THREE STEPS AT A TIME: LEARNING REPRESENTATIONS FROM ACTION SEQUENCES IN CONTRASTIVE RL

Anonymous authors
Paper under double-blind review

ABSTRACT

While self-supervised approaches to reinforcement learning have achieved strong results by learning representations of states and actions, a key open question is the time scale over which actions should be modeled. Departing from the standard formulation relying on single-step actions, we extend contrastive reinforcement learning (CRL), a prototypical self-supervised method, to operate over action chunks, and find that this results in large, pervasive gains across established offline and online benchmarks: $+31.7\%$ and $+93.1\%$ across 18 and 11 environments respectively. While action-chunking-driven gains are generally explained through the ability to model non-Markovian, temporally extended policies, and to propagate unbiased multi-step returns, interestingly, we find that these arguments only partially apply to CRL. Our empirical studies suggest that, in the context of CRL, an action chunk carries more information about the goal than a single action, measurably improving the critic's representations, and rendering the algorithm significantly more effective.

1 INTRODUCTION

Self-supervised reinforcement learning algorithms have gradually emerged as promising methods for learning diverse behavior with minimal supervision. Among them, contrastive reinforcement learning (CRL) (Eysenbach et al., 2022) tackles this challenge by learning representations that encode the solution to a simple classification problem: *will this goal be visited in the future, when starting from a given state and executing a single action?*

In this work, we study whether stronger critic representations can be learned by conditioning this classification problem on *sequences* of actions. In line with prior work on action chunking (Zhao et al., 2023; Li et al., 2026), we find that this minimal modification induces consistent and substantial performance gains across established benchmarks, spanning from offline manipulation to online locomotion and navigation across numerous tasks, as summarized in Figure 1.

However, we find that the standard explanations for *why* action chunking helps only partially apply to CRL, and a synergy specific to CRL is at play in this setting. On top of the ability to model non-Markovian data, and to express temporally consistent behavior (Li et al., 2025), we find that action chunking gives the critic more information about the goal than a single action, measurably improving its representations, and rendering the algorithm significantly more effective. Moreover, we find that conditioning on sequences of actions is particularly useful on noisy datasets, where individual actions are often weakly informative about future outcomes.

Finally, we observe that this phenomenon compounds with previously researched benefits of action chunking, and with the model scaling properties of CRL (Wang et al., 2026). Our results suggest that explicit modeling of action sequences should be explored beyond the supervised or temporal-difference objectives for which it was originally proposed. Our contributions are: (i) we investigate a simple extension of CRL that conditions the critic on a chunk of actions rather than a single action; (ii) we empirically validate this approach across established offline and online benchmarks, and report increases in performance of $+31.7\%$ and $+93.1\%$ respectively; (iii) we study the causes of these improvements and find that, beyond the known benefits of action chunking, it synergizes

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.

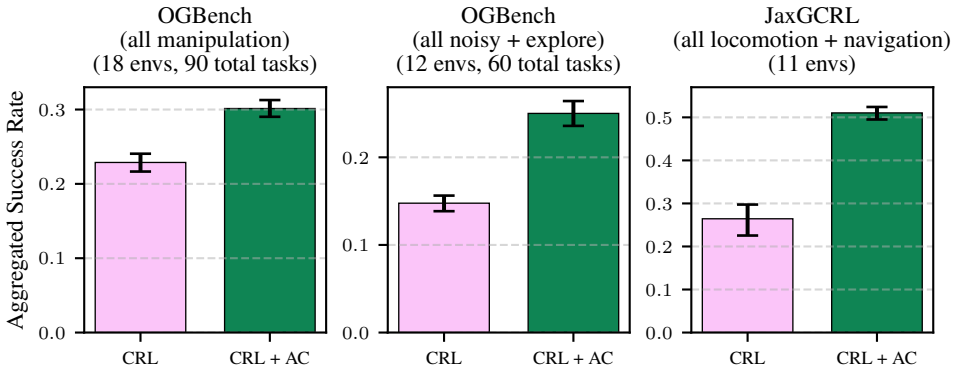

Figure 1: **Action chunking boosts goal-reaching performance.** We compare CRL + AC with a fixed action chunk length (H=3) against standard CRL across three settings, reporting 95% bootstrap confidence intervals. CRL + AC consistently outperforms CRL, with average improvements of $+31.7\%$ on the OGBench manipulation suite, $+69.4\%$ on suboptimal OGBench datasets (`noisy` and `explore`), and $+93.1\%$ on online JaxGCRL locomotion and navigation tasks.

specifically with CRL: conditioning the critic on an action chunk gives it more information about the goal than a single action, producing measurably better critic representations.

2 Related Work

Goal-conditioned RL and action representations. Our study focuses on the goal-conditioned RL (GCRL) setup in which an agent learns to achieve various tasks through conditioning on goals (Kaelbling, 1993; Schaul et al., 2015). In this setup, both the policy and the critic are conditioned on an additional goal input, such as proprioception state configuration (Ghosh et al., 2018), a visual scene representation (Nair et al., 2018), or a language instruction (Myers et al., 2026; Hermann et al., 2017; Bahdanau et al., 2019). In reinforcement learning (RL), and particularly in GCRL, representations of states and goals can be organized by their consequences for control rather than perceptual similarity. For example, actionable representations place two goals close together when they induce similar action distributions (Ghosh et al., 2018); bisimulation-based representations group states that produce equivalent rewards and transition distributions under each action (Givan et al., 2003; Ferns et al., 2012; Gelada et al., 2019; Zhang et al., 2021; Hansen-Estruch et al., 2022); and temporal-distance representations organize state–goal pairs by their reachability or the number of actions required to connect them (Dayan, 1993; Wang et al., 2023; Myers et al., 2024; Steccanella & Jonsson, 2022).

Contrastive Learning and Contrastive RL. Contrastive Learning is a widely used method to learn rich representations from unlabeled data across vision (Chen et al., 2020; Wang & Isola, 2020; Yeh et al., 2022; Oord et al., 2018) and other modalities (Radford et al., 2021; Zhai et al., 2023). In our work, we focus on temporal contrastive learning (Oord et al., 2018), specifically Contrastive Reinforcement Learning (CRL) (Eysenbach et al., 2022), which casts GCRL as a representation-learning problem in which the objective of the value function is to distinguish states likely to occur in the near future from those unlikely to occur. Recent work has shown promising results in terms of model scalability (Zheng et al., 2024; Wang et al., 2026), and sought to explain how learned representations drive exploration (Liu et al., 2025a; Bastankhah et al., 2025); we find that controlling action representations also provides significant gains. In parallel, another line of work has addressed the asymmetry of temporal distances by introducing quasimetric formulations (Myers et al., 2024; 2026). Recent methods reduce reliance on static trajectory-level context when relating state-action pairs to goals (Ziarko et al., 2025). We build on this view, arguing that what the critic learns about the goal depends not only on the agent's state but also on the informativeness of its actions.

Action Chunking. Action chunking (AC) (Zhao et al., 2023) parameterizes the policy's output as a sequence of H atomic actions from the original action space, $(a_t, \ldots, a_{t+H-1}) \in \mathcal{A}^H$, rather than as a single action. At the environment level, each control input remains an atomic action in $\mathcal{A}$. Under open-loop execution, however, the policy commits to a predicted chunk $a_{t:t+H-1} \in \mathcal{A}^H$ as a single temporally extended decision, whereas closed-loop variants execute only a couple of first actions before predicting a new chunk from the subsequent observations (Zhao et al., 2023; Liu et al., 2025b; Li et al., 2026). While the idea of temporally-extended action is not new (Randlov, 1998; Sutton et al.,

1999), the action chunking formulation has recently gained traction for mitigating control latency in real-world deployments (Black et al., 2025a;b) and as an effective way to learn from demonstration datasets with multimodal action distributions (Zhao et al., 2023; Chi et al., 2024). Beyond these practical advantages, prior work has studied how open-loop execution of action chunks can reduce the effective decision horizon (Zhao et al., 2023) and improve behavioral consistency (Zhang et al., 2025), but also might result in reduced reactivity (Liu et al., 2025b). In this work, we study action chunking in the context of GCRL, where it has been shown to effectively propagate action values, handle non-Markovian data (Li et al., 2026), and promote temporally coherent behavior (Shin et al., 2026; Li et al., 2025). As discussed later, we find that action chunking displays a special synergy with CRL, as sequences of actions carry more information about the intended goal than isolated atomic actions.

3 Background

We model the environment as a reward-free Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \mu_0, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are state and action spaces, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is a stochastic transition kernel, $\mu_0 \in \Delta(\mathcal{S})$ is an initial state distribution and $\gamma \in [0, 1)$ is a discount factor. The goal space coincides with the state space: $\mathcal{G} := \mathcal{S}$. A stationary policy is defined as a mapping from state-goal pairs to action distributions $\pi : \mathcal{S} \times \mathcal{G} \to \Delta(\mathcal{A})$. For a given policy π conditioned on goal g and starting from a state-action pair (s, a), we can model the probability of visiting future state s_f as

$$p^{\pi(\cdot|\cdot,g)}(s_f \mid s, a) := (1-\gamma) \sum_{t \geq 0} \gamma^t \mathbb{P}^{\pi(\cdot|\cdot,g)}(s_t = s_f \mid s_0 = s, a_0 = a), \tag{1}$$

where the MDP unrolls through $a_t \sim \pi(s_t, g)$ and $s_t \sim P(s_{t-1}, a_{t-1})$, and $\mathbb{P}^{\pi(\cdot|\cdot,g)}(s_t = s_f \mid s_0 = s, a_0 = a)$ denotes the t-step transition probabilities. This quantity is the *successor measure* (Dayan, 1993; Blier et al., 2021). The successor measure is equivalent to the Q-function for a goal-conditioned reward $R(s, g) = \mathbf{1}_{s=g}$:

$$Q^\pi_g(s, a) = \mathbb{E}_{\pi,P} \sum_{t \geq 0} \gamma^t R(s_t, g) = \frac{p^\pi(g|s, a)}{1-\gamma}.$$

For some goal distribution $\mu_g \in \Delta(\mathcal{G})$ we aim to find a policy that maximizes the likelihood of reaching the goal (i.e., maximizes its Q-function):

$$\arg\max_\pi \mathbb{E}_{\substack{g \sim \mu_g \\ s_0 \sim \mu_0 \\ a_0 \sim \pi}} p^{\pi(\cdot|\cdot,g)}(s_f = g \mid s_0, a_0). \tag{2}$$

CRL recasts goal-conditioned RL as a contrastive representation learning problem, training encoders $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ and $\psi : \mathcal{G} \to \mathbb{R}^d$ such that their inner product $f(s, a, g) = \phi(s, a)^\top \psi(g)$ recovers a log-probability ratio between a positive and a negative distribution. In practice, given a state s_t, positive samples are future states from the same trajectory $s_{t+\Delta}$ where Δ is drawn geometrically (thus following the successor measure p^β embedded in the dataset), and negative samples are uniformly sampled from the dataset (thus recovering the marginal of p^β). We can then train the representation to discriminate samples from the positive and negative distribution:

$$\mathcal{L}(\phi, \psi) = \mathbb{E}_{\substack{(s,a) \sim p^\beta(s,a) \\ g^+ \sim p^\beta(g|s,a) \\ g^- \sim p^\beta(g)}} \Big[\log \sigma\big(\phi(s, a)^\top \psi(g^+)\big) + \log\big(1 - \sigma\big(\phi(s, a)^\top \psi(g^-)\big)\big) \Big]. \tag{3}$$

This objective can be shown to lower bound the mutual information $I(s, a; g)$ (Eysenbach et al., 2022). At its optimum, the exponentiated dot products between learned representations are equal up to a constant to Q-values:

$$\exp(\phi^\star(s, a)^\top \psi^\star(g)) = \frac{p^\beta(g|s, a)}{p^\beta(g)} = \frac{Q^\beta_g(s, a)}{Z_g}, \tag{4}$$

where Z_g is a constant that is action independent and thus does not need to be estimated. Thus the representations can be used to rank actions at each state. A policy can be trained to maximize these dot-products with the following loss:

$$\mathcal{L}(\pi) = -\mathbb{E}_{\substack{s,g \sim p^\beta(s,g) \\ a \sim \pi(\cdot|s,g)}} \big[\phi(s, a)^\top \psi(g) \big]. \tag{5}$$

4 ACTION-CHUNKED CONTRASTIVE RL

We study a simple extension of CRL that conditions the critic on a sequence of H future actions — an action chunk — rather than a single action. We refer to it as CRL + AC.

Training Time: Action-Chunked Contrastive Objective. We condition the critic on a sequence of H consecutive actions, $a_{1:H} = (a_1, \dots, a_H) \in \mathcal{A}^H$, rather than a single action:

$$f(s, a_{1:H}, g) = \phi(s, a_{1:H})^\top \psi(g). \tag{6}$$

The positive goal g is sampled geometrically from future states along the trajectory, exactly as in standard CRL and independently of H (i.e., g need not be the state H steps ahead). The actor π is similarly modified to output a full action chunk, i.e., $\pi(s, g) = a_{1:H} \in \mathcal{A}^H$, and is trained via the same policy improvement objective as CRL:

$$\mathcal{L}(\pi) = -\mathbb{E}_{\substack{s,g\sim p^\beta(g) \\ a_{1:H}\sim\pi(\cdot|s,g)}} \big[\, f(s, a_{1:H}, g) \,\big]. \tag{7}$$

Aside from the critic's input and the actor's output, the training pipeline is largely unchanged. The replay buffer stores trajectories exactly as before; the only difference is at sampling time, where $a_{1:H}$ is assembled by concatenating H consecutive stored actions and the goal g is drawn geometrically from future states along the trajectory.

Test-time: Chunk Execution and Goal Conditioning. During inference, the policy predicts an action chunk $a_{1:H}$ but need not execute all of it before requerying. At a replanning interval $H_{\text{exec}} \leq H$, the policy executes only the first H_{exec} actions of each predicted chunk before requerying the policy: $H_{\text{exec}}{=}H$ recovers full open-loop execution (the chunk is run to completion with no intermediate replanning), while $H_{\text{exec}}{=}1$ replans at every step. Full open-loop execution ($H_{\text{exec}}{=}H$) is our default for the main results; we ablate the replanning interval in Section 5, and replan at every step ($H_{\text{exec}}{=}1$) in our analysis (Section 5.2) to isolate the effect of open-loop execution.

Implementation. The modification to standard CRL is minimal. The replay buffer stores trajectories in the standard way; action chunks and goals are constructed at sampling time — $a_{1:H}$ by concatenating H consecutive actions, and g by sampling future states along the trajectory, geometrically sampled from s. Architecturally, the critic and actor networks are the only components that change, and both remain multilayer perceptrons: the critic takes the flattened chunk $a_{1:H}$ as input instead of a single action, and the actor outputs $a_{1:H}$ instead of a single action. Algorithm 1 shows the resulting critic and actor losses, adapted from Eysenbach et al. (2022).

Since only the critic's input and the actor's output change, CRL + AC is agnostic to how the policy is extracted from the critic: any extraction procedure (e.g., DDPG-style objective (Barth-Maron et al., 2018), AWR (Peng et al., 2019), or FQL (Park et al., 2025b)) can be applied on top of the chunked critic. In our experiments (Section 5) we use FQL in the offline setting and a DDPG-style actor with entropy regularization in the online setting (Williams & Peng, 1991; Heess et al., 2015).

5 EXPERIMENTS

In this section we will first evaluate action chunking in CRL thoroughly across two settings: online and offline (Section 5.1). We then ask why chunking helps CRL specifically, identifying a representational effect beyond the standard explanations (Section 5.2). Section 5.3 ablates design choices.

Experimental setup. In the online setting, the agent collects its own experience while learning, so any effect of action chunking is mediated jointly through representation learning and exploration. In the offline setting, we evaluate CRL + AC on datasets of varying data quality, from near-expert demonstrations to noisy and exploratory data, isolating the effect of data quality on performance.

In both regimes, we keep the original algorithm and hyperparameters unchanged unless otherwise specified, modifying only the contrastive critic's input and the actor's output to operate on chunked actions. Training protocols are kept unchanged across all experiments. We report 95% stratified confidence intervals computed with RLiable (Agarwal et al., 2021). Full details on hyperparameters, evaluation protocols, detailed figure descriptions, additional ablations, and learning curves are provided in Appendices C-F. Code is available at `https://anonymous.4open.science/r/action_chunked_contrastive_rl-260B/`.

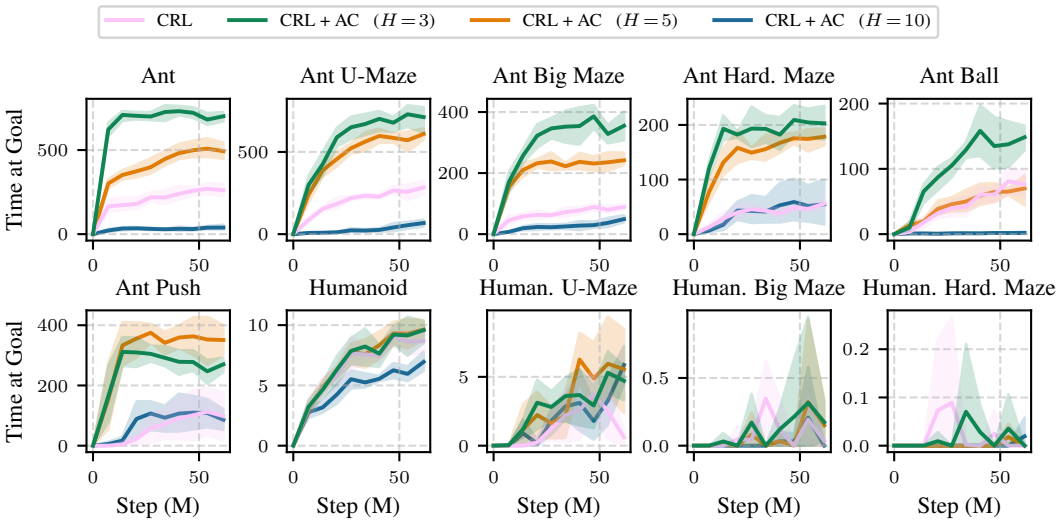

Figure 2: **Online locomotion and navigation results.** We report time-at-goal learning curves of CRL and CRL + AC with action chunk length $H \in \{3, 5, 10\}$ on 10 JaxGCRL locomotion and navigation tasks, with 95% bootstrap confidence intervals. Time-at-goal captures both task success and behavioral quality, rewarding policies that reach and stably maintain the goal (Bortkiewicz et al., 2025). Overall, CRL + AC with $H \in \{3, 5\}$ consistently and substantially outperforms CRL, with H=3 being the best or near-best chunk length across the majority of tasks, improving the performance by $+111.2\%$ on average when all 11 JaxGCRL environments are considered.

5.1 Results

Online Reinforcement Learning. We first investigate whether action chunking improves CRL in the online setting, and whether the improvement holds broadly across a diverse range of tasks. We evaluate on JaxGCRL (Bortkiewicz et al., 2025), a goal-conditioned RL framework, using its broad range of locomotion and navigation tasks with the default network architecture and training protocol. We run experiments for standard CRL and CRL + AC with action chunk lengths $H \in \{3, 5, 10, 15\}$. The actor's target entropy is scaled to the dimensionality of the action chunk, rather than that of an individual action, ensuring consistent entropy regularization under the expanded action space. We report two metrics: success rate and time-at-goal, the latter adopted from JaxGCRL (Bortkiewicz et al., 2025) and used in prior work (Wang et al., 2026).

Figure 2 demonstrates that action chunking improves performance on online locomotion and navigation tasks. Averaged across all evaluated online tasks, it increases time-at-goal by $+111.2\%$ and success rate by $+93.1\%$ (Appendix F.5). Action chunking thus improves CRL in the online setting, and the gains are consistent across the task suite rather than confined to individual environments.

Offline Reinforcement Learning. Having established that action chunking helps online, we next ask whether the benefit transfers to the offline setting, where exploration plays no role and data quality is fixed in advance, and whether it depends on that data quality. We evaluate CRL and CRL + AC (with action chunk length H=3) on OGBench (Park et al., 2025a), a standard offline goal-conditioned RL benchmark. We train on three dataset types: `play`, `noisy`, and `explore`, primarily across manipulation tasks. To better align with current state-of-the-art offline RL methods, we combine action chunking with a flow matching policy (Lipman et al., 2022) and FQL (Park et al., 2025b) policy extraction. Since FQL is sensitive to the α parameter, which balances Q-improvement and behavioral cloning losses, we sweep $\alpha \in \{1, 3, 10\}$ for each environment and for both CRL and CRL + AC, selecting the best-performing value. We additionally compare a smaller-than-usual discount factor of $\gamma = 0.95$ against the standard $\gamma = 0.99$ for both CRL and CRL + AC, and find $\gamma = 0.95$ improves both; we therefore adopt it throughout all offline experiments (Appendix E.5).

Table 1 presents the results. Action chunking yields consistent improvements across all settings: an average of +31.7% on manipulation tasks overall, with gains increasing to +69.4% on the suboptimal `noisy` and `explore` dataset variants. These results confirm that the benefit of action chunking is not limited to the online setting. Performance on `play` environments remains on par with standard CRL, showing that the gains on noisy data do not come at the cost of performance on clean data.

OGBench Environment	CRL	CRL + AC	GCBC	GCIVL	GCIQL	QRL	HIQL
cube-single-play-v0	74 ±4	**81** ±3	6 ±2	53 ±4	68 ±6	5 ±1	15 ±3
cube-double-play-v0	**29** ±4	25 ±5	1 ±1	36 ±3	40 ±5	1 ±0	6 ±2
cube-triple-play-v0	**12** ±8	**5** ±2	1 ±1	1 ±0	3 ±1	0 ±0	3 ±1
cube-quadruple-play-v0	**0** ±0	**0** ±0	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0
Cube (play avg.)	**29** ±2	**28** ±1	2 ±0	22 ±1	28 ±1	1 ±0	6 ±1
cube-single-noisy-v0	**86** ±4	74 ±3	8 ±3	71 ±9	99 ±1	25 ±6	41 ±6
cube-double-noisy-v0	10 ±1	**25** ±3	1 ±1	14 ±3	23 ±3	3 ±1	2 ±1
cube-triple-noisy-v0	**2** ±1	**3** ±2	1 ±1	9 ±1	2 ±1	1 ±0	2 ±1
cube-quadruple-noisy-v0	**0** ±0	**0** ±0	0 ±0	0 ±0	0 ±0	0 ±0	0 ±0
Cube (noisy avg.)	**24** ±1	**26** ±1	2 ±1	23 ±2	31 ±0	7 ±1	11 ±1
puzzle-3x3-play-v0	**28** ±3	21 ±7	2 ±0	6 ±1	95 ±1	1 ±0	12 ±2
puzzle-4x4-play-v0	58 ±5	**62** ±3	0 ±0	13 ±2	26 ±3	0 ±0	7 ±2
puzzle-4x5-play-v0	18 ±2	**19** ±0	0 ±0	7 ±1	14 ±1	0 ±0	4 ±1
puzzle-4x6-play-v0	**16** ±1	15 ±1	0 ±0	10 ±2	12 ±1	0 ±0	3 ±1
Puzzle (play avg.)	**29** ±4	**28** ±4	0 ±0	9 ±0	37 ±0	0 ±0	7 ±0
puzzle-3x3-noisy-v0	35 ±3	**56** ±3	1 ±0	42 ±19	94 ±3	0 ±0	51 ±11
puzzle-4x4-noisy-v0	1 ±1	**42** ±3	0 ±0	20 ±3	29 ±7	0 ±0	16 ±4
puzzle-4x5-noisy-v0	4 ±1	**16** ±1	0 ±0	19 ±0	19 ±0	0 ±0	5 ±1
puzzle-4x6-noisy-v0	4 ±2	**15** ±2	0 ±0	17 ±2	18 ±2	0 ±0	2 ±1
Puzzle (noisy avg.)	9 ±3	**34** ±2	0 ±0	23 ±3	40 ±1	0 ±0	18 ±2
scene-play-v0	33 ±9	**54** ±2	5 ±1	42 ±4	51 ±4	5 ±1	38 ±3
scene-noisy-v0	17 ±4	**30** ±3	1 ±1	26 ±5	26 ±2	9 ±2	25 ±4
Scene (avg.)	24 ±7	**44** ±6	3 ±0	34 ±2	38 ±1	7 ±1	31 ±1
antmaze-medium-explore-v0	**5** ±3	**4** ±4	2 ±1	19 ±3	13 ±2	1 ±1	37 ±10
antmaze-large-explore-v0	**0** ±0	**0** ±0	0 ±0	10 ±3	0 ±0	0 ±0	4 ±5
antmaze-teleport-explore-v0	19 ±3	**25** ±3	2 ±1	32 ±2	7 ±3	2 ±2	34 ±15
AntMaze (avg.)	**8** ±2	**9** ±4	1 ±0	20 ±1	7 ±1	1 ±0	25 ±3

Table 1: **Offline manipulation and suboptimal data results.** We summarize per-environment success rate on OGBench (Park et al., 2025a) with action chunk length H=3. CRL + AC improves over CRL by $+31.7\%$ on manipulation tasks and $+69.4\%$ on noisy and exploratory dataset variants, without degrading performance on any task group. Gains are most pronounced on noisy datasets, particularly *Puzzle* and *Scene*. *Italic rows*: group aggregate computed via RLiable with 95% bootstrap CIs over seeds $\times$ tasks. Baselines are copied from the OGBench (Park et al., 2025a) paper for reference, with CIs obtained via parametric bootstrap from reported means and standard deviations. Bold denotes performance within the 95% CI of the best among CRL and CRL + AC.

5.2 Why does action chunking help?

Having evaluated CRL with action chunking, we now turn to interpreting the sources of the performance gains. Prior work has put forth three explanations for why action chunking has proven useful for prior RL algorithms (Li et al., 2026):

1. the ability to model non-Markovian demonstrations;
2. unbiased H-step returns for value estimation; and
3. temporally extended exploration/exploitation.

As we will show in the experiments below, the effectiveness of action chunking in CRL may not be explained by these arguments alone, and a mechanism particular to CRL is at play. Unlike most of the methods that have been reported to benefit from action chunking, CRL is strongly based on mutual information between state-action pairs and future outcomes. We thus put forth a new explanation for why action chunking might help: by conditioning the critic on a chunk of actions rather than a single action, it adds information about the goal, producing better critic representations.

The Representational Benefit. To test whether the success of CRL + AC stems solely from the explanations discussed in prior work, we construct a setting where none of them apply and ask whether the gains survive. On three offline datasets where CRL + AC clearly outperforms CRL, we distill the chunked critic into a single-action one: we replace its state-action encoder $\phi(s, a_{1:H})$ with a single-action encoder $\phi'(s, a)$ trained by expectile regression onto the chunked critic's values (DQC (Li et al., 2025); see Appendix E.1), and extract a conventional single-step policy from it. This policy outputs and executes a single action, so it cannot benefit from any of the standard explanations: *(1)* it cannot model non-Markovian behavior, *(2)* unbiased H-step returns play no role, since CRL is not a temporal difference method, and *(3)* neither exploration nor temporally extended

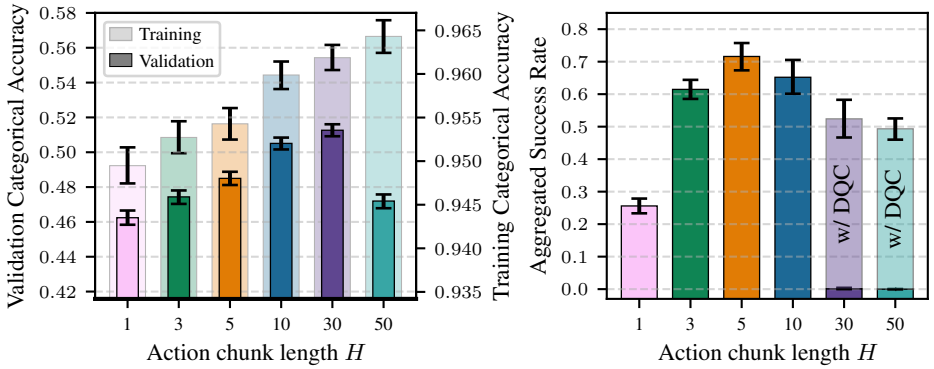

Figure 3: **Action chunking adds information about the goal; performance peaks at moderate chunk length.** Results on `cube-single-noisy-v0` (σ=1). *(Left)* Validation categorical accuracy (dark bars, left axis) increases with action chunk length H as the critic recovers more information about the goal. The thick line on the x-axis marks the state-only critic's accuracy (H=0, no action conditioning; Eq. 9), so each bar's height shows the accuracy gained over the state-only baseline. Accuracy peaks at H=30; at H=50 it drops even as *training* accuracy (light bars, left axis) continues to rise, indicating that the long-chunk critic overfits. *(Right)* Aggregated success rate peaks at a short chunk length (H=5) and collapses to near zero by H=30 and H=50. Distilling each long-chunk critic into a short-chunk (H_π=5) critic, and extracting policy from it (light bars, w/ DQC) recovers success to $\sim$50%. All methods execute a single action and replan at each step (H_{exec}=1), so the collapse does not stem from open-loop execution, but from the difficulty of extracting an action from a long-chunk critic. We provide the full per-H learning curves and the DQC training curves in Figure 6 (Appendix E.1). Error bars show 95% bootstrap confidence intervals.

exploitation applies, since the data is pre-collected and the policy executes one action at a time. All it inherits from the chunked critic are its representations. If the chunking benefit came entirely from the standard explanations, this policy should perform no better than standard CRL; if the benefit is representational, it should retain a positive gain, even if smaller than that of the full-chunk policy.

In practice, the distilled single-step policy outperforms standard CRL on all three datasets for most of training, with relative gains ranging from $\sim$30% to $\sim$100% depending on the dataset (Figure 5). Since none of the standard explanations applies here, the surviving gain must originate in the critic's representations — and because distillation can only lose information, it is a lower bound on their true benefit. This benefit is also distinct from those identified in prior work and compounds with them: the full-chunk policy further improves the distilled single-step one (Figure 6). We attribute this further gain to non-Markovian modeling, the only standard explanation still available in this setting.

Action Chunking Adds Information. Having observed the representational benefit, we ask what action chunking adds to critic representations. We hypothesize that a chunk gives the critic more information about the goal than a single action. To test this, we measure the critic's *categorical accuracy*: for a batch of B state-action/future state pairs, the critic scores all $B\times B$ pairings $\phi((s, a_{1:H})_i)^\top \psi(g_j)$, and accuracy is the fraction of states for which the true future g_i scores highest among the B candidates. Higher accuracy indicates that the critic's representation discriminates the correct future more sharply. To ensure any change reflects action information alone, we vary only the critic's action conditioning, holding the validation dataset, batch size B, and discount γ fixed, so the number of negatives and the data distribution are unchanged. Crucially, we measure the action's contribution against a state-only critic (Eq. 9): the state already carries most of the discriminative signal, so the question is what the action, and then the chunk, add on top of it.

Empirically, categorical accuracy increases monotonically with H on the validation dataset (Figure 3, Left). Relative to the state-only critic, conditioning on a single action (H=1) yields $\sim$4.5 percentage points higher accuracy, and conditioning on a chunk yields a larger gain, peaking at $\sim$9.5 percentage points at H=30, nearly double the single-action gain. These results support our hypothesis: conditioning on a chunk gives the critic more information about the goal than a single action, and the critic captures it. The exception is H=50, where validation accuracy drops even as *training* accuracy peaks. This is overfitting, not a loss of information: the critic memorizes the training data instead of generalizing. Formally, the rise in accuracy with H is consistent with the data-processing inequality: the mutual information between the state-action input and the goal can only increase with action chunk length, $I(s, a_{1:H}; g) \geq I(s, a; g)$.

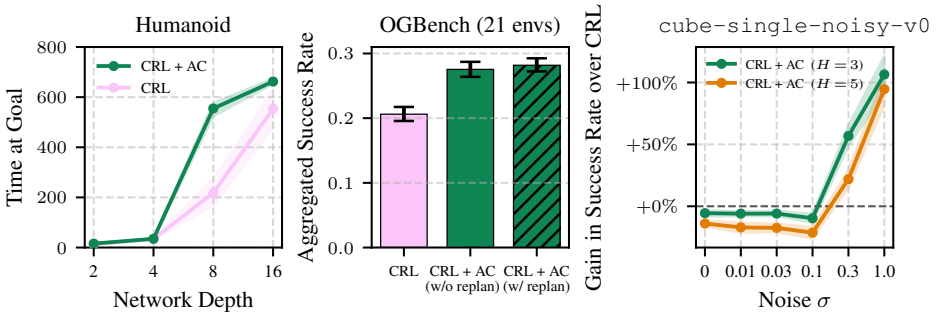

Figure 4: **Ablations on Action-Chunked CRL.** *(Left)* **Performance across network depths.** We evaluate the effect of network depth scaling on a challenging Humanoid task, for CRL and CRL + AC with action chunk length H=3. CRL + AC matches or exceeds CRL at every depth. *(Middle)* **Effect of replanning on CRL + AC.** On 21 OGBench environments, we compare CRL + AC (H=3) executing the full chunk open-loop (H_{exec}=H) against replanning every step (H_{exec}=1), alongside standard CRL. Replanning every step achieves comparable average performance to open-loop execution, though its effect varies across environments (Appendix E.4). *(Right)* **Action-Chunked CRL and noise.** We report the success rate gain of CRL + AC over CRL for action chunk lengths $H \in \{3, 5\}$ across `cube-single-noisy-v0`. We vary the noise level σ injected into expert actions during offline dataset collection, and train both CRL and CRL + AC on the resulting datasets. Gains become consistent for $\sigma \geq 0.3$, reaching up to 100% at the highest noise level.

Why Does Performance Collapse at Large H? Critic quality and policy success diverge at large H: validation categorical accuracy keeps rising with H, peaking at H=30, yet success peaks at H=5 ($\sim$72%) and falls to near zero by H=30 (Figure 3). If longer chunks yield better critics, why does performance collapse? We test whether the collapse is a failure of the critic or of policy extraction. Since execution is fixed to single-step replanning (H_{exec}=1), open-loop execution cannot be the cause, leaving policy extraction as the suspect: extracting a policy from a long-chunk critic requires the actor to model a distribution over a full length-H chunk, which grows harder with H (Li et al., 2025). If the critic is the problem, its information should be unusable; if extraction is the problem, a simpler policy should be able to recover it. We therefore apply the same DQC distillation used earlier to isolate the representational benefit, now with a moderate policy horizon (H_π=5) rather than a single action, applied to the H=30 and H=50 critics (details in Appendix E.1).

Empirically, this recovers success from near zero to $\sim$50% (Figure 3, Right, light bars). And since distillation can only lose information, never add it, then the recovered policy cannot be more informed than the original H=30 and H=50 critics. The fact that a policy extracted from the distilled short-chunk critic reaches $\sim$50% therefore shows the critics themselves held usable information; the undistilled collapse was a failure of policy extraction, not of critic quality. At large H, the bottleneck is extracting a policy from the critic, not the critic itself, and the representational benefit still holds.

5.3 Ablation Studies

Network Depth Scaling. Action chunking expands what the critic and actor can express; a natural question is whether the same benefit could be obtained simply by increasing network capacity. We therefore ablate network depth for CRL and CRL + AC, following the scaling setup of Wang et al. (2026) (details in Appendix E.3). The left panel of Figure 4 shows that on the Humanoid task the benefit of action chunking persists as network depth increases: CRL + AC matches or exceeds CRL at every depth, so its advantage compounds with capacity rather than being substituted by it. The effect is more pronounced on Ant Hardest Maze (Appendix E.3), where action chunking is markedly more compute-efficient: a 2-layer CRL + AC network matches the performance of an $8\times$ deeper (16-layer) CRL network; added depth is therefore not a substitute for action chunking.

Open-loop vs. Replanning. By default we execute chunks open-loop, which conflates the representational effect with the effect of committing to a sequence of actions; we therefore ablate the evaluation replanning on 21 OGBench environments with action chunk length H=3 (Figure 4, Middle). At evaluation we execute only the first $H_{\text{exec}} \leq H$ actions of each predicted chunk before requerying the policy (H_{exec}=H recovers full open-loop execution, H_{exec}=1 replans at every step). Replanning every step is slightly beneficial offline, though it varies across environments, whereas online it degrades performance, falling below standard CRL (Appendix E.4). Offline, open-loop execution is not the source of the gains. Online, the representational benefit and open-loop execution are entan-

gled: the latter is necessary, so we cannot cleanly isolate the former. We expect the representational benefit to be present online as well, but leave disentangling the two to future work.

Varying Noise. The offline results showed larger gains on noisy datasets; to test whether noise is the driver, we systematically vary the noise injected into expert actions during data collection on `cube-single-noisy-v0`, and then compare CRL and CRL + AC with $H \in \{3, 5\}$ (Figure 4, Right). The advantage of CRL + AC grows almost monotonically with noise. The same trend holds on `puzzle-3x3-noisy-v0` (Appendix E.2), confirming the effect is not specific to one environment. Note that this experiment varies the dataset at a fixed chunk length, complementary to our earlier analysis. Our critic metrics (Figure 7) confirm action chunking adds information at every noise level, but this advantage stays roughly constant while the performance gap widens: noise amplifies the added information through a mechanism our metrics do not capture. Action chunking thus becomes more valuable as noise increases but the amplification mechanism remains open.

Takeaways of the Experimental Section

- **Action chunking consistently improves CRL.** CRL + AC improves over CRL across offline and online tasks ($+31.7\%$ offline manipulation, $+69.4\%$ on `noisy`/`explore`, $+111.2\%$ time-at-goal online) without degrading any task group.
- **Action chunking improves the critic by adding information.** An action chunk gives the critic more information about the goal than a single action, and the critic captures it: over a state-only critic, a chunk nearly doubles the validation accuracy gain of a single action. Distillation confirms that better critic representations contribute to the gain.
- **Critic quality and policy quality decouple at large H.** Beyond a moderate chunk length, a better critic does not mean a better policy: accuracy keeps rising to $H{=}30$ while success collapses (Figure 3). With execution held fixed, the bottleneck is *policy extraction*, not critic quality. Distilling the long-chunk critic into a short-chunk one, and then extracting a policy, recovers performance from near zero to $\sim 50\%$. The critic's information was usable all along.
- **Network scaling does not substitute for action chunking.** CRL + AC retains its advantage at every depth tested, and a shallow CRL + AC network can match a much deeper CRL one. Added depth is therefore not a substitute for action chunking.

6 Conclusions

Treating CRL as a representative self-supervised method, we study the effect of action chunking beyond the settings where it has previously been studied, and show that conditioning the critic on short action chunks yields large gains across offline and online benchmarks. Investigating the source of these gains, we find that the explanations proposed for action chunking in other settings: modeling non-Markovian behavior, multi-step value estimation, and temporally extended execution do not fully account for its effectiveness here; instead, a mechanism specific to CRL is at play: conditioning on a chunk gives the critic more information about the goal than a single action, improving its representations and, in turn, the effectiveness of the contrastive critic. We further characterize when this benefit saturates: very long chunks improve the critic but hinder policy extraction, and we show that decoupling the critic and policy horizons largely recovers performance.

While we were able to observe consistent trends and significant gains, these remain conditional on the selection of the right chunk length. In practice, we have found a mild horizon of three steps ($H{=}3$) to be generally the most beneficial; nevertheless, designing an offline procedure to automatically choose chunk lengths, possibly per state, is an important direction for future work. A natural next direction is whether the benefit extends to other temporal enrichments of the critic's input: action chunking, motivated by its established use in RL, is only one such method, and whether alternatives like conditioning on future states or past history yield similar gains remains an open question. More generally, our work suggests that action chunking synergizes particularly well with CRL, and improves performance through mechanisms beyond those previously studied. Similar phenomena may arise beyond CRL: preliminary results in Appendix E.6 show that action chunking also improves other self-supervised RL methods, motivating its broader study in self-supervised RL.

AI USE STATEMENT

In this work, we used generative AI tools for the following. Required-disclosure tasks: assisting with implementing methods, and providing feedback on experimental design and the presentation of results. All AI-assisted code was verified and tested by the authors. All experiments, including which to run, were designed and decided by the authors; the AI's role was limited to feedback and discussion, not conceiving the experiments. We have not used generative AI tools to propose or refine our research hypotheses, or to interpret our results. The research hypotheses, and the interpretation of results, are the authors' own. The remaining required-disclosure tasks, namely generating synthetic datasets, developing theoretical models or conceptual frameworks, formulating mathematical claims, providing critical ingredients for or assisting in the writing of proofs, assisting with translation, cleaning and reformatting datasets, and supporting qualitative and thematic data analysis, are not applicable to this work. Recommended-disclosure tasks: drafting and editing the manuscript to improve readability, creating and modifying scientific figures (all checked against the underlying data), brainstorming, and proposing the title and keywords. The research ideas, experimental design decisions, empirical results, and final claims are the authors' own. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including all text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We describe the experimental setup in Section 5. Complete details on hyperparameters, evaluation protocols, and additional ablation studies are provided in Appendices C–E. The code for both the online and offline experiments, along with instructions for reproducing the results, is available at https://anonymous.4open.science/r/action_chunked_contrastive_rl-260B/.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

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

Dzmitry Bahdanau, Felix Hill, Jan Leike, Edward Hughes, Arian Hosseini, Pushmeet Kohli, and Edward Grefenstette. Learning to Understand Goal Specifications by Modelling Reward, December 2019. URL http://arxiv.org/abs/1806.01946. arXiv:1806.01946 [cs.AI].

Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

Mahsa Bastankhah, Grace Liu, Dilip Arumugam, Thomas L. Griffiths, and Benjamin Eysenbach. Demystifying the Mechanisms Behind Emergent Exploration in Goal-conditioned RL, October 2025. URL http://arxiv.org/abs/2510.14129. arXiv:2510.14129 [cs.LG].

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-Time Execution of Action Chunking Flow Policies. October 2025a. URL https://openreview.net/forum?id=UkR2zO5uww.

Kevin Black, Allen Z. Ren, Michael Equi, and Sergey Levine. Training-Time Action Conditioning for Efficient Real-Time Chunking, December 2025b. URL http://arxiv.org/abs/2512.05964. arXiv:2512.05964 [cs.RO].

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

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

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PmLR, 2020.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2024. URL https://arxiv.org/abs/2303.04137.

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

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

Norman Ferns, Prakash Panangaden, and Doina Precup. Metrics for finite markov decision processes. *arXiv preprint arXiv: 1207.4114*, 2012.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2170–2179. PMLR, 2019. URL http://proceedings.mlr.press/v97/gelada19a.html.

Dibya Ghosh, Abhishek Gupta, and Sergey Levine. Learning Actionable Representations with Goal Conditioned Policies. September 2018. URL https://openreview.net/forum?id=Hye9lnCct7.

Robert Givan, Thomas Dean, and Matthew Greig. Equivalence notions and model minimization in markov decision processes. *Artificial Intelligence*, 147(1):163–223, 2003. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(02)00376-4. URL https://www.sciencedirect.com/science/article/pii/S0004370202003764.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

Philippe Hansen-Estruch, Amy Zhang, Ashvin Nair, Patrick Yin, and Sergey Levine. Bisimulation makes analogies in goal-conditioned reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8407–8426. PMLR, 2022. URL https://proceedings.mlr.press/v162/hansen-estruch22a.html.

Nicolas Heess, Gregory Wayne, David Silver, Timothy Lillicrap, Tom Erez, and Yuval Tassa. Learning continuous control policies by stochastic value gradients. *Advances in neural information processing systems*, 28, 2015.

Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, Marcus Wainwright, Chris Apps, Demis Hassabis, and Phil Blunsom. Grounded Language Learning in a Simulated 3D World, June 2017. URL http://arxiv.org/abs/1706.06551. arXiv:1706.06551 [cs.CL].

Leslie Pack Kaelbling. Learning to Achieve Goals. In *International Joint Conference on Artificial Intelligence*, 1993. URL https://api.semanticscholar.org/CorpusID:5538688.

Daniel Lawson, Adriana Hugessen, Charlotte Cloutier, Glen Berseth, and Khimya Khetarpal. Self-predictive representations for combinatorial generalization in behavioral cloning. *arXiv preprint arXiv:2506.10137*, 2025.

Qiyang Li, Seohong Park, and Sergey Levine. Decoupled Q-Chunking, December 2025. URL http://arxiv.org/abs/2512.10926. arXiv:2512.10926 [cs.LG].

Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement Learning with Action Chunking, May 2026. URL http://arxiv.org/abs/2507.07969. arXiv:2507.07969 [cs.LG].

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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

Yuejiang Liu, Jubayer Ibn Hamid, Annie Xie, Yoonho Lee, Maximilian Du, and Chelsea Finn. Bidirectional Decoding: Improving Action Chunking via Guided Test-Time Sampling, April 2025b. URL http://arxiv.org/abs/2408.17355. arXiv:2408.17355 [cs.RO].

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

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

Vivek Myers, Bill Zheng, Anca Dragan, Kuan Fang, and Sergey Levine. Temporal representation alignment: Successor features enable emergent compositionality in robot instruction following. *Advances in Neural Information Processing Systems*, 38:149934–149961, 2026.

Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual Reinforcement Learning with Imagined Goals. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/7ec69dd44416c46745f6edd947b470cd-Abstract.html.

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

Seohong Park, Tobias Kreiman, and Sergey Levine. Foundation policies with Hilbert representations, 2024. URL https://arxiv.org/abs/2402.15567.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025a.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning, 2025b. URL https://arxiv.org/abs/2502.02538.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019. URL https://arxiv.org/abs/1910.00177.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Jette Randlov. Learning Macro-Actions in Reinforcement Learning. In *Advances in Neural Information Processing Systems*, volume 11. MIT Press, 1998. URL https://proceedings.neurips.cc/paper_files/paper/1998/hash/8f19793b2671094e63a15ab883d50137-Abstract.html.

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

Yongjae Shin, Jongseong Chae, Seongmin Kim, Jongeui Park, and Youngchul Sung. Adaptive Action Chunking via Multi-Chunk Q Value Estimation, May 2026. URL http://arxiv.org/abs/2605.10044. arXiv:2605.10044 [cs.LG].

Lorenzo Steccanella and Anders Jonsson. State representation learning for goal-conditioned reinforcement learning. *ECML/PKDD*, 2022. doi: 10.48550/arXiv.2205.01965.

Richard S Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Ahmed Touati and Yann Ollivier. Learning one representation to optimize all rewards. *Advances in Neural Information Processing Systems*, 34:13–23, 2021.

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

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International conference on machine learning*, pp. 9929–9939. PMLR, 2020.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*, pp. 36411–36430. PMLR, 2023.

Ronald J Williams and Jing Peng. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. In *European conference on computer vision*, pp. 668–684. Springer, 2022.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid Loss for Language Image Pre-Training, September 2023. URL http://arxiv.org/abs/2303.15343. arXiv:2303.15343 [cs.CV].

Amy Zhang, Rowan Thomas McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL https://openreview.net/forum?id=-2FCwDKRREu.

Thomas T. Zhang, Daniel Pfrommer, Chaoyi Pan, Nikolai Matni, and Max Simchowitz. Action Chunking and Exploratory Data Collection Yield Exponential Improvements in Behavior Cloning for Continuous Control, November 2025. URL http://arxiv.org/abs/2507.09061. arXiv:2507.09061 [cs.LG].

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware, April 2023. URL http://arxiv.org/abs/2304.13705. arXiv:2304.13705 [cs.RO].

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

Alicja Ziarko, Michal Bortkiewicz, Michal Zawalski, Benjamin Eysenbach, and Piotr Milos. Contrastive Representations for Temporal Reasoning, September 2025. URL http://arxiv.org/abs/2508.13113. arXiv:2508.13113 [cs.LG].

A Extended Related Works

Self-supervised Reinforcement Learning Contrastive reinforcement learning (Eysenbach et al., 2022) is a prototypical self-supervised reinforcement learning algorithm, as it operates over datasets of trajectories with no reward labels. As such, it fits within a broader family of self-supervised algorithms for representation and reinforcement learning, which we will now briefly discuss as they may also benefit from action chunking. While CRL is contrastive and Monte-Carlo at heart, most self-supervised methods rely on either temporal-difference learning (Touati & Ollivier, 2021; Park et al., 2024), on self-prediction (Grill et al., 2020; Lawson et al., 2025) or both (Bagatella et al., 2025). In some cases, these methods are designed for zero-shot reinforcement learning (Touati & Ollivier, 2021; Park et al., 2024; Bagatella et al., 2025), i.e. for retrieving an optimal policy for arbitrary reward function; in others, they are mostly targeting representation learning (Grill et al., 2020; Lawson et al., 2025) or dynamics modeling (Maes et al., 2026). Despite their difference, these works are generally designed to be either action-independent (Lawson et al., 2025), or to operate over single-step actions; due to their close connection to CRL, it is possible that they may benefit from action chunking through known and unknown mechanisms.

B Action-Chunked Contrastive RL Details

Algorithm 1 shows the critic and actor losses, adapted from Eysenbach et al. (2022). The only modifications to standard CRL are highlighted: the state-action encoder takes a flattened action chunk of shape `(batch, H * action_dim)` rather than a single action, and the policy outputs a full chunk. Note that this algorithm serves only as a conceptual baseline. For detailed descriptions of the algorithm implementations used in this study, please refer to Section 5 and Appendices C–E.

Algorithm 1 Action-Chunked Contrastive Reinforcement Learning: the critic and actor losses.

```
from jax.numpy import einsum, eye
from optax import  sigmoid_binary_cross_entropy
def critic_loss(states, action_chunks, goals):
    # action_chunks:  (batch_dim, H * action_dim) - flattened chunk
    sa_repr = sa_encoder(states, action_chunks)  # (batch_dim, repr_dim)
    g_repr  = g_encoder(goals)                   # (batch_dim, repr_dim)
    logits  = einsum('ik,jk->ij', sa_repr, g_repr)
    return sigmoid_binary_cross_entropy(
                logits=logits, labels=eye(logits.shape[0]))
def actor_loss(states, goals):
    # policy outputs full chunk (batch_dim, H * action_dim)
    action_chunks = policy.sample(states, goal=goals)
    sa_repr = sa_encoder(states, action_chunks)  # (batch_dim, repr_dim)
    g_repr  = g_encoder(goals)                   # (batch_dim, repr_dim)
    logits  = einsum('ik,ik->i', sa_repr, g_repr)
    return -1.0 * logits
```

C Experimental Details

C.1 Hyperparameters

Hyperparameters for the offline OGBench and online JaxGCRL experiments are reported in Tables 2 and 3, respectively. We keep all hyperparameters unchanged from the original CRL implementation unless otherwise specified.

C.2 Computational Resources

All experiments were conducted on a single NVIDIA GH200. OGBench runs took approximately 3 hours per setting, i.e., with or without action chunking. JaxGCRL runs took between 15 minutes and 9 hours, depending on the experimental setting, with a total runtime of 472 hours.

Hyperparameter	Value
train_steps	1,000,000
eval_episodes	20
lr	3×10^{-4}
batch_size	1024
actor_hidden_dims	6×512
value_hidden_dims	6×512
latent_dim	512
layer_norm	True
discount	0.95
actor_loss	FQL
alpha	$\{1, 3, 10\}$ (tuned per env.)
num_flow_steps	10
action_chunk_length	depends on experiment
replanning_interval	full action chunk length
contrastive_loss	Sigmoid BCE
value_geom_sample	True
value_p_trajgoal	1.0
num_seeds	3

Table 2: **OGBench Offline CRL + AC Hyperparameters.** Default values used across all experiments unless otherwise specified.

Hyperparameter	Value
num_timesteps	60,000,000
max_replay_size	10,000
min_replay_size	1,000
episode_length	1,000
unroll_length	62
discount	0.99
num_envs	512
batch_size	256
lr	3×10^{-4}
contrastive_loss	InfoNCE
energy_function	L2
logsumexp_penalty	0.1
network_depth	2
network_width	256
representation_dim	64
action_chunk_length	depends on experiment
replanning_interval	full action chunk length
num_seeds	5

Table 3: **JaxGCRL Online CRL + AC Hyperparameters.** Default values used across all experiments unless otherwise specified.

C.3 FIGURES

Figure 1. This figure summarizes the main results as three grouped aggregates. CRL + AC uses action chunk length H=3. The left panel (*OGBench manipulation*) aggregates the 18 manipulation environments (90 tasks); the middle panel (*OGBench noisy + explore*) aggregates the 12 noisy/explore environments (60 tasks); the right panel (*JaxGCRL*) aggregates 11 online locomotion and navigation environments. The per-environment values underlying the two OGBench panels are the same runs reported in Table 1. We exclude the visual (pixel-observation) OGBench environments, evaluating only on state-based observations. For the offline (OGBench) panels, CRL and CRL + AC are trained with a flow-matching policy (Lipman et al., 2022) and FQL (Park et al., 2025b) policy extraction, with γ=0.95 and the per-environment α selected as described in Section 5. For the online (JaxGCRL) panel, we use the default JaxGCRL architecture, training protocol, and DDPG-style policy extraction. All hyperparameters are listed in Appendix C.1, and are identical for CRL and CRL + AC except for the chunked critic input and actor output. Aggregated success rate and 95% confidence intervals are computed with RLiable (Agarwal et al., 2021) via stratified bootstrap over seeds $\times$ tasks (N=3 seeds per OGBench environment and N=5 seeds per JaxGCRL environment). For online tasks, success rate is measured at the end of training. Reported group improvements (31.7%, 69.4%, 93.1%) are relative gains of CRL + AC over CRL on each aggregate. At evaluation, the policy executes the full predicted action chunk open-loop before requerying (H_{exec}=H); no intermediate replanning is used. The corresponding per-environment learning curves are provided in Appendix F.1 (offline) and Appendix F.5 (online).

Figure 2. This Figure demonstrates the learning curves for 10 JaxGCRL locomotion and navigation environments. We report average time-at-goal and 95% confidence intervals for standard CRL and CRL + AC with action chunk length $H \in \{3, 5, 10\}$. Compared with success rate, time-at-goal provides a more informative measure of policy quality, as it rewards agents that both reach the goal quickly and remain there consistently (Bortkiewicz et al., 2025). We use the default JaxGCRL architecture, training protocol, and DDPG-style policy extraction. All hyperparameters are listed in Appendix C.1. Overall, CRL + AC with H=3 achieves the strongest performance, yielding an improvement of +111.2% over standard CRL on average. CRL + AC with H=5 also outperforms CRL in most environments, although it rarely exceeds the performance of CRL + AC with H=3. In contrast, CRL + AC with H=10 is frequently the weakest-performing variant among the evaluated methods. Additional learning curves for the JaxGCRL environments are reported in Appendix F.5.

Table 1. All runs use action chunk length H=3 and execute the full chunk open-loop at evaluation (H_{exec}=H). CRL and CRL + AC both use a flow-matching policy with FQL policy extraction,

γ=0.95, and per-environment $\alpha \in \{1, 3, 10\}$ selected on validation; the two methods are identical except for the chunked critic input and actor output. We report success rate at the end of training, averaged over the environment's evaluation goals and N=3 seeds; group aggregates (italic rows) and 95% CIs are computed with RLiable via stratified bootstrap over seeds $\times$ tasks. The mentioned gains are relative improvements of the RLiable aggregate: 32% over manipulation (all `play`/`noisy` cube, puzzle, and scene groups) and 69% over the suboptimal `noisy`/`explore` variants. Grey columns are baselines reported by Park et al. (2025a) for reference only, not re-run here, with CIs obtained by parametric bootstrap from their published means and standard deviations; these are computed differently from our seed×task bootstrap and are therefore not directly comparable in width. Bold marks entries within the 95% CI of the better of CRL and CRL + AC (both bold when tied). Overall, the group average improves, and gains concentrate on the noisier datasets, consistent with the ablation (Appendix E.2). We provide the full learning curves in Appendix F.1.

Figure 3. This experiment uses `cube-single-noisy-v0` with a high injected noise level (σ=1.0, versus the default 0.1) to obtain a more challenging dataset. All methods execute a single action and replan at every step (H_{exec}=1), so execution is held fixed across chunk lengths and cannot explain the differences. We use α=1.0 for all runs, the best value for CRL and for CRL + AC at $H \in \{3, 5\}$ on this dataset. DQC is not tuned: we set the expectile parameter κ=0.9 and distill into a policy critic of horizon H_π=5, chosen because H=5 was the best-performing chunk length for CRL + AC. Categorical accuracy is reported on held-out (validation) and training data; all other settings match the offline OGBench configuration (Appendix C.1). We expand on the DQC diagnostic and provide full learning curves in Appendix E.1.

Figure 4. This Figure summarizes three ablations that probe when and why action chunking helps; each is a condensed view of a fuller study reported in Appendix E. The left panel scales network depth and shows that action chunking's benefit compounds with capacity rather than being substituted by it (results for all tested environments are included in Appendix E.3, while the learning curves for this ablation study are reported in Appendix F.7). The middle panel compares open-loop chunk execution (H_{exec}=H) against replanning every step (H_{exec}=1), showing that replanning matches open-loop execution on average in the offline setting (Appendix E.4). We summarize each below. The right panel varies the noise injected into the offline data and shows that the gain of CRL + AC over CRL grows with noise (full sweep, additional environments, and the corresponding critic metrics in Appendix E.2).

C.4 Reproducibility

We provide all the experimental details, hyperparameters, evaluation protocols, and release the code to ensure reproducibility of the results.

D Evaluation Protocols

D.1 OGBench Evaluation Protocol

We evaluate the policy every 50,000 training steps by rolling out 20 episodes across 5 tasks per environment (OGBench default), totaling 20 evaluation checkpoints over 1,000,000 training steps. We report the mean success rate aggregated over tasks, averaged over 3 seeds with 95% bootstrap confidence intervals computed via RLiable (Agarwal et al., 2021). All runs and seeds are included in the reported results.

D.2 JaxGCRL Evaluation Protocol

We evaluate the policy 65 times throughout training, including an initial evaluation before training begins, with evaluations spaced uniformly across the training run. For readability, the reported learning curves display only 10 evaluation points. We report the mean success rate aggregated over tasks, averaged over 5 seeds with 95% bootstrap confidence intervals computed via RLiable. All runs and seeds are included in the reported results.

D.3 Benchmarks

We provide a list of all environments we used in the conducted experiments.

OGBench environments list:

- `cube-single-play-v0`
- `cube-double-play-v0`
- `cube-triple-play-v0`
- `cube-quadruple-play-v0`
- `cube-single-noisy-v0`
- `cube-double-noisy-v0`
- `cube-triple-noisy-v0`
- `cube-quadruple-noisy-v0`
- `puzzle-3x3-play-v0`
- `puzzle-4x4-play-v0`
- `puzzle-4x5-play-v0`
- `puzzle-4x6-play-v0`
- `puzzle-3x3-noisy-v0`
- `puzzle-4x4-noisy-v0`
- `puzzle-4x5-noisy-v0`
- `puzzle-4x6-noisy-v0`
- `scene-play-v0`
- `scene-noisy-v0`
- `antmaze-medium-explore-v0`
- `antmaze-large-explore-v0`
- `antmaze-teleport-explore-v0`

JaxGCRL environments list:

- Ant
- Ant U-Maze
- Ant Big Maze
- Ant Hardest Maze
- Ant Ball
- Ant Push
- Humanoid
- Humanoid U-Maze
- Humanoid Big Maze
- Humanoid Hardest Maze
- Cheetah

E Additional Experiments

E.1 Analysis: Decoupling Critic and Policy Horizons (DQC)

We use DQC (Li et al., 2025) in two ways: to isolate the representational benefit of action chunking by extracting a single-action policy from a chunked critic, and as a diagnostic to determine whether the performance collapse at large H (Subsection 5.2) stems from a poorly trained critic or from the policy failing to extract from a well-trained one (Figure 6, Left). We jointly train the action-chunked critic with its standard contrastive loss and, alongside it, a short-chunk state-action encoder $\phi'(s, a_{1:H_\pi})$ trained to match the long-chunk critic's values by expectile regression:

$$\mathcal{L}_\kappa(\phi') = \mathbb{E}\Big[\ell_\kappa\Big(\mathtt{sg}\big[\phi(s, a_{1:H})^\top \psi(g)\big] - \phi'(s, a_{1:H_\pi})^\top \mathtt{sg}\big[\psi(g)\big]\Big)\Big], \tag{8}$$

where ℓ_κ is the expectile loss with parameter κ and $\mathtt{sg}[\cdot]$ denotes the stop-gradient, which prevents the distillation loss from affecting the long-chunk critic. The goal encoder ψ is shared and the long-chunk encoder ϕ is trained by the contrastive objective; only ϕ' is trained by $\mathcal{L}_\kappa$. The policy is then extracted from ϕ' and ψ using the same procedure as our base runs. We use $(\kappa, H_\pi) = (0.9, 5)$ for the large-H recovery and $(\kappa, H_\pi) = (0.95, 1)$ for the single-action distillation; all other hyperparameters match the main results.

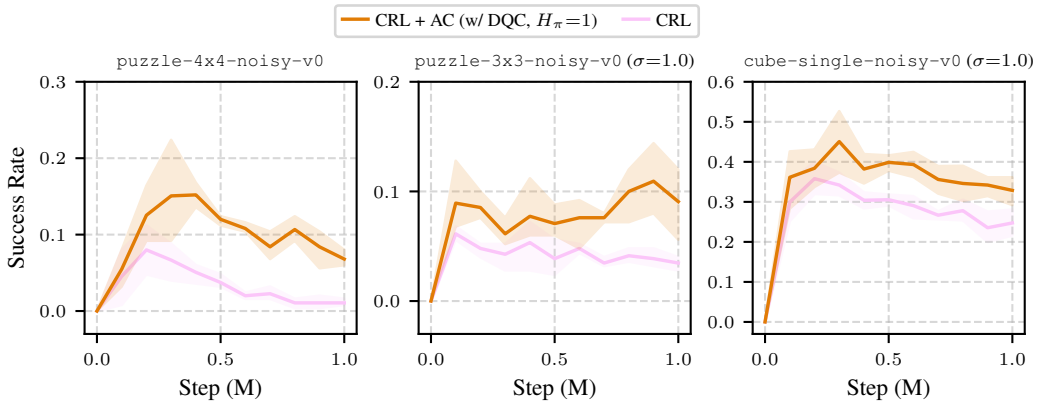

Figure 5: **The representational benefit survives distillation to a single-step policy.** On three datasets where CRL + AC performs well (`puzzle-4x4-noisy`, `puzzle-3x3-noisy` at σ=1.0, and `cube-single-noisy` at σ=1.0), we distill the CRL + AC with action chunk length H=5 critic into a single-action critic (H_π=1) via DQC and extract a conventional single-step policy (orange), then compare against standard CRL (pink). Despite outputting and executing only a single action, the distilled policy outperforms standard CRL throughout training on all three datasets, indicating that action chunking improves the critic's representation itself rather than acting solely through non-Markovian modeling, or temporally-extended execution/exploration. Shaded regions denote 95% bootstrap confidence intervals.

We first apply this decoupling to a well-performing chunked critic. In Figure 5, on three datasets where CRL + AC performs well we distill the H=5 critic into a single-action (H_π=1, κ=0.95) critic and extract a conventional one-step policy. This policy conditions on and executes a single action, yet outperforms standard CRL throughout training. Since the resulting policy is single-step, the gain cannot be attributed to modeling non-Markovian behavior, unbiased H-step returns for value estimation, or temporally extended execution/exploration; it therefore isolates a representational benefit of action chunking on the CRL critic itself (Subsection 5.2).

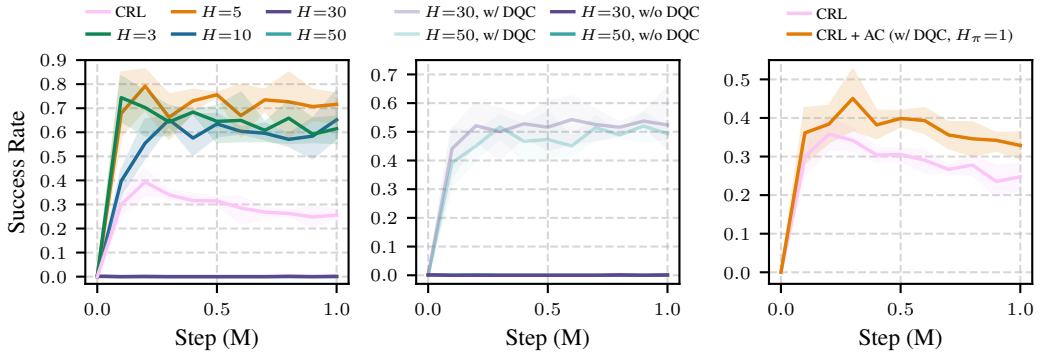

Figure 6: **Success over training across chunk lengths, and the DQC diagnostic.** Learning curves on `cube-single-noisy` (σ=1.0), executing a single action with replanning at every step (H_{exec}=1). *(Left)* Success rate for CRL and CRL + AC across chunk lengths $H \in \{3, 5, 10, 30, 50\}$. Moderate chunks improve substantially over CRL, peaking at H=5, while H=30 and H=50 remain at $\sim$0% throughout training, showing the large-H collapse is present from the start of learning rather than a late-training instability. *(Middle)* DQC recovers performance at large H: the H=30 and H=50 critics stay at $\sim$0% without DQC but reach $\sim$50% when distilled into a short-chunk (H_π=5) critic, showing the collapse is a policy-extraction failure, not a lack of critic information. Shaded regions denote 95% bootstrap CIs. *(Right)* Distilling the H=5 critic into a single-action (H_π=1) critic and extracting a one-step policy still outperforms standard CRL, despite conditioning on and executing a single action: indicating the representational benefit. The single-step policy does not fully match the chunked policy, however; the remaining gap can be attributed to the lossiness of distillation or to the chunked policy's ability to model non-Markovian behavior.

The same decoupling serves a second purpose: recovering performance from critics whose chunks are too long to extract from directly. In the middle panel of Figure 6, the undistilled H=30 and H=50 critics remain at $\sim$0% success throughout training, whereas distilling each into a short-chunk (H_π=5, κ=0.9) critic and extracting a policy recovers success to $\sim$50% in both cases. Since the same critic yields $\sim$0% when extracted directly but $\sim$50% after distillation, its information was usable all along: the undistilled failure is one of policy extraction, not critic quality.

E.2 Varying Noise Ablation

We extend the main-body noise ablation (Figure 4, Right) with two additional critic metrics across the noise sweep: validation categorical accuracy and validation contrastive loss (Figure 7). Across both environments and all noise levels, accuracy increases and loss decreases as more actions are included in the critic input: from CRL (V), which uses only the state, to CRL, to CRL + AC (H=3) and CRL + AC (H=5), confirming that action chunking adds action information. The state-only critic CRL (V) is already accurate ($\sim$57% on cube), so the action contributes only a few points; but since the policy selects actions by maximizing the critic at a fixed state, these few points are the ones that matter for performance, which is why a small accuracy gain produces a large gain in success rate. Finally, the effect of noise is environment-dependent: on cube it sharply degrades discrimination, giving CRL + AC more room to improve, whereas on puzzle the metrics stay nearly flat: the state remains informative regardless of action noise — yet chunking still improves performance.

The state-only critic CRL (V) replaces the state–action encoder $\phi(s, a)$ with a state-only encoder $\phi(s)$ in the contrastive objective, discarding the action:

$$\mathcal{L}(\phi, \psi) = \mathbb{E}_{\substack{(s,a)\sim p^\beta(s,a) \\ g^+\sim p^\beta(g|s,a) \\ g^-\sim p^\beta(g)}} \Big[\log \sigma\big(\phi(s)^\top \psi(g^+)\big) + \log\big(1 - \sigma\big(\phi(s)^\top \psi(g^-)\big)\big) \Big]. \tag{9}$$

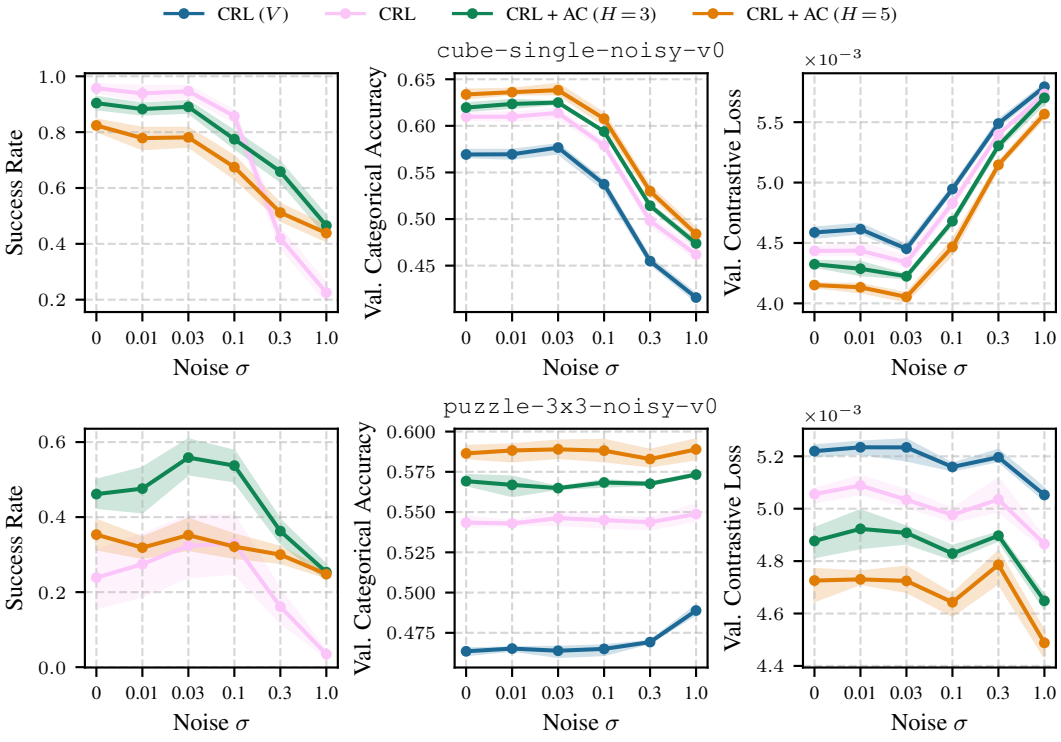

Figure 7: **Action chunking improves critic discrimination across noise levels.** Varying expert-action noise σ during offline data collection on `cube-single-noisy-v0` (top) and `puzzle-3x3-noisy-v0` (bottom). Columns: task success, validation categorical accuracy, validation contrastive loss. Across both environments and all noise levels, accuracy increases and loss decreases from CRL (V) to CRL to CRL + AC (H=3) to CRL + AC (H=5), i.e. as more actions are included: reflecting more action information. CRL (V), which uses only the state, is already accurate, so the action adds only a few points, but they are the ones that matter for performance. Noise degrades discrimination sharply on cube but barely on puzzle, where the state stays informative. Shaded regions denote 95% bootstrap CIs.

E.3 Network Depth Ablation

For the network depth ablation, we adopt the network architecture design of Wang et al. (2026), using layer normalization after every dense layer and introducing skip connections every 4 layers. We additionally increase the batch size to 512 and the training budget to 100,000,000 environment steps on Ant Hardest Maze and 400,000,000 environment steps on Humanoid. As in the main paper (Figure 4, Middle), CRL + AC uses action chunk length $H{=}3$.

Figure 8 extends the network depth ablation from the main paper by adding results on Ant Hardest Maze alongside Humanoid, sweeping network depth over $\{2, 4, 8, 16\}$. CRL + AC outperforms or matches standard CRL at every depth on both tasks. On Humanoid the advantage of CRL + AC persists as network depth increases. On Ant Hardest Maze the advantage is roughly constant across depths but yields a large compute saving: a 2-layer CRL + AC network is comparable to the $8\times$ deeper CRL network (16 layers). In both cases the benefit of action chunking compounds with network depth rather than being substituted by it. Full results, including additional chunk lengths and learning curves per-depth, are provided in Appendix F.7.

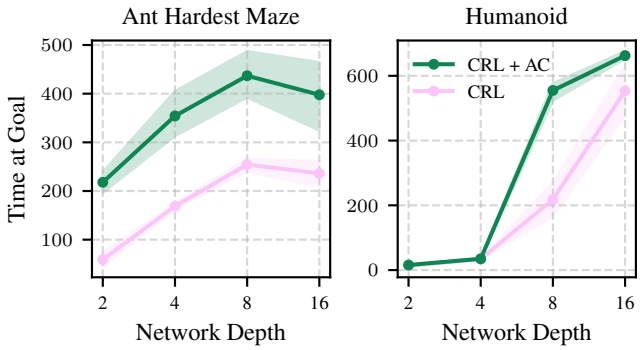

Figure 8: **Performance across network depths.** We evaluate the effect of network depth scaling on two challenging JaxGCRL tasks, Ant Hardest Maze and Humanoid, for CRL and CRL + AC with action chunk length $H = 3$. CRL + AC matches or exceeds CRL at every depth. On Ant Hardest Maze CRL + AC with a 2-layer network is comparable to CRL with a $8\times$ deeper network (16 layers), indicating better compute-efficiency. Shaded regions denote 95% bootstrap CIs.

E.4 Replanning Interval Ablation

The replanning interval H_{exec} denotes the number of actions executed before requerying the policy. We ablate this parameter in both the offline and online settings, comparing the execution of the whole action chunk ($H_{\text{exec}}=H$) with replanning after every action ($H_{\text{exec}}=1$). In the online setting, the replanning interval is shortened only during evaluation, and during experience collection, it remains equal to the action chunk length.

Figure 9 shows that reducing the replanning interval has no significant effect on the aggregated success rate in the offline setting, although the impact varies across environments (per-environment learning curves are provided in Appendix F.2). In contrast, replanning after every action in the online setting substantially degrades performance, reducing it to well below that of standard CRL (learning curves are included in Appendix F.6).

Simultaneously leveraging the improved contrastive critics learned with action chunks while recovering the policy reactivity of closed-loop execution remains an open problem in the online setting. The cause of the stark contrast between the offline and online settings is also unclear. It may arise from several differences between these regimes, including their training protocols and evaluation environments. As test-time execution is orthogonal to our main contribution, namely the representational benefit of action chunking on the contrastive critic, we leave these questions to future work.

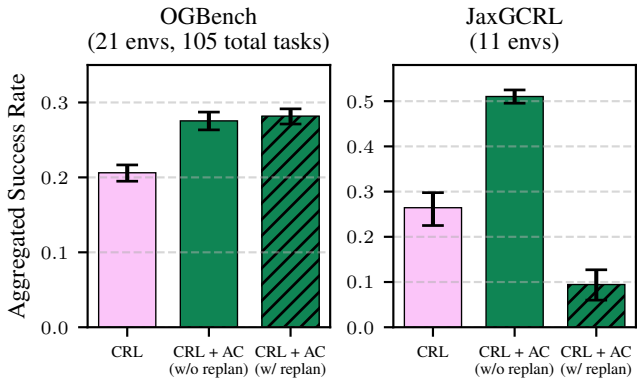

Figure 9: **Effect of replanning on action chunked CRL.** We compare the aggregated success rate of CRL + AC with replanning ($H_{\text{exec}}=1$) against CRL + AC without replanning ($H_{\text{exec}}=H$) and standard CRL across 2 benchmarks, reporting 95% bootstrap confidence intervals. The action chunk length H is set to 3 for both CRL + AC variants. Overall, replanning does not significantly affect performance on the offline OGBench benchmark, but it is detrimental to performance on the online JaxGCRL benchmark.

E.5 Discount Gamma Ablation

We ablate the discount factor γ for both CRL and CRL + AC, comparing $\gamma = 0.95$ against the standard $\gamma = 0.99$. Results in Figure 10 show that $\gamma = 0.95$ performs better across all evaluated tasks, and we therefore adopt it as the default in all offline experiments.

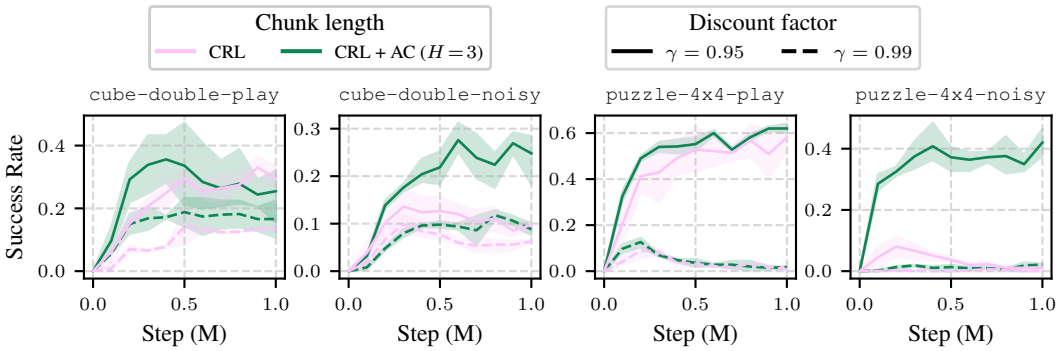

Figure 10: **Effect of discount factor γ on CRL and CRL + AC.** We compare $\gamma = 0.95$ and $\gamma = 0.99$ for both CRL ($H = 1$) and CRL + AC ($H = 3$) across four representative OGBench tasks spanning two task types (Cube, Puzzle) and two dataset types (`play`, `noisy`). For each γ, the best-performing FQL $\alpha \in \{1, 3, 10\}$ is selected per environment and method. $\gamma = 0.95$ consistently outperforms $\gamma = 0.99$ for both methods across all settings, motivating our choice of $\gamma = 0.95$ throughout all experiments.

E.6 Action Chunking in Other Self-Supervised RL Methods

This paper studies the mechanisms through which action chunking improves CRL. CRL, however, is only one instance of self-supervised reinforcement learning, a broad family of methods that learn representations and behaviors from unlabeled interaction data (see Appendix A for a discussion of related work in this area). Action chunking may benefit these other methods as well, possibly through mechanisms that differ from the one we identify in CRL and that remain to be understood.

To encourage research in this direction, we report preliminary results of incorporating action chunking into two additional self-supervised methods, TD-JEPA (Bagatella et al., 2025) and Forward-Backward (Touati & Ollivier, 2021). We find that action chunking improves both methods (Figure 11), raising success rate by $\sim$30% on TD-JEPA and $\sim$50% on Forward-Backward. We do not isolate the source of these gains here — unlike CRL, these methods are temporal-difference-based, so the multi-step-return explanation for action chunking applies and is entangled with any representational effect. Whether the representational mechanism we identify in CRL also drives the gains here is an open question, and a promising direction for future work.

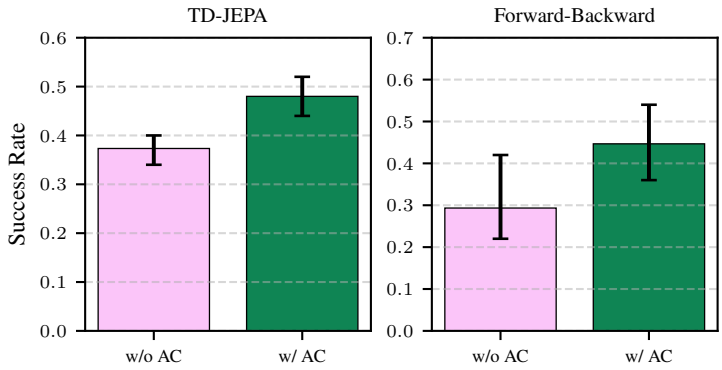

Figure 11: **Action chunking improves other self-supervised RL methods.** We report success rate with and without action chunking (action chunk length H=3, executed open-loop) for TD-JEPA and Forward-Backward on `cube-single-play-v0`. Action chunking raises success rate by $\sim$30% on TD-JEPA and $\sim$50% on Forward-Backward. These preliminary results suggest the benefit of action chunking extends beyond CRL, though we do not isolate the source of improvement in these temporal-difference-based methods. Error bars show 95% bootstrap confidence intervals.

F Learning Curves

F.1 OGBench Best FQL

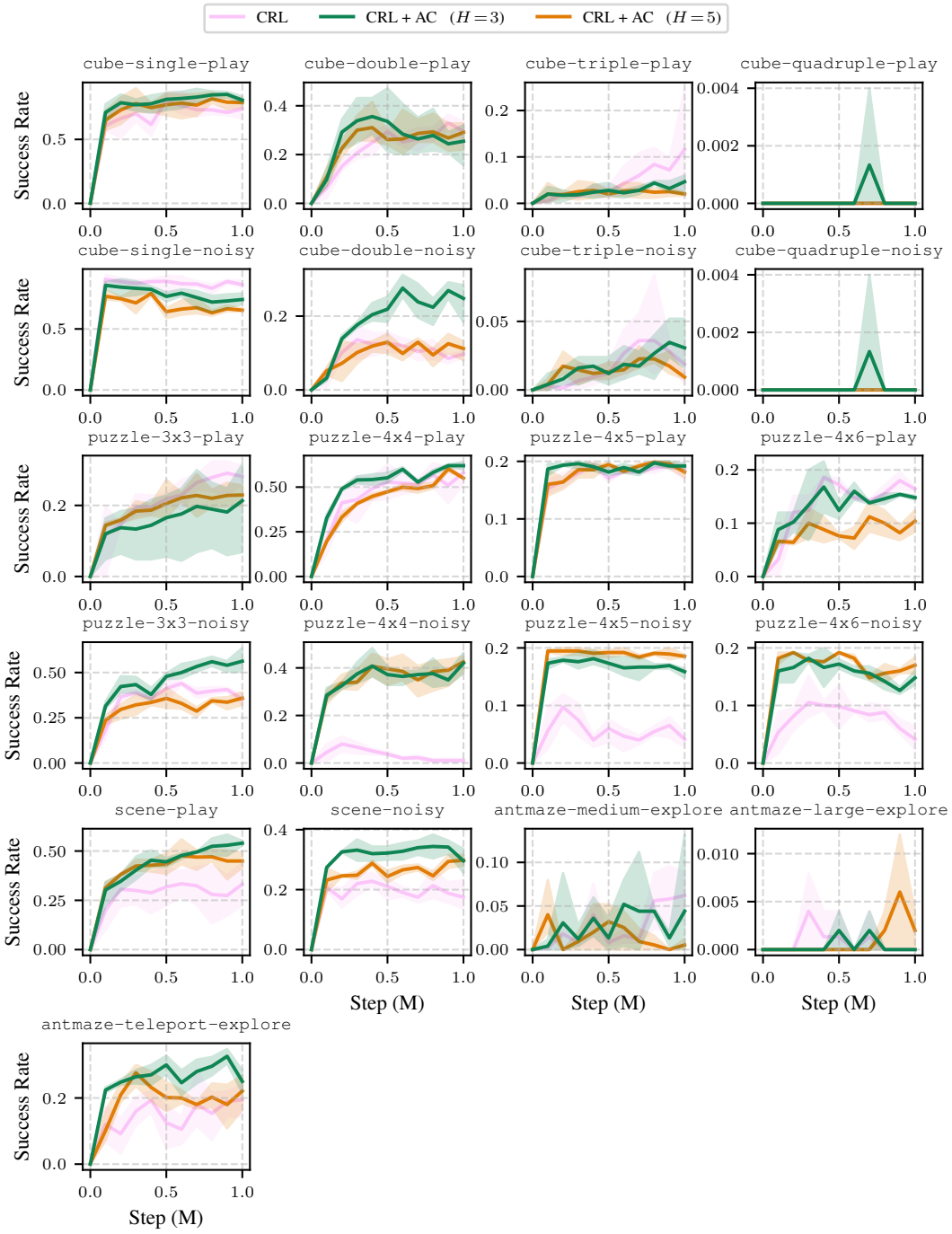

Figure 12: **Offline OGBench learning curves across all environments.** We present success rate learning curves for CRL and CRL + AC with $H \in \{1, 3, 5\}$ across 21 OGBench environments, with 95% bootstrap confidence intervals computed via RLiable. For each method and chunk length, the best-performing FQL α is selected per environment. CRL + AC consistently outperforms CRL across environments, with an average improvement of 32%.

F.2 OGBench Best FQL (Replanning)

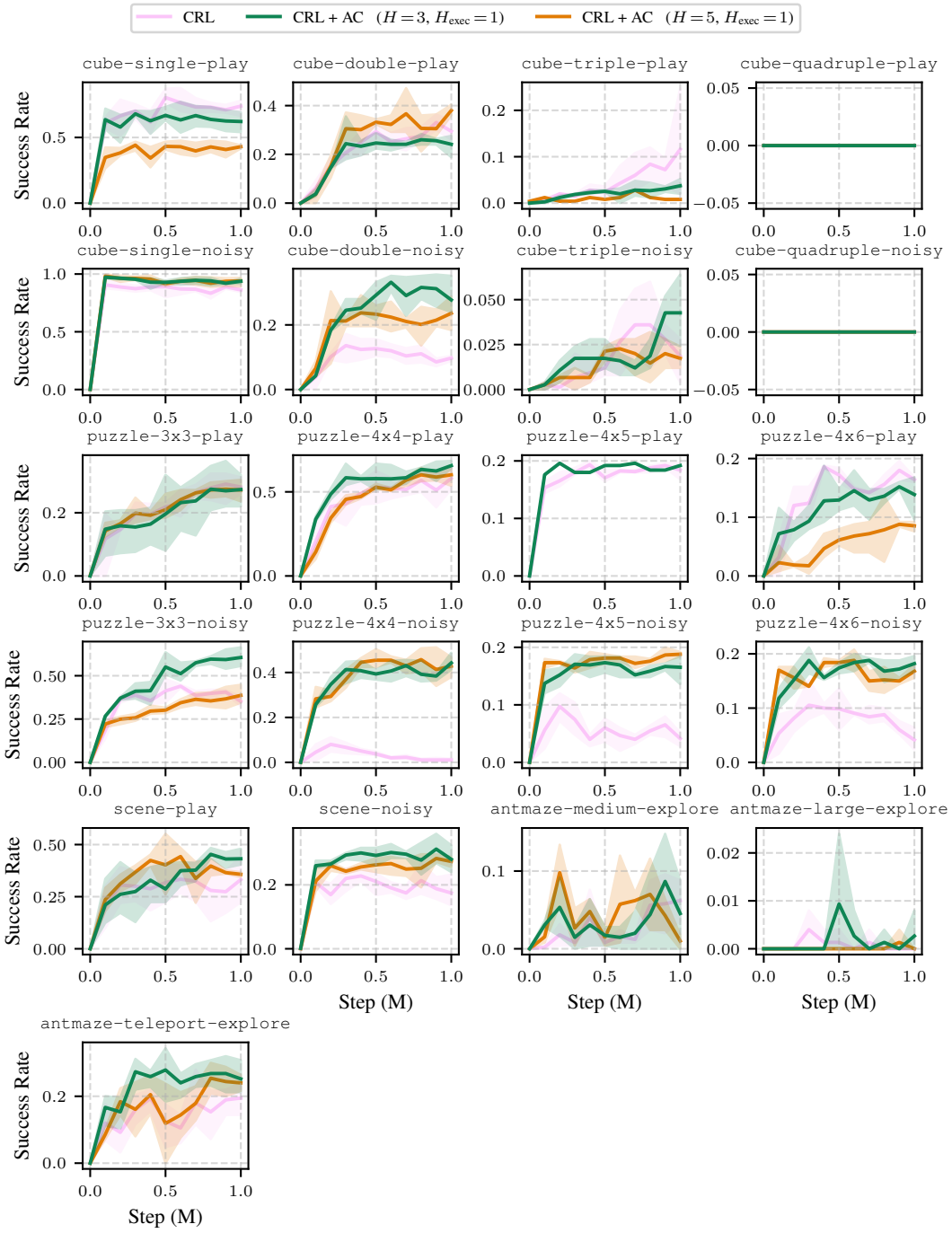

Figure 13: **Offline OGBench (Replanning) learning curves across all environments.** We present success rate learning curves for CRL and CRL + AC with $H \in \{1, 3, 5\}$ and replanning interval $H_{\text{exec}} = 1$ across 21 OGBench environments, with 95% bootstrap confidence intervals computed via RLiable. For each method and chunk length, the best-performing FQL α is selected per environment. CRL + AC with replanning outperforms CRL across many environments, with an average improvement of $\sim$30% — comparable to CRL + AC performance without replanning.

F.3 OGBench All

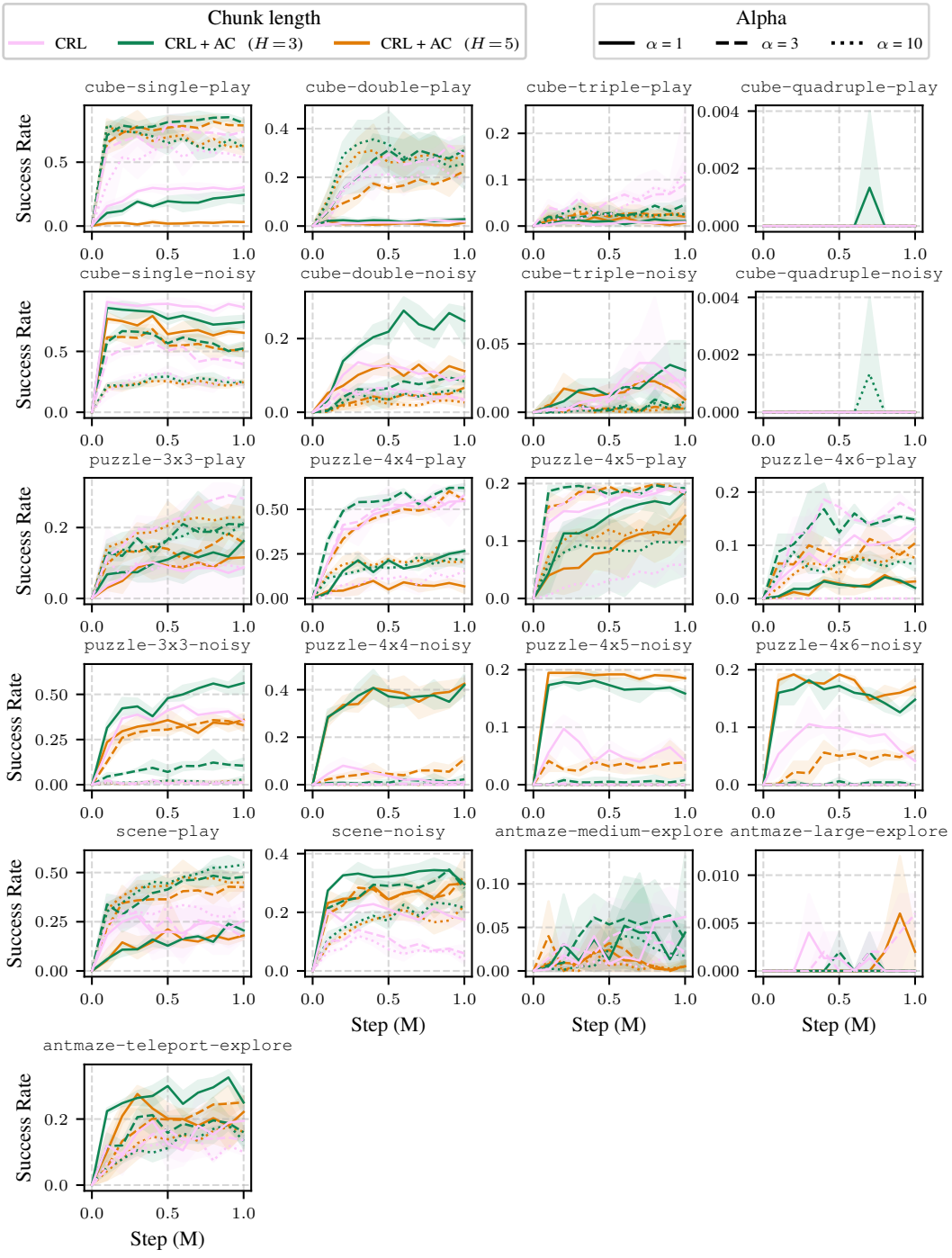

Figure 14: **Offline OGBench learning curves across all FQL α values.** We present success rate learning curves for CRL and CRL + AC with $H \in \{1, 3, 5\}$ across 21 OGBench environments for all evaluated FQL regularization strengths $\alpha \in \{1, 3, 10\}$, with 95% bootstrap confidence intervals computed via RLiable. In the main experiments we select the best-performing α per environment and method.

F.4 OGBench Varying Noise

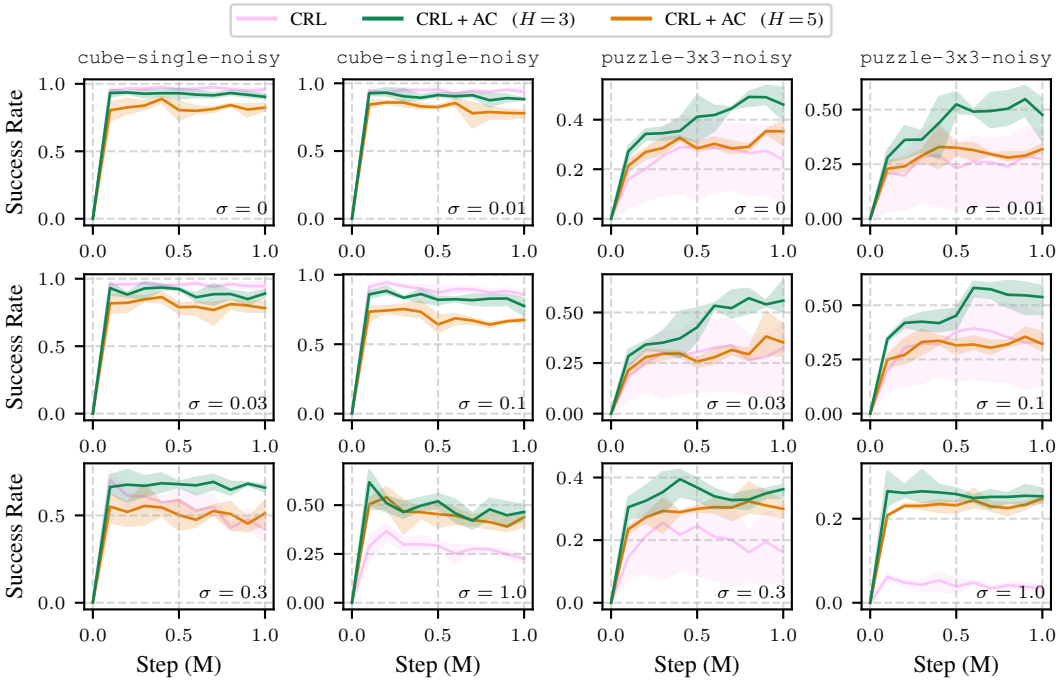

Figure 15: **Full learning curves for the varying noise experiment.** We present the complete learning curves corresponding to the experiment described in Figure 7, showing success rate over training steps for CRL and CRL + AC with $H \in \{3, 5\}$ across all noise levels σ. The performance gap between CRL + AC and CRL grows consistently with noise level, confirming that action chunking becomes increasingly beneficial as data quality degrades.

F.5 JAXGCRL

F.5.1 SUCCESS RATE

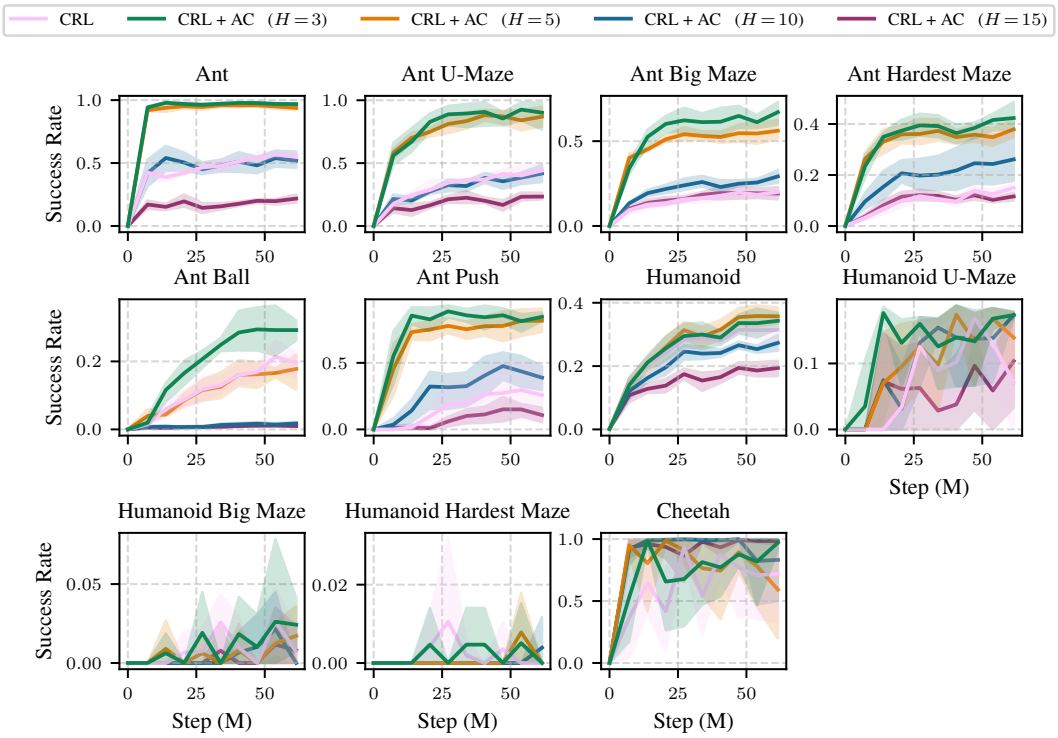

Figure 16: **Online JaxGCRL learning curves — success rate.** We present success rate learning curves for CRL and CRL + AC with $H \in \{3, 5, 10, 15\}$ across 11 JaxGCRL locomotion and navigation environments, with 95% bootstrap confidence intervals. CRL + AC with $H = 3$ is the best-performing variant overall, achieving approximately 90% improvement over standard CRL on average.

F.5.2 TIME AT GOAL

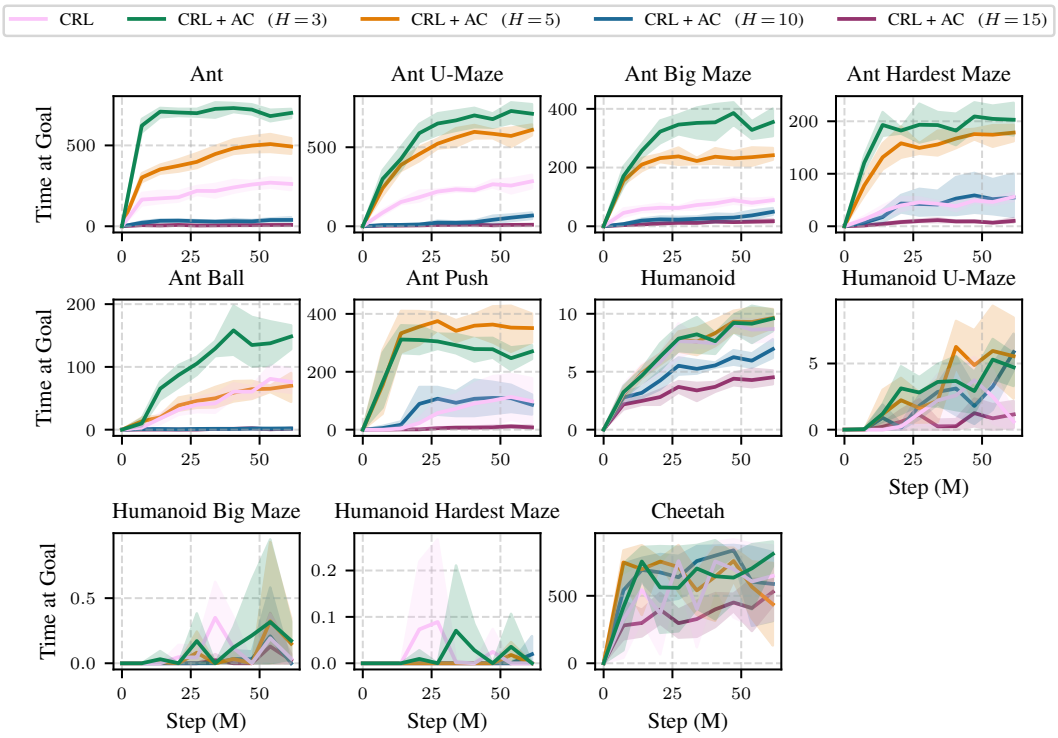

Figure 17: **Online JaxGCRL learning curves — time at goal.** We present time at goal learning curves for CRL and CRL + AC with $H \in \{3, 5, 10, 15\}$ across 11 JaxGCRL locomotion and navigation environments, with 95% bootstrap confidence intervals. Time at goal measures how long the agent spends at the goal state, rewarding both task success and behavioral stability (Bortkiewicz et al., 2025). CRL + AC with $H = 3$ is the best-performing variant, achieving $+111.2\%$ improvement over standard CRL on average.

F.6 JAXGCRL (REPLANNING)

F.6.1 SUCCESS RATE

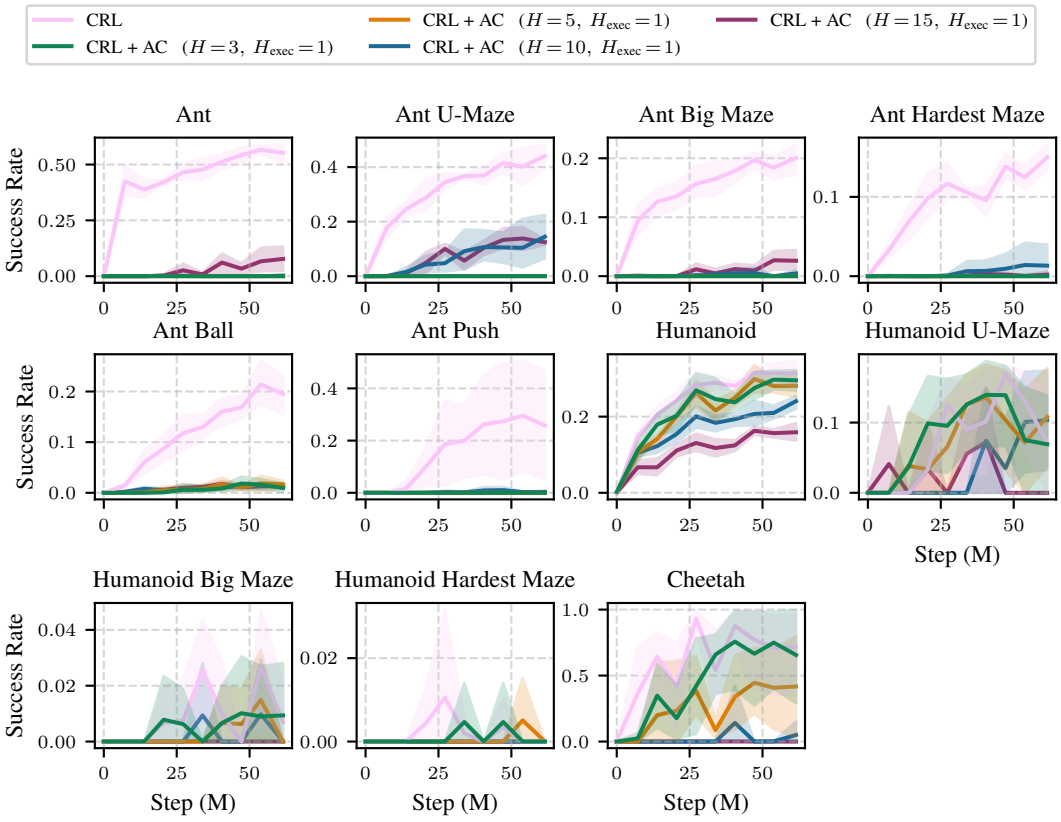

Figure 18: **Online JaxGCRL (Replanning) learning curves — success rate.** We present success rate learning curves for CRL and CRL + AC with $H \in \{3, 5, 10, 15\}$ and replanning interval H_{exec}=1 across 11 JaxGCRL locomotion and navigation environments, with 95% bootstrap confidence intervals. On these online tasks, single-step replanning (H_{exec}=1) underperforms open-loop chunk execution (Appendix E.4), in contrast to the offline setting; we leave a full explanation of this offline-online difference to future work.

F.6.2 TIME AT GOAL

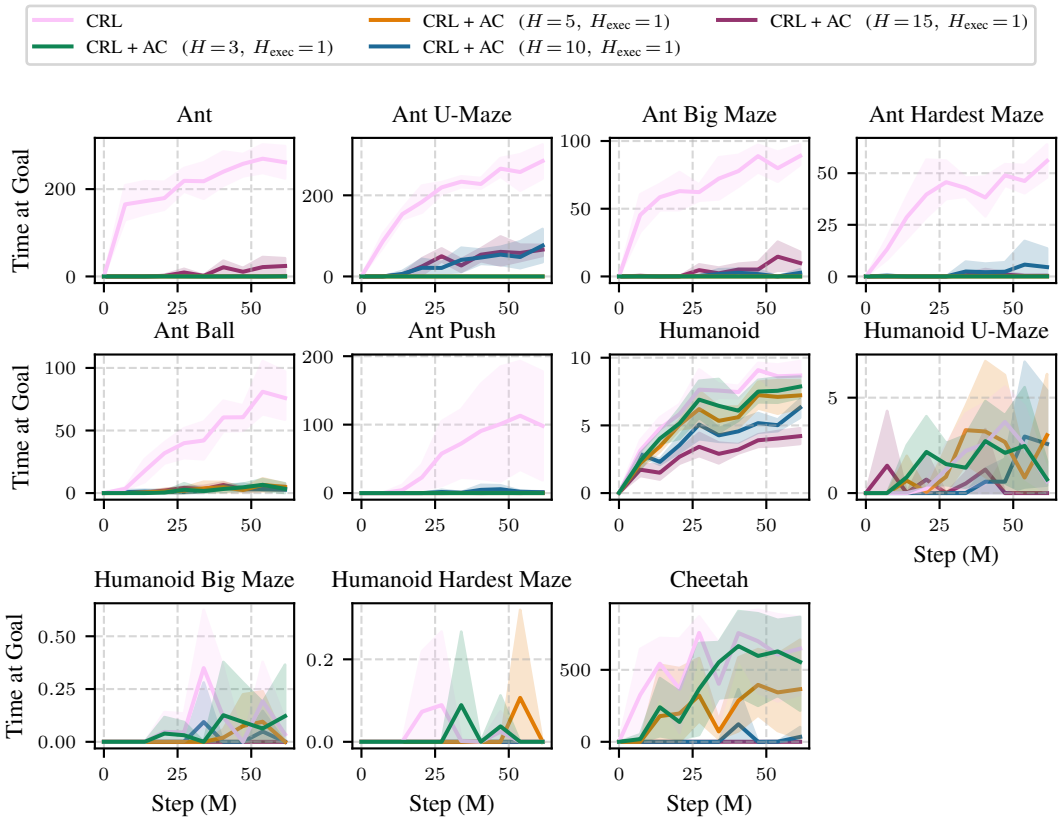

Figure 19: **Online JaxGCRL (Replanning) learning curves — time at goal.** We present time at goal learning curves for CRL and CRL + AC with $H \in \{3, 5, 10, 15\}$ and replanning interval H_{exec}=1 across 11 JaxGCRL locomotion and navigation environments, with 95% bootstrap confidence intervals. On these online tasks, single-step replanning (H_{exec}=1) underperforms open-loop chunk execution (Appendix E.4), in contrast to the offline setting; we leave a full explanation of this offline-online difference to future work.

F.7 JaxGCRL Network Depth Scaling

F.7.1 Success Rate

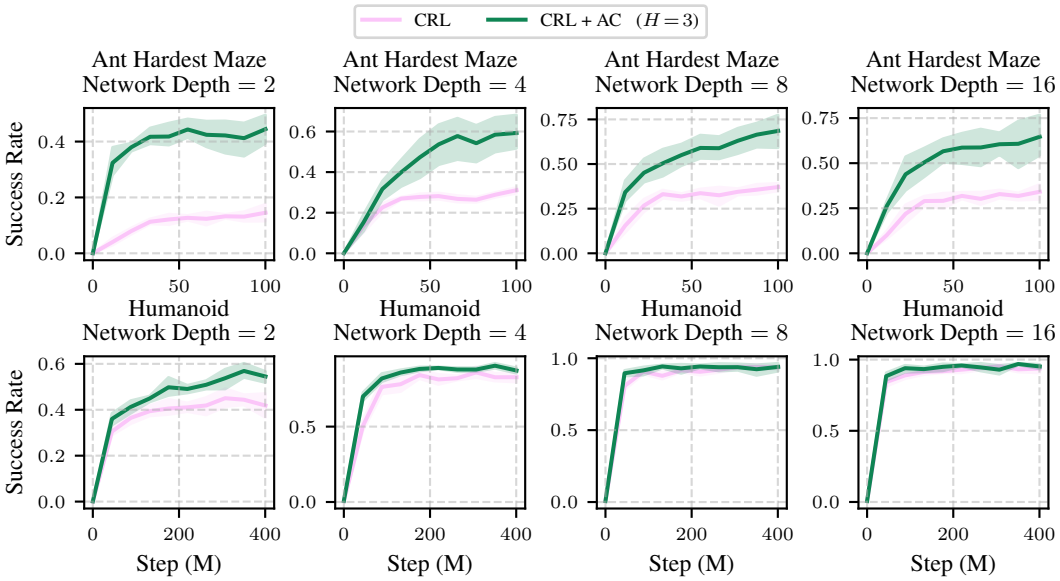

Figure 20: **Full learning curves for the network depth scaling experiment.** We present success rate learning curves for CRL and CRL + AC with action chunk length $H = 3$ across network depths on the Ant Hardest Maze and Humanoid JaxGCRL environments, with 95% bootstrap confidence intervals.

F.7.2 Time at Goal

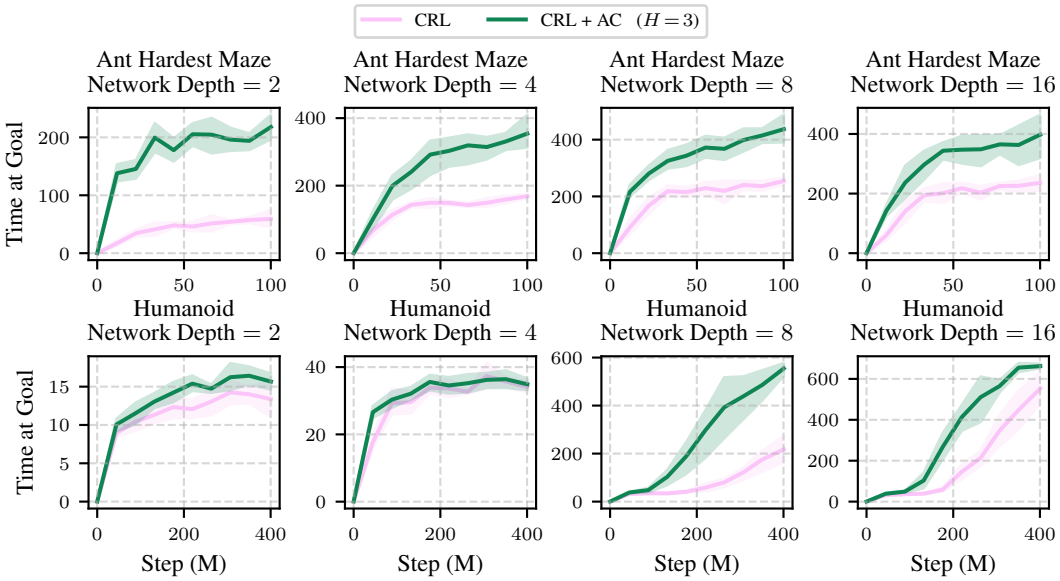

Figure 21: **Full learning curves for the network depth scaling experiment.** We present time at goal learning curves for CRL and CRL + AC with action chunk length $H = 3$ across network depths on the Ant Hardest Maze and Humanoid JaxGCRL environments, with 95% bootstrap confidence intervals.