	Stitch	7 ±5	20 ±13	12 ±6	34 ±2	0 ±0	0 ±0	0 ±0	84 ±15	0 ±0	19 ±2
antmaze-large-stitch		3 ±3	6 ±4	18 ±2	8 ±1	4 ±1	8 ±5	5 ±1	18 ±2	8 ±1	31 ±4
humanoidmaze-medium-stitch		29 ±5	38 ±3	12 ±2	29 ±7	38 ±1	59 ±1	35 ±3	18 ±2	36 ±0	70 ±3
humanoidmaze-large-stitch		6 ±3	6 ±1	1 ±1	1 ±1	3 ±1	12 ±2	7 ±1	3 ±1	5 ±1	18 ±2
antsoccer-arena-stitch		24 ±8	2 ±3	21 ±3	34 ±2	1 ±1	2 ±1	1 ±0	1 ±1	1 ±0	38 ±2
Average		13.8	14.2	12.8	21.3	9.2	16.4	9.4	24.8	9.8	35.2

Table 2: **Evaluation on standard deterministic tasks.** We report success rate (%) and standard deviation over four seeds on ten deterministic OGBench (Park et al., 2025a) tasks.

Environment	BC	FBC	IVL	IQL	TD-n	CRL	MC	QRL	TDP	COE	TRL	GTRL
pointmaze-large-navigate	25 ±3	71 ±4	48 ±10	34 ±3	31 ±7	33 ±7	4 ±4	7 ±7	30 ±5	34 ±7	33 ±5	79 ±6
antmaze-large-navigate	22 ±4	22 ±4	21 ±6	37 ±8	55 ±6	85 ±7	39 ±5	67 ±8	27 ±3	26 ±3	46 ±5	87 ±1
humanoidmaze-medium-navigate	7 ±2	9 ±3	24 ±4	36 ±3	60 ±3	72 ±5	49 ±3	18 ±17	7 ±0	9 ±2	57 ±1	75 ±4
humanoidmaze-large-navigate	2 ±1	1 ±1	3 ±1	5 ±1	20 ±2	28 ±5	6 ±2	3 ±2	1 ±1	2 ±0	8 ±1	41 ±9
antsoccer-arena-navigate	3 ±1	17 ±2	62 ±2	77 ±3	64 ±4	35 ±5	42 ±3	13 ±1	5 ±1	5 ±2	73 ±4	74 ±1
cube-single-play	7 ±2	18 ±5	88 ±2	95 ±1	95 ±2	63 ±5	98 ±1	6 ±3	3 ±1	13 ±8	95 ±2	93 ±4
cube-double-play	1 ±1	4 ±2	59 ±2	64 ±4	10 ±2	35 ±3	8 ±2	1 ±1	1 ±0	0 ±0	30 ±5	37 ±1
scene-play	4 ±2	23 ±2	68 ±10	61 ±3	75 ±3	26 ±5	25 ±5	6 ±2	12 ±1	8 ±2	77 ±2	84 ±4
puzzle-3x3-play	1 ±1	3 ±1	2 ±1	98 ±0	99 ±0	5 ±1	81 ±12	1 ±1	2 ±0	2 ±0	99 ±0	100 ±0
puzzle-4x4-play	0 ±0	1 ±0	5 ±2	28 ±4	6 ±3	0 ±0	5 ±2	0 ±0	0 ±0	0 ±0	34 ±4	57 ±8
Average	7.2	16.9	38.0	53.5	51.5	38.2	35.7	12.2	8.8	9.9	55.2	72.7

5.2 Does GTRL help on standard tasks as well?

We further evaluate our algorithm on ten standard deterministic tasks, spanning maze navigation, ball control, object manipulation, and puzzle solving (Park et al., 2025a). Our goal here is to determine whether grounding divide-and-conquer value can give comparable performance to existing GCRL algorithms on deterministic tasks as well. According to Table 2, GTRL averages 72.7% and is ahead of the second-best TRL by 17.5 points (**+32% relative**). The largest gains are on the four maze navigation tasks, where GTRL adds between 18 and 46 points over the baselines. The exceptions are *antsoccer-arena-navigate* and the *two cube* tasks, where IQL and MC methods finish ahead. Of these, every triangle inequality method struggles on *cube-double-play*, and GTRL at 37% is still the best of them. GTRL therefore has the highest average in all three environment types, without trading one for another. Figure 2 shows performance on all three environments.

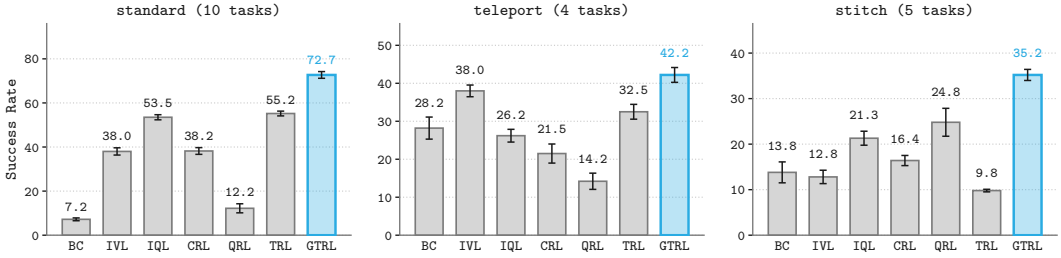


Figure 2: **Average success rate in each environment.** Bars are the average over the tasks of that environment, and the error bars propagate the per-task seed standard deviations.

5.3 Ablation

We run ablations on three core components of GTRL (counterfactual goals, target selection, and hindsight weighting) and present our findings below:

Counterfactual goals. GTRL draws half of the critic batch from the whole dataset instead of the state's own trajectory, and gives each of those goals the one-step target of Equation 6. We ablate this by sampling in-trajectory goals alone, as in Park et al. (2026). This leaves the divide-and-conquer product as the only target. Figure 3 shows a sharp drop on every task, by up to 60.6 points on the deterministic tasks and 20.1 points on the stochastic ones. Counterfactual goals are what make the critic generalize past its own trajectories instead of memorizing them.

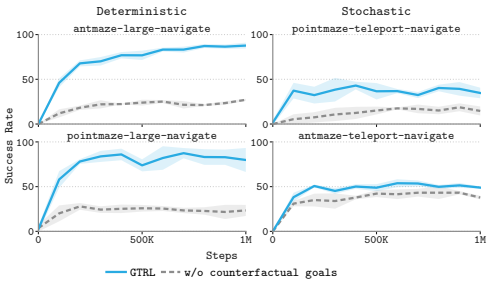


Figure 3: Ablation study on **counterfactual goals.**

Target Selection On decomposable goals, the divide-and-conquer product and the one-step target are both valid, and Equation 8 takes the larger of the two. We ablate this by always taking the divide-and-conquer target on decomposable goals, and the one-step target on non-decomposable goals. Figure 4 shows a massive drop on the deterministic and the stochastic tasks, which is almost as costly as dropping the goals themselves. Target selection is what lets the one-step target fix the composition instead of just filling in for it.

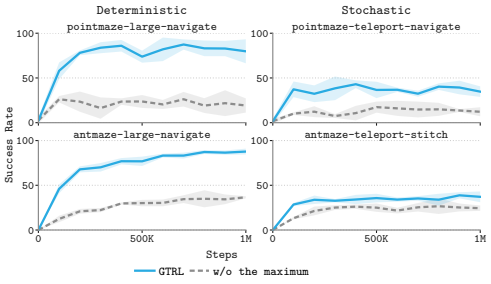


Figure 4: Ablation study on **target selection.**

Hindsight weighting Equation 9 weights each decomposable goal by how reachable it was from the other successors of the same action. We ablate it by setting $h = 1$ everywhere, which returns the loss to plain hindsight relabeling. Figure 5 shows a small drop in deterministic tasks as the transition probability is one. However, on stochastic tasks, enabling hindsight weighting leads to improved performance. Further experiments on the choice of expectile and the goal sampling probability are reported in Appendix B.

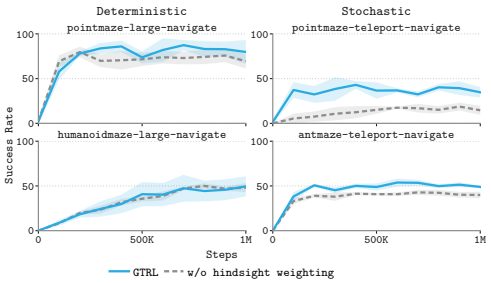


Figure 5: Ablation study on **hindsight weighting.**

6 Conclusion

We presented GTRL, an offline goal-conditioned value learning algorithm that grounds divide-and-conquer updates in temporal differences. Divide-and-conquer reaches a distant goal in $O(\log T)$ updates rather than $O(T)$, but under stochastic dynamics the rule converges to the value of the luckiest route through the data. It also never trains the state-goal pairs that no trajectory connects. GTRL removes both faults with a one-step TD target, which averages over successors and needs no subgoal. Divide-and-conquer value composition remains unchanged, so the long horizon survives. Across nineteen OGBench tasks, GTRL has the highest average success rate on stochastic, stitch, and deterministic tasks, and it does not trade performance in one environment for another.

Limitations. Divide-and-conquer value composition still draws its subgoal from the data, so unconnected pairs rely on one-step backups. The hindsight weight is also read off the critic, so it inherits the critic's error. Natural next steps are to learn subgoals for unconnected pairs and to estimate the weight independently of the critic.

AI USE STATEMENT

We used Claude Opus, a frontier model, to edit the prose of this paper. Its role was to tighten wording and to shorten passages that ran long. The research question, the method, the proofs, the code, and the experiments are our own, and no part of them was produced by a generative AI tool. We read every edit Claude suggested, and we take responsibility for the final content of this paper.

ETHICS STATEMENT

This work studies value learning on simulated offline benchmarks. Every dataset we use comes from OGBench (Park et al., 2025a), which is public and released for research use, and none of it involves human subjects or personally identifying information. Our contribution is a value learning algorithm for goal reaching, so it carries the same risks as offline reinforcement learning in general and none that are specific to it. We declare no conflicts of interest.

REPRODUCIBILITY STATEMENT

Our code is included in the supplementary material, and its README gives the commands needed to reproduce every number we report. We will release it publicly upon publication. Appendix B describes the compute, the baselines, the datasets, the implementation and the evaluation protocol, and Appendix B.6 lists every hyperparameter. Algorithm 1 states the update in full. The assumptions behind each of our claims are stated where the claim is made, and the proofs are in Appendix A.

REFERENCES

Ravi Tej Akella, Benjamin Eysenbach, Jeff Schneider, and Ruslan Salakhutdinov. Distributional distance classifiers for goal-conditioned reinforcement learning. In *ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems*, 2023.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Neural Information Processing Systems (NeurIPS)*, 2017. URL https://arxiv.org/abs/1707.01495.

Erdemt Bao, Xing Lei, and Jun Chen. NFTR: From provable mode-averaging to geodesic subgoal selection in offline goal-conditioned RL. *arXiv preprint arXiv:2607.07855*, 2026.

Léonard Blier and Yann Ollivier. Unbiased methods for multi-goal reinforcement learning. *arXiv preprint arXiv:2106.08863*, 2021.

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

David Brandfonbrener, William F. Whitney, Rajesh Ranganath, and Joan Bruna. Offline RL without off-policy evaluation. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Elliot Chane-Sane, Cordelia Schmid, and Ivan Laptev. Goal-conditioned reinforcement learning with imagined subgoals. In *International Conference on Machine Learning (ICML)*, pp. 1430–1440, 2021.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993. doi: 10.1162/neco.1993.5.4.613.

Vikas Dhiman, Shurjo Banerjee, Jeffrey M. Siskind, and Jason J. Corso. Floyd-Warshall reinforcement learning: Learning from past experiences to reach new goals. *arXiv preprint arXiv:1809.09318*, 2018.

Ben Eysenbach, Russ R Salakhutdinov, and Sergey Levine. Search on the replay buffer: Bridging planning and reinforcement learning. *Advances in neural information processing systems*, 32, 2019.

Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. C-learning: Learning to achieve goals via recursive classification. In *International Conference on Learning Representations (ICLR)*, 2021.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Ruslan Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/e7663e974c4ee7a2b475a4775201ce1f-Abstract-Conference.html.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning (ICML)*, pp. 2052–2062, 2019.

Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

Tom Jurgenson, Or Avner, Edward Groshev, and Aviv Tamar. Sub-goal trees: A framework for goal-based reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2020.

Leslie Pack Kaelbling. Learning to achieve goals. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1094–1099, 1993.

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

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

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy Q-learning via bootstrapping error reduction. In *Neural Information Processing Systems (NeurIPS)*, pp. 11761–11771, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, pp. 1179–1191, 2020.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement learning with action chunking. In *Neural Information Processing Systems (NeurIPS)*, 2025.

Qiyang Li, Seohong Park, and Sergey Levine. Decoupled Q-chunking. In *International Conference on Learning Representations (ICLR)*, 2026.

Yunfei Li, Yilin Wu, Huazhe Xu, Xiaolong Wang, and Yi Wu. Solving compositional reinforcement learning problems via task reduction. In *International Conference on Learning Representations (ICLR)*, 2021.

Bo Liu, Yihao Feng, Qiang Liu, and Peter Stone. Metric residual network for sample efficient goal-conditioned reinforcement learning. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 37, pp. 8799–8806, 2023. doi: 10.1609/aaai.v37i7.26058.

Yecheng Jason Ma, Jason Yan, Dinesh Jayaraman, and Osbert Bastani. Offline goal-conditioned reinforcement learning via f-advantage regression. In *Neural Information Processing Systems (NeurIPS)*, pp. 310–323, 2022.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Vivek Myers, Chongyi Zheng, Anca Dragan, Sergey Levine, and Benjamin Eysenbach. Learning temporal distances: Contrastive successor features can provide a metric structure for decision-making. In *International Conference on Machine Learning (ICML)*, 2024.

Vivek Myers, Bill Zheng, Benjamin Eysenbach, and Sergey Levine. Offline goal-conditioned reinforcement learning with quasimetric representations. *Advances in Neural Information Processing Systems*, 38:19654–19679, 2025.

Soroush Nasiriany, Vitchyr Pong, Steven Lin, and Sergey Levine. Planning with goal-conditioned policies. In *Neural Information Processing Systems (NeurIPS)*, pp. 14814–14825, 2019.

Whitney K. Newey and James L. Powell. Asymmetric least squares estimation and testing. *Econometrica*, 55(4):819–847, 1987. doi: 10.2307/1911031. URL https://www.jstor.org/stable/1911031.

Giambattista Parascandolo, Lars Buesing, Josh Merel, Leonard Hasenclever, John Aslanides, Jessica B. Hamrick, Nicolas Heess, Alexander Neitz, and Theophane Weber. Divide-and-conquer Monte Carlo tree search for goal-directed planning. *arXiv preprint arXiv:2004.11410*, 2020.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. HIQL: Offline goal-conditioned RL with latent states as actions. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline RL? In *Neural Information Processing Systems (NeurIPS)*, 2024.

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, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes RL scalable. In *Neural Information Processing Systems (NeurIPS)*, 2025b.

Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-learning. In *International Conference on Machine Learning (ICML)*, pp. 48104–48127, 2025c.

Seohong Park, Aditya Oberai, Pranav Atreya, and Sergey Levine. Transitive RL: Value learning via divide and conquer. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2510.22512.

Keiran Paster, Sheila A. McIlraith, and Jimmy Ba. You can't count on luck: Why decision transformers and RvS fail in stochastic environments. In *Neural Information Processing Systems (NeurIPS)*, pp. 38966–38979, 2022.

Piotr Piekos, Aditya Ramesh, Francesco Faccio, and Jürgen Schmidhuber. Efficient value propagation with the compositional optimality equation. In *NeurIPS Workshop on Goal-Conditioned Reinforcement Learning*, 2023.

Silviu Pitis, Harris Chan, Kiarash Jamali, and Jimmy Ba. An inductive bias for distances: Neural nets that respect the triangle inequality. In *International Conference on Learning Representations (ICLR)*, 2020.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, Vikash Kumar, and Wojciech Zaremba. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International Conference on Machine Learning (ICML)*, pp. 1312–1320, 2015.

Liam Schramm, Yunfu Deng, Edgar Granados, and Abdeslam Boularias. USHER: Unbiased sampling for hindsight experience replay. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 2073–2082, 2023. URL https://proceedings.mlr.press/v205/schramm23a.html.

Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. *arXiv preprint arXiv:2104.05859*, 2021.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2018.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2023.

Stephen Tian, Suraj Nair, Frederik Ebert, Sudeep Dasari, Benjamin Eysenbach, Chelsea Finn, and Sergey Levine. Model-based visual planning with self-supervised functional distances. In *International Conference on Learning Representations (ICLR)*, 2021.

Tongzhou Wang and Phillip Isola. Improved representation of asymmetrical distances with interval quasimetric embeddings. *arXiv preprint arXiv:2211.15120*, 2022.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning (ICML)*, 2023.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Rui Yang, Yiming Lu, Wenzhe Li, Hao Sun, Meng Fang, Yali Du, Xiu Li, Lei Han, and Chongjie Zhang. Rethinking goal-conditioned supervised learning and its connection to offline RL. In *International Conference on Learning Representations (ICLR)*, 2022.

Sherry Yang, Dale Schuurmans, Pieter Abbeel, and Ofir Nachum. Dichotomy of control: Separating what you can control from what you cannot. In *International Conference on Learning Representations (ICLR)*, 2023.

Chongyi Zheng, Ruslan Salakhutdinov, and Benjamin Eysenbach. Contrastive difference predictive coding. In *International Conference on Learning Representations (ICLR)*, 2024.

A PROOFS

In this section, we provide the grounds for the two propositions of Section 4.1. Let $d = d_G$ be the shortest path length in the graph G defined there, with $d(s, g) = \infty$ when g is unreachable from s. Let $\mathcal{T}$ be the operator that applies the update at every pair at once, so $V_{n+1} = \mathcal{T}V_n$. On a base pair $\mathcal{T}$ writes the base value, which is 1 at $s = g$ and γ when (s, g) is an edge of G, and at every other pair it writes $\max_{w \in \mathcal{S}} V(s, w)\, V(w, g)$.

A.1 PROOF OF PROPOSITION 1

Part (i). $\mathcal{T}$ is monotone. All values lie in $[0, 1]$, so $V \le V'$ gives $V(s, w)V(w, g) \le V'(s, w)V'(w, g)$ at every subgoal w, and the base pairs do not depend on V at all. Also $\mathcal{T}V \ge V_0$ for every V, since $\mathcal{T}V$ equals V_0 on the base pairs and is non-negative elsewhere. So $V_1 \ge V_0$, and monotonicity gives $V_{n+1} \ge V_n$ for every n.

Divide-and-conquer value V_{DC} is a fixed point of $\mathcal{T}$. Joining a shortest path from state s to subgoal w with a shortest path from w to g gives a walk from s to g, so $d(s, g) \le d(s, w) + d(w, g)$. Since discount factor $\gamma < 1$ this gives $V_{\mathrm{DC}}(s, w)\, V_{\mathrm{DC}}(w, g) \le V_{\mathrm{DC}}(s, g)$. The choice $w = s$ attains it, because $d(s, s) = 0$. On a base pair V_{DC} already takes the base value, since $d(s, s) = 0$ and $d(s, g) = 1$ on an edge. So $\mathcal{T}V_{\mathrm{DC}} = V_{\mathrm{DC}}$, and as $V_0 \le V_{\mathrm{DC}}$, monotonicity gives $V_n \le V_{\mathrm{DC}}$ for every n.

It remains to show that $V_n(s, g) = V_{\mathrm{DC}}(s, g)$ whenever $d(s, g) \le 2^n$. At $n = 0$ these are exactly the base pairs. Assume it at n and take a pair with $2^n < m := d(s, g) \le 2^{n+1}$. Then $m \ge 2$, so (s, g) is not a base pair and $\mathcal{T}$ takes the maximum there. Fix a shortest path $s = x_0, \ldots, x_m = g$ and let $w = x_{\lceil m/2 \rceil}$. Every prefix and suffix of a shortest path is itself shortest, so $d(s, w) = \lceil m/2 \rceil$ and $d(w, g) = \lfloor m/2 \rfloor$, and both are at most 2^n. The inductive hypothesis applies to each leg, and w is one of the candidates, so

$$V_{n+1}(s, g) \;\ge\; V_n(s, w)\, V_n(w, g) \;=\; \gamma^{\lceil m/2 \rceil + \lfloor m/2 \rfloor} \;=\; \gamma^m \;=\; V_{\mathrm{DC}}(s, g),$$

and $V_{n+1} \le V_{\mathrm{DC}}$ makes it an equality. Pairs that already agree stay in agreement, since V_n is nondecreasing and bounded above by V_{DC}.

If $d(s, g) = \infty$ then $V_{\mathrm{DC}}(s, g) = 0$. Every w then has $d(s, w) = \infty$ or $d(w, g) = \infty$, since two finite legs would join into a finite walk. So $V_n \le V_{\mathrm{DC}}$ forces one factor of every product to be zero, and $V_n(s, g) = 0$ for every n.

Let the graph diameter of the transition graph G be $\mathrm{diam}(G)$, which is restricted to reachable pairs. Now take $n = \lceil \log_2 \mathrm{diam}(G) \rceil$, the least n with $2^n \ge \mathrm{diam}(G)$. Every pair with finite d satisfies $d(s, g) \le \mathrm{diam}(G) \le 2^n$ and so agrees with V_{DC}, and the remaining pairs agree by the previous paragraph. Hence $V_n = V_{\mathrm{DC}}$.

Part (ii). Fix a policy π and a realization that reaches g from s. Every transition it takes has positive probability under p, so it is an edge of G, and the realization traces a walk from s to g of length $T^\pi(s, g)$. A walk is no shorter than the shortest path, so $T^\pi(s, g) \ge d(s, g)$. A realization that never reaches g has $T^\pi(s, g) = \infty$, so the bound holds there too. Therefore $\gamma^{T^\pi(s,g)} \le \gamma^{d(s,g)}$ almost surely, and taking the expectation and then the maximum over π gives $V^* \le V_{\mathrm{DC}}$.

A random variable bounded above by a constant has that constant as its mean only if it equals it almost surely. So $\mathbb{E}[\gamma^{T^\pi(s,g)}] = \gamma^{d(s,g)}$ holds exactly when $T^\pi(s, g) = d(s, g)$ almost surely. Section 3 defines V^* as a maximum over π, so equality at (s, g) means some policy meets that condition. When $d(s, g) = \infty$ no policy reaches g and both sides are 0.

Part (iii). Let p be deterministic and let (s, s') be an edge of G. Then $p(s' \mid s, a) > 0$ for some a, and a deterministic $p(\cdot \mid s, a)$ puts all of its mass on one successor, so $p(s' \mid s, a) = 1$. The agent can therefore traverse any edge of G at will. A shortest path visits distinct states, so the policy that takes the action leading from x_t to x_{t+1} at each of them is well defined, and it reaches g in exactly $d(s, g)$ steps. Part (ii) then gives $V_{\mathrm{DC}}(s, g) = V^*(s, g)$. If $d(s, g) = \infty$ both sides are 0. □

Part (ii) also locates where the rule leaves V^*. Let (s, g) be an edge of G with $s \ne g$. Every policy has $T^\pi(s, g) \ge 1$ almost surely, so $V^*(s, g) \le \gamma$, which is the base value, with equality exactly

when $p(g \mid s, a) = 1$ for some a. At $s = g$ the base value is $1 = V^*(s, s)$. So the base case sits above V^* at exactly the transitions whose outcome the agent cannot force, which is the claim made in Section 4.1.

A.2 Proof of Proposition 2

Equation 5 draws one trajectory $\tau \in \mathcal{D}$, two indices $i < j$ on it, and a subgoal index $k \in \{i, \dots, j-1\}$. The prediction it regresses is $Q(s_i, a_i, s_j)$, so the pair that draw forms a target for is (s_i, s_j). Both states lie on τ and $i < j$, so $(s_i, s_j) \in \mathcal{R}$. No draw forms a target for any other pair, so every target of the loss is for a pair in $\mathcal{R}$. The two factors of that target are read at (s_i, s_k) and (s_k, s_j), which lie on τ in that order as well, so the loss touches the critic only at pairs in $\mathcal{R}$.

Now take $(s, g) \notin \mathcal{R}$. No draw regresses $Q(s, a, g)$, so the loss does not depend on the value at that pair. In the tabular case its gradient with respect to that entry is zero at every step, and the entry keeps the value it was initialized to. □

With a shared function approximator the entry does move, but only because the network generalizes from the pairs in $\mathcal{R}$. The loss itself still says nothing about it.

A.3 Where the GTRL Fixed Point Lies

The two propositions above are about the transitive update. This one bounds what GTRL converges to. Let β be the policy that collected $\mathcal{D}$ and let $Q^\beta(s, a, g) = \mathbb{E}[\gamma^{T^\beta(s,g)} \mid a_0 = a]$ be its hitting time value. Let $\mathcal{T}_\mathrm{A}$ be the operator that maps a table Q to the fit of Equation 10 against the targets Q produces.

We work in the same tabular setting as before, with $\mathcal{S}$ and $\mathcal{A}$ finite and values in $[0, 1]$, and we assume three things. Each entry is fit exactly, so it equals the $\kappa(s, g)$ expectile of its own targets. The successor s' in Equation 6 is drawn from $p(\cdot \mid s, a)$ and the next action is the logged a', which is what the weight of Equation 9 restores. And $p > 0$, so every pair of dataset states is drawn with positive probability.

Proposition 3 (Where the GTRL fixed point lies)**.** *Iterating $\mathcal{T}_\mathrm{A}$ from $Q_0 = 0$ converges to a fixed point Q_A. At every state-action pair of $\mathcal{D}$ and every goal g in $\mathcal{D}$,*

$$Q^\beta(s, a, g) \;\le\; Q_\mathrm{A}(s, a, g) \;\le\; V_\mathrm{DC}(s, g), \qquad Q^\beta(s, a, g) \;\le\; Q^*(s, a, g) \;\le\; V_\mathrm{DC}(s, g).$$

The proof uses three properties of the expectile of Equation 2 (Newey & Powell, 1987). It lies between the smallest and the largest target. It does not fall when the targets rise or when κ rises. At $\kappa = 0.5$ it is the mean of the targets, which is the statement in Section 3.1.

Monotonicity. The targets do not fall when Q rises. Equation 5 multiplies two entries, both non-negative, Equation 6 scales one entry by γ, and Equation 8 takes their maximum. So $Q \le Q'$ gives $\mathcal{T}_\mathrm{A} Q \le \mathcal{T}_\mathrm{A} Q'$.

Upper bound. Suppose $Q(s, a, g) \le V_\mathrm{DC}(s, g)$ at every entry. On a decomposable goal the target is $Q(s, a, w)\, Q(w, a_w, g) \le \gamma^{d(s,w)+d(w,g)}$, and $d(s, g) \le d(s, w) + d(w, g)$, so it is at most $V_\mathrm{DC}(s, g)$. The one-step target is $\gamma\, Q(s', a', g) \le \gamma^{1+d(s',g)}$, and (s, s') is an edge of G, so $d(s, g) \le 1 + d(s', g)$ and the same bound holds. Its two base cases are $1 = V_\mathrm{DC}(s, s)$ and $\gamma = V_\mathrm{DC}(s, s')$. Every target is therefore at most $V_\mathrm{DC}(s, g)$, and so is the fit. Since $Q_0 = 0 \le V_\mathrm{DC}$, every iterate keeps the bound.

Lower bound. Let $\mathcal{T}_\beta$ be the one-step backup of β, that is $(\mathcal{T}_\beta Q)(s, a, g) = \gamma\, \mathbb{E}[Q(s', a', g)]$ for $s \neq g$, with $Q(g, \cdot, g) = 1$. It is a γ contraction and its fixed point is Q^β. Equation 8 is at least the one-step target at every goal, and Equation 7 keeps $\kappa \ge 0.5$ at every goal, so the fit is at least the mean of the one-step target. Hence $\mathcal{T}_\mathrm{A} Q \ge \mathcal{T}_\beta Q$ for every $Q \ge 0$. Both operators are monotone and both start from $Q_0 = 0$, so $\mathcal{T}_\mathrm{A}^n 0 \ge \mathcal{T}_\beta^n 0$ at every n, and the right side converges to Q^β.

Convergence. $\mathcal{T}_{\mathrm{A}}0 \geq 0$, so monotonicity makes the iterates nondecreasing, and the upper bound keeps them below V_{DC}. A nondecreasing bounded sequence converges, and $\mathcal{T}_{\mathrm{A}}$ is continuous on a finite table, so the limit Q_{A} is a fixed point and carries both bounds.

The second chain. β is one policy, so $Q^{\beta} \leq Q^{*}$. For $s \neq g$ we have $Q^{*}(s,a,g) = \gamma\, \mathbb{E}_{s'}[V^{*}(s',g)]$, and Proposition 1 gives $V^{*}(s',g) \leq \gamma^{d(s',g)} \leq \gamma^{d(s,g)-1}$, so $Q^{*}(s,a,g) \leq V_{\mathrm{DC}}(s,g)$. At $s = g$ both sides are 1. □

The upper bound says what the maximum of Equation 8 costs. Taking the larger of the two targets does not push the fixed point past the divide-and-conquer value of Proposition 1, so GTRL is no more optimistic than the rule it grounds.

The lower bound is what counterfactual goals buy. Outside $\mathcal{R}$ the transitive loss forms no target, so by Proposition 2 the value there is whatever it was initialized to. GTRL has the one-step target there instead, and the bound floors it at Q^{β}, which is positive whenever β reaches g from s with positive probability.

B Experimental Details

B.1 Experimental Setup

We ran every experiment on a single workstation with one NVIDIA GeForce RTX 5090 with 32 GB of memory, an Intel Core i9-14900K with 24 cores, and 61 GB of system memory, under Ubuntu 26.04 with CUDA 13.2. GTRL and all baselines are written in JAX 0.10.2, with Flax 0.12.8 for the networks and Optax 0.2.8 for the optimizer, on Python 3.11. The tasks come from OGBench (Park et al., 2025a), which runs on MuJoCo 3.12.0 through Gymnasium 1.3.0.

B.2 Baselines

Offline GCRL algorithms differ mostly in how they fit the value function, so we group the baselines the same way as Section 5. We use the reference implementations released with OGBench (Park et al., 2025a) and Transitive RL (Park et al., 2026).

Behavioral cloning. BC fits a goal-conditioned Gaussian policy to the dataset actions and learns no value at all. FBC replaces that policy with a flow matching model, which can fit an action distribution with several modes (Park et al., 2025a).

Temporal difference. IQL (Kostrikov et al., 2022) fits the critic with the one-step recursion of Section 3.1 and handles the maximum over actions with the asymmetric regression of Equation 2. IVL is its goal-conditioned state value form (Park et al., 2023). TD applies the same backup over a single step, and TD-n over n steps, clipping n when the goal arrives sooner (Sutton & Barto, 2018). All four carry the horizon dependence of Section 3.1.

Monte Carlo. CRL (Eysenbach et al., 2022) learns the value by telling the states a trajectory reached apart from states drawn at random. MC regresses the critic onto γ^{T} read off the trajectory (Tian et al., 2021; Shah et al., 2021). Neither one bootstraps, so neither inherits the error of the last backup, but both fit the value of the behavior policy rather than the optimal one.

Triangle inequality. QRL (Wang et al., 2023) builds the triangle inequality into the network by parameterizing the value as a quasimetric. TDP (Kaelbling, 1993; Dhiman et al., 2018; Jurgenson et al., 2020) runs Equation 3 with the hard maximum, which we approximate by drawing subgoal candidates from the dataset. COE (Piekos et al., 2023) trains a separate generator to propose the subgoal instead of searching for it. TRL (Park et al., 2026) is Equation 5, which replaces the maximum with an expectile and takes the subgoal from the state's own trajectory.

B.3 Datasets

Every task and dataset comes from OGBench (Park et al., 2025a) and we use them as released.

The teleport mazes are the stochastic group. They are the size of the large maze and they hold teleporters. Entering one sends the agent to one of three exits at random, and one exit is a dead end. So the same action has several successors, and the base case of Section 4.1 fails on them.

The stitch datasets hold short trajectories, each at most four cell units long, and a task needs up to eight of them joined together. So most state and goal pairs never lie on a common trajectory, which is the coverage problem of Proposition 2.

The remaining ten tasks are deterministic. Their navigate datasets come from a noisy expert that repeatedly reaches random goals. The play datasets of cube, scene and puzzle come from scripted policies that move objects around without a task in mind.

Every task uses its *-oraclerep-v0* variant (Park et al., 2026), where the goal is a fixed low-dimensional slice of the state. Table 3 gives the specification of each task.

Table 3: **Dataset specifications.** Transitions, trajectories and dimensions of the nineteen tasks. A dataset trajectory is not as long as an evaluation episode, and the last column is the step budget the agent is given at evaluation time.

Environment	Dataset	Transitions	Trajectories	Traj. length	Obs. dim	Goal dim	Action dim	Episode length
pointmaze-teleport-navigate	Teleport	1M	1000	1000	2	2	2	1000
pointmaze-teleport-stitch		1M	5000	200	2	2	2	1000
antmaze-teleport-navigate		1M	1000	1000	29	2	8	1000
antmaze-teleport-stitch		1M	5000	200	29	2	8	1000
pointmaze-large-stitch	Stitch	1M	5000	200	2	2	2	1000
antmaze-large-stitch		1M	5000	200	29	2	8	1000
humanoidmaze-medium-stitch		2M	5000	400	69	2	21	2000
humanoidmaze-large-stitch		2M	5000	400	69	2	21	2000
antsoccer-arena-stitch		1M	5000	200	42	2	8	1000
pointmaze-large-navigate	Deterministic	1M	1000	1000	2	2	2	1000
antmaze-large-navigate		1M	1000	1000	29	2	8	1000
humanoidmaze-medium-navigate		2M	1000	2000	69	2	21	2000
humanoidmaze-large-navigate		2M	1000	2000	69	2	21	2000
antsoccer-arena-navigate		1M	1000	1000	42	2	8	1000
cube-single-play		1M	1000	1000	28	3	5	200
cube-double-play		1M	1000	1000	37	6	5	500
scene-play		1M	1000	1000	40	7	5	750
puzzle-3x3-play		1M	1000	1000	55	9	5	500
puzzle-4x4-play		1M	1000	1000	83	16	5	500

B.4 Implementation Details

GTRL is built on the Transitive RL implementation (Park et al., 2026). Critic is an ensemble of two networks. Each returns a logit and the value is its sigmoid, so Q stays in $(0, 1)$ where Equation 1 puts it. The loss L inside Equation 2 is the binary cross entropy between the prediction and the target, as in TRL. The target critic $\bar{Q}$ is a Polyak average of the critic at rate η. The policy is trained against the smaller of the two ensemble values, and Equation 4 divides that value by its own absolute mean, so α_{BC} does not have to be retuned when the scale of Q moves.

The composition $\bar{Q}(s, a, w)\,\bar{Q}(w, a_w, g)$ needs the subgoal w to lie in the same space as the state. The *oraclerep* tasks break this, since the goal is a slice of a state and not a state. We follow TRL and keep two critics. The first is conditioned on full states and carries the loss of Equation 10. The second is conditioned on the oracle goal and is fit to the first by regression. Only the second is used for policy extraction.

Equation 9 reuses $\bar{Q}(s', a', g)$, which Equation 6 has already computed, so the weight costs one further forward pass for $\bar{Q}(s, a, g)$ and nothing else. Both terms come from the target critic, so no gradient flows through the weight. We clip it and then normalize it to mean one over the decomposable goals of the batch.

B.5 Training and Evaluation

We train every agent for 10^6 gradient steps on four seeds. Each OGBench task has five evaluation goals and we run 50 episodes for each. A run is scored by its success rate over all five goals, averaged over the checkpoints at 800K, 900K and 1M, following OGBench (Park et al., 2025a). Every table reports the mean and the standard deviation of that score over the four seeds.

B.6 HYPERPARAMETERS

Table 4 lists the hyperparameters that are the same on every task. Table 5 lists the three that are tuned per task, which are the distance weight λ of Equation 5, the behavioral cloning coefficient α_{BC} of Equation 4, and the discount factor γ. Baseline hyperparameters are left as they were in the (Park et al., 2026) default.

Table 4: **Shared hyperparameters.** These are the same on all nineteen tasks.

Hyperparameter	Value
Gradient steps	10^6
Optimizer	Adam
Learning rate	3×10^{-4}
Batch size	1024
MLP size	$[512, 512, 512]$
Nonlinearity	GELU
Layer normalization	True
Target network update rate η	0.005
Subgoal expectile κ $(m = 1)$	0.7
One-step expectile κ $(m = 0)$	0.5
Hindsight weight clip c	1.0
Value goal ratio $(p_{\text{cur}}, p_{\text{traj}}, p_{\text{rand}})$	$(0, 0.5, 0.5)$
Value goal geometric sampling	True
Actor goal ratio $(p_{\text{cur}}, p_{\text{traj}}, p_{\text{rand}})$	$(0, 1, 0)$, and $(0, 0.5, 0.5)$ on the five stitch tasks
Actor goal geometric sampling	False
Policy extraction	Reparameterized gradients
Oracle distillation	True
Evaluation episodes per goal	50

Table 5: **Task-specific hyperparameters.** The distance weight λ, the behavioral cloning coefficient α_{BC}, and the discount factor γ.

Environment	Dataset	λ	α_{BC}	γ
pointmaze-teleport-navigate	Teleport	0.1	0.1	0.99
pointmaze-teleport-stitch		0.7	10.0	0.99
antmaze-teleport-navigate		0.0	0.25	0.99
antmaze-teleport-stitch		0.0	0.25	0.99
pointmaze-large-stitch	Stitch	0.7	0.03	0.99
antmaze-large-stitch		0.0	0.25	0.99
humanoidmaze-medium-stitch		0.0	0.1	0.995
humanoidmaze-large-stitch		0.0	0.1	0.995
antsoccer-arena-stitch		0.25	0.1	0.99
pointmaze-large-navigate	Deterministic	0.1	0.1	0.99
antmaze-large-navigate		0.0	0.25	0.99
humanoidmaze-medium-navigate		0.0	0.1	0.995
humanoidmaze-large-navigate		0.0	0.1	0.995
antsoccer-arena-navigate		0.5	0.25	0.99
cube-single-play		0.4	2.5	0.99
cube-double-play		0.0	7.5	0.99
scene-play		0.7	1.5	0.99
puzzle-3x3-play		0.4	1.5	0.99
puzzle-4x4-play		0.1	1.0	0.99

C ALGORITHM

Algorithm 1 states the full GTRL update. Each gradient step samples a transition and a goal, then marks the goal as decomposable when it lies ahead of the state on the same trajectory. A decomposable goal draws a subgoal and carries both divide-and-conquer and one-step TD targets, and the

Algorithm 1 Grounded Transitive RL (GTRL)

Require: Dataset $\mathcal{D}$, discount γ, clip constant c, target update rate η

Initialize critic Q_θ, target critic $\bar{Q}_{\bar{\theta}} \leftarrow Q_\theta$, and policy π_ϕ
for each gradient step **do**
  Sample transitions (s, a, s') from $\mathcal{D}$ with the action a' logged at s', and a goal g for each
  Set $m = 1$ where g lies ahead of s on the same trajectory, and $m = 0$ otherwise
  Where $m = 1$, sample a subgoal w uniformly between s and g with its logged action a_w
  $y_{\mathrm{DC}} \leftarrow \bar{Q}(s, a, w)\, \bar{Q}(w, a_w, g)$ ▷ Equation 5
  $y_{\mathrm{TD}} \leftarrow \gamma\, \bar{Q}(s', a', g)$, or a base case where $g = s$ or $g = s'$ ▷ Equation 6
  $y_{\mathrm{A}} \leftarrow \max\{y_{\mathrm{DC}}, y_{\mathrm{TD}}\}$ where $m = 1$, and y_{TD} elsewhere ▷ Equation 8
  $\kappa \leftarrow 0.7$ where $m = 1$, and 0.5 elsewhere ▷ Equation 7
  $h \leftarrow \bar{Q}(s, a, g) \,/\, \gamma \bar{Q}(s', a', g)$ where $m = 1$, and 1 elsewhere ▷ Equation 9
  Clip h to $[1/(1 + c), 1 + c]$, then normalize it to mean 1 over the goals with $m = 1$
  $\theta \leftarrow \theta - \nabla_\theta\, \mathcal{L}_{\mathrm{GTRL}}(Q_\theta)$ ▷ Equation 10
  $\phi \leftarrow \phi + \nabla_\phi\, J(\phi)$ ▷ Equation 4
  $\bar{\theta} \leftarrow \eta\, \theta + (1 - \eta)\, \bar{\theta}$
end for

update keeps the larger of the two. A non-decomposable goal has no subgoal, so it carries the one-step target alone. The expectile and the hindsight weight follow the same mark. Critic loss changes from Transitive RL (Park et al., 2026). The critic, the target network, and the policy are kept as they were.

D FULL RESULTS

Table 6 reports the per-goal results behind Table 2. Every task carries five evaluation goals, and the *overall* row of each block is their average.

Table 6: **Full results on standard deterministic tasks.** We report success rate (%) and standard deviation over four seeds, for each of the five evaluation goals of the ten deterministic OGBench (Park et al., 2025a) tasks. Experiments used *oracle-rep* variants. Best results are marked **RED**.

Environment	Task	BC	FBC	IVL	IQL	TD-n	CRL	MC	QRL	TDP	COE	TRL	GTRL
pointmaze-large-navigate	task1	53 ±24	98 ±3	94 ±6	92 ±9	74 ±13	23 ±18	13 ±12	31 ±28	40 ±19	67 ±26	52 ±19	100 ±1
	task2	3 ±4	17 ±6	0 ±0	0 ±0	0 ±0	42 ±28	0 ±0	5 ±6	0 ±0	0 ±0	1 ±2	57 ±35
	task3	17 ±9	76 ±11	100 ±0	76 ±13	40 ±13	67 ±14	3 ±4	0 ±0	4 ±4	12 ±8	7 ±5	100 ±0
	task4	23 ±10	79 ±8	3 ±5	0 ±0	9 ±9	13 ±18	3 ±6	1 ±2	56 ±12	44 ±27	41 ±13	42 ±15
	task5	30 ±22	87 ±5	44 ±39	0 ±0	31 ±8	21 ±8	0 ±0	0 ±0	49 ±16	48 ±16	64 ±14	97 ±3
	overall	25 ±3	71 ±4	48 ±10	34 ±3	31 ±7	33 ±7	4 ±4	7 ±7	30 ±5	34 ±7	33 ±5	79 ±6
antmaze-large-navigate	task1	3 ±3	13 ±11	11 ±13	23 ±13	63 ±16	87 ±9	30 ±8	51 ±15	6 ±5	6 ±6	57 ±13	88 ±2
	task2	22 ±12	24 ±8	19 ±5	44 ±20	76 ±5	64 ±25	47 ±8	69 ±12	23 ±6	26 ±11	66 ±4	79 ±5
	task3	53 ±10	48 ±6	55 ±6	82 ±6	87 ±6	91 ±3	86 ±6	96 ±3	69 ±9	67 ±4	80 ±4	88 ±5
	task4	17 ±9	7 ±3	4 ±3	15 ±7	21 ±8	92 ±4	17 ±10	57 ±16	17 ±10	14 ±1	8 ±8	89 ±1
	task5	17 ±5	16 ±4	18 ±12	20 ±7	28 ±6	90 ±6	13 ±5	61 ±12	17 ±4	18 ±5	18 ±11	88 ±3
	overall	22 ±4	22 ±4	21 ±6	37 ±8	55 ±6	85 ±7	39 ±5	67 ±8	27 ±3	26 ±3	46 ±5	87 ±1
humanoidmaze-medium-navigate	task1	10 ±7	7 ±9	29 ±4	31 ±9	95 ±2	91 ±8	79 ±13	8 ±12	4 ±3	6 ±4	76 ±6	97 ±0
	task2	6 ±4	8 ±5	37 ±12	82 ±9	94 ±1	96 ±1	88 ±5	22 ±29	7 ±3	9 ±2	96 ±2	93 ±2
	task3	8 ±5	15 ±5	16 ±7	6 ±6	8 ±7	72 ±17	1 ±1	30 ±21	12 ±3	13 ±3	8 ±8	59 ±8
	task4	2 ±1	3 ±3	0 ±0	0 ±0	8 ±12	8 ±7	2 ±3	9 ±9	0 ±0	1 ±1	11 ±3	32 ±14
	task5	11 ±5	12 ±2	39 ±16	63 ±12	97 ±3	93 ±4	76 ±7	20 ±19	14 ±4	14 ±7	92 ±3	97 ±2
	overall	7 ±2	9 ±3	24 ±4	36 ±3	60 ±3	72 ±5	49 ±3	18 ±17	7 ±0	9 ±2	57 ±1	75 ±4
humanoidmaze-large-navigate	task1	1 ±1	0 ±0	9 ±3	10 ±1	34 ±14	49 ±17	6 ±5	3 ±3	0 ±0	0 ±0	9 ±6	53 ±12
	task2	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0	1 ±1	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0	4 ±4
	task3	3 ±1	2 ±1	3 ±2	11 ±6	64 ±14	63 ±25	22 ±11	9 ±6	3 ±4	4 ±3	29 ±9	85 ±5
	task4	3 ±3	1 ±1	1 ±1	2 ±1	0 ±0	9 ±8	1 ±1	0 ±0	2 ±2	4 ±2	4 ±4	42 ±14
	task5	2 ±2	2 ±1	2 ±1	0 ±0	0 ±0	20 ±17	0 ±0	1 ±1	0 ±0	2 ±2	0 ±0	22 ±22
	overall	2 ±1	1 ±1	3 ±1	5 ±1	20 ±2	28 ±5	6 ±2	3 ±2	1 ±1	2 ±0	8 ±1	41 ±9
antsoccer-arena-navigate	task1	7 ±4	30 ±4	76 ±5	86 ±6	78 ±8	52 ±11	71 ±4	20 ±5	12 ±1	6 ±5	89 ±3	88 ±3
	task2	6 ±3	24 ±7	62 ±5	92 ±4	69 ±9	36 ±4	49 ±11	21 ±3	7 ±6	11 ±2	85 ±5	80 ±5
	task3	2 ±3	14 ±6	82 ±8	87 ±5	87 ±5	49 ±10	55 ±6	13 ±6	1 ±2	4 ±3	89 ±5	90 ±4
	task4	2 ±1	11 ±7	34 ±6	59 ±2	35 ±11	13 ±7	22 ±7	3 ±3	3 ±1	2 ±1	48 ±12	49 ±6
	task5	1 ±1	6 ±3	54 ±4	63 ±7	52 ±12	22 ±8	12 ±5	8 ±3	3 ±1	3 ±1	53 ±3	63 ±7
	overall	3 ±1	17 ±2	62 ±2	77 ±3	64 ±4	35 ±5	42 ±3	13 ±1	5 ±1	5 ±2	73 ±4	74 ±1
cube-single-play	task1	7 ±9	17 ±5	87 ±3	97 ±3	91 ±5	64 ±6	98 ±2	6 ±7	4 ±3	7 ±7	98 ±2	93 ±5
	task2	8 ±6	18 ±12	93 ±4	96 ±3	96 ±4	61 ±8	99 ±1	8 ±4	5 ±4	6 ±12	97 ±3	96 ±3
	task3	9 ±6	22 ±5	93 ±2	99 ±1	98 ±1	69 ±9	98 ±2	9 ±3	1 ±2	2 ±3	99 ±1	94 ±7
	task4	6 ±2	20 ±6	85 ±7	92 ±4	93 ±2	56 ±10	97 ±1	2 ±2	3 ±2	27 ±17	93 ±6	93 ±2
	task5	5 ±5	14 ±7	82 ±4	89 ±5	94 ±3	66 ±18	98 ±3	3 ±2	0 ±0	24 ±21	87 ±7	88 ±6
	overall	7 ±2	18 ±5	88 ±2	95 ±1	95 ±2	63 ±5	98 ±1	6 ±3	3 ±1	13 ±8	95 ±2	93 ±4
cube-double-play	task1	6 ±5	17 ±8	88 ±4	92 ±6	48 ±10	77 ±8	40 ±8	7 ±4	6 ±2	1 ±2	73 ±5	80 ±4
	task2	0 ±0	1 ±1	78 ±5	84 ±7	0 ±0	42 ±9	0 ±0	0 ±0	0 ±0	0 ±0	23 ±7	47 ±4
	task3	0 ±0	0 ±0	75 ±6	85 ±2	0 ±0	39 ±10	0 ±0	0 ±0	0 ±0	0 ±0	30 ±11	38 ±4
	task4	0 ±0	1 ±2	8 ±5	12 ±9	0 ±0	1 ±1	0 ±0	0 ±0	0 ±0	0 ±0	3 ±3	2 ±1
	task5	0 ±0	2 ±2	47 ±14	45 ±8	1 ±1	17 ±4	0 ±0	0 ±0	0 ±0	0 ±0	18 ±7	19 ±2
	overall	1 ±1	4 ±2	59 ±2	64 ±4	10 ±2	35 ±3	8 ±2	1 ±1	1 ±0	0 ±0	30 ±5	37 ±1
scene-play	task1	14 ±8	60 ±6	97 ±3	99 ±1	96 ±4	71 ±13	72 ±14	19 ±8	43 ±4	32 ±8	97 ±2	96 ±5
	task2	2 ±1	13 ±3	92 ±4	96 ±2	94 ±2	14 ±4	26 ±8	2 ±2	9 ±2	2 ±2	95 ±3	96 ±2
	task3	2 ±3	21 ±9	83 ±8	89 ±8	78 ±6	33 ±9	25 ±15	1 ±1	3 ±4	2 ±3	97 ±3	92 ±2
	task4	3 ±1	19 ±8	43 ±28	13 ±11	79 ±7	12 ±3	4 ±3	6 ±3	2 ±2	3 ±1	76 ±17	81 ±12
	task5	0 ±0	3 ±4	23 ±16	10 ±9	31 ±13	2 ±2	1 ±1	1 ±1	0 ±0	0 ±0	18 ±9	57 ±5
	overall	4 ±2	23 ±2	68 ±10	61 ±3	75 ±3	26 ±5	25 ±5	6 ±2	12 ±1	8 ±2	77 ±2	84 ±4
puzzle-3x3-play	task1	4 ±4	10 ±1	4 ±2	99 ±1	99 ±1	15 ±8	93 ±12	3 ±3	5 ±1	7 ±3	99 ±1	100 ±0
	task2	1 ±2	1 ±1	2 ±2	99 ±1	99 ±1	6 ±3	90 ±10	0 ±0	1 ±1	2 ±2	99 ±1	100 ±0
	task3	1 ±1	1 ±1	2 ±1	99 ±1	99 ±1	1 ±1	69 ±15	0 ±0	0 ±0	1 ±2	100 ±0	99 ±1
	task4	0 ±0	1 ±1	1 ±1	98 ±2	99 ±1	1 ±1	67 ±15	0 ±0	2 ±3	1 ±1	98 ±1	100 ±1
	task5	1 ±1	2 ±2	1 ±1	95 ±2	99 ±1	2 ±2	83 ±14	0 ±0	1 ±1	0 ±0	99 ±1	100 ±0
	overall	1 ±1	3 ±1	2 ±1	98 ±0	99 ±0	5 ±1	81 ±12	1 ±1	2 ±0	2 ±0	99 ±0	100 ±0
puzzle-4x4-play	task1	1 ±1	1 ±1	6 ±4	33 ±14	9 ±7	1 ±1	7 ±2	0 ±0	1 ±1	0 ±0	47 ±5	76 ±8
	task2	0 ±0	1 ±1	4 ±4	0 ±0	6 ±3	0 ±0	3 ±4	0 ±0	0 ±0	1 ±1	17 ±5	13 ±6
	task3	0 ±0	1 ±1	6 ±1	58 ±10	4 ±4	1 ±1	4 ±3	0 ±0	1 ±1	0 ±0	38 ±13	66 ±9
	task4	1 ±1	1 ±1	6 ±6	22 ±5	8 ±4	1 ±1	4 ±3	0 ±0	0 ±0	0 ±0	34 ±2	65 ±9
	task5	0 ±0	0 ±0	6 ±1	29 ±9	3 ±3	0 ±0	7 ±4	0 ±0	1 ±1	0 ±0	32 ±6	65 ±9
	overall	0 ±0	1 ±0	5 ±2	28 ±4	6 ±3	0 ±0	5 ±2	0 ±0	0 ±0	0 ±0	34 ±4	57 ±8

E ADDITIONAL ABLATIONS: SPLIT EXPECTILE

Equation 7 fits a decomposable goal at $\kappa = 0.7$, which is optimistic over subgoals, and the rest at $\kappa = 0.5$, which is the mean over outcomes. We ablate the split by setting $\kappa = 0.7$ on both. Figure 6 shows a loss of 40.2 points on *humanoidmaze-large-navigate* and 25.3 points on *antmaze-large-navigate*, with smaller drops of 9.0 and 8.6 points on the stochastic tasks. Half of the batch carries the one-step target, so fitting it optimistically inflates values everywhere.

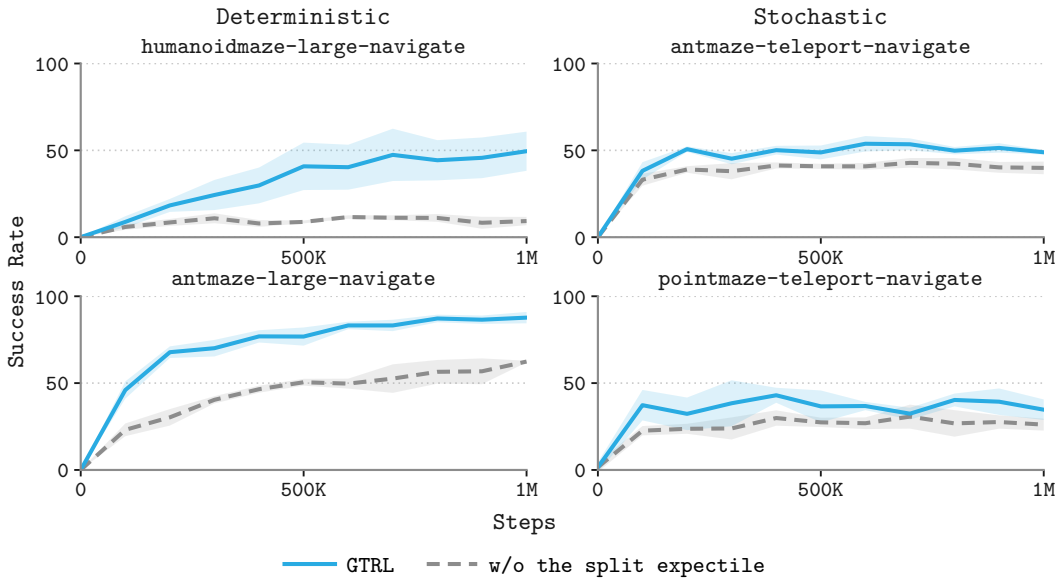


Figure 6: Ablation study on **the split expectile.**