GENERATIVE EFFICIENT GAP AMPLIFICATION FOR GOAL-CONDITIONED REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In goal-conditioned reinforcement learning (GCRL), trajectories that successfully reach the same goal can differ substantially in behavioral efficiency. Existing work on goal relabeling and generative sampling improves data reuse, but pays limited attention to how efficiently goals are reached, potentially reinforcing redundant behaviors when learning from successful experience. We propose GEAR (Generative Efficient Gap Amplification for GCRL), a generative framework that guides policy learning by contrasting efficient and redundant goal distributions. GEAR trains two diffusion-based goal generators through weighted sampling to reflect trajectory efficiency while retaining useful information from longer successful trajectories. The *efficient* generator emphasizes goals from trajectories that successfully reach the goal in fewer steps; the *redundant* generator emphasizes goals from longer successful and failed trajectories and is adversarially trained to produce plausible distractors. A dual-critic architecture evaluates the same state-action pair under both goal conditions, and the resulting value gap serves as an auxiliary contrastive signal that encourages the actor to favor actions that better support efficient goals while providing less support for redundant goals. Experiments on DMC, ManiSkill2, and Habitat show improved task performance and sample efficiency. On a physical robotic arm, GEAR achieves **90%, 75%, and 80%** success on object lifting, block stacking, and target grasping among distractors, respectively, outperforming all evaluated baselines on each task. Code is available at https://anonymous.4open.science/r/GEAR-1T66/.

1 INTRODUCTION

Goal-conditioned reinforcement learning (GCRL) (Ding et al., 2019; Eysenbach et al., 2022; Yalcinkaya et al., 2024) enables agents to learn policies conditioned on goals, and has shown effectiveness across diverse domains such as robotic control (Raj & Kos, 2024; Zhang et al., 2024), task planning (Luo et al., 2025; Wang et al., 2024b), and autonomous systems (Molinaro et al., 2024; Yang et al., 2024). By framing tasks through goal inputs, GCRL supports generalizable and efficient behavior learning (Yang et al., 2023; Jin et al., 2024). However, agents may reach goals via significantly different trajectories, some direct and purposeful, others indirect or accidental. As illustrated in Fig.1, representative GCRL methods, such as HER(Andrychowicz et al., 2017), pay limited attention to how efficiently goals are reached. This can reinforce redundant behaviors when learning from successful experience.

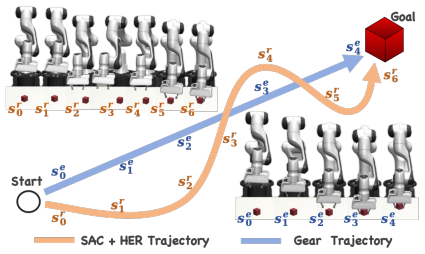

Figure 1: GEAR promotes more direct goal-reaching than SAC+HER, which often exhibits indirect and redundant behaviors

Recent efforts in goal-conditioned reinforcement learning (GCRL) have sought to improve learning performance by addressing inefficiencies in how agents reach goals. A notable method, Hindsight Experience Replay (HER) (Andrychowicz et al., 2017), enhances sample efficiency by relabeling trajectories with achieved goals, but pays limited attention to how efficiently those goals were reached. This can mislead policy updates by reinforcing suboptimal strategies. To prioritize more useful experiences, subsequent approaches adopt curriculum learning (Fang et al., 2019; Narvekar et al., 2020) or heuristic-based goal selection (Cheng et al., 2021; Feng & Zhong, 2024; Cao et al., 2025), estimating the training value of different goals. However, these methods often rely on fixed criteria

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.

or handcrafted rules and rarely model the behavioral path that led to success. As a result, they struggle to distinguish concise, purposeful behaviors from indirect or redundant ones. Generative goal sampling (Reuss et al., 2023; Yu et al., 2024b) offers a promising alternative. GOAL-GAN (Florensa et al., 2018) adapts goal difficulty to the agent's capabilities, SkewFit (Pong et al., 2020) encourages rare-goal exploration, and latent models like RIG (Nair et al., 2018) or GoPlan (Wang et al., 2024a) learn structured goal spaces. While these methods improve goal diversity, they still lack mechanisms to assess or promote the efficiency of the behaviors used to reach those goals.

Identifying effective goals from diverse goal-reaching behaviors remains a core challenge in GCRL. We introduce **GEAR** (Generative Efficient Gap Amplification for Goal-Conditioned Reinforcement Learning), a generative and contrastive framework that enhances trajectory efficiency by distinguishing purposeful behaviors from misleading ones. Instead of treating all achieved goals as equally informative, GEAR emphasizes efficient experience while retaining useful information from longer successful trajectories. GEAR comprises three key components. First, an *efficient goal generator* is trained via weighted sampling to emphasize the distribution of goals achieved along trajectories that successfully reach the goal in fewer steps. Second, a *redundant goal generator* aims to produce distractive goals that correspond to inefficiently reached targets or visually plausible distractors. To this end, it uses weighted sampling to emphasize goal distributions from longer successful trajectories and failed trajectories, and is further optimized through adversarial training. Third, a *dual-critic architecture* (Hong et al., 2024) evaluates the same state-action pair under each goal type, and their Q-value gap provides an auxiliary contrastive signal that guides the actor toward efficient behavior. *We use this gap as a learned training signal rather than a task-independent metric of trajectory efficiency.* As training progresses, the redundant generator synthesizes increasingly deceptive goals, implicitly forming a self-curriculum. This dynamic interaction can be interpreted as a *goal-space Stackelberg game* (Huang et al., 2022), where the generator acts as an adversary and the agent learns to resist misleading supervision by developing more robust and efficient policies. Although GEAR introduces additional optimization overhead, runtime analysis shows that the overhead remains small in practice. Our main contributions are as follows:

- We present **GEAR**, a generative goal-conditioned reinforcement learning framework that constructs an auxiliary efficiency-gap signal to distinguish efficient goal-reaching behaviors from misleading ones, thereby improving policy learning efficiency.
- We design a dual-generator architecture that uses weighted sampling to train the efficient generator with an emphasis on goal distributions from trajectories that successfully reach the goal in fewer steps while adversarially generating plausible distractor goals. This adversarial interplay implicitly forms an adaptive curriculum.
- We introduce a dual-critic mechanism that evaluates the same state-action behavior under efficient and redundant goal conditions. The resulting learned value gap serves as a useful contrastive signal for policy optimization.
- We empirically show that GEAR outperforms strong baselines, including HER, RIG, and QVPO, across MuJoCo-based control and visual navigation, improving task performance, sample efficiency, and training stability. GEAR also achieves the highest success rate among all evaluated methods on each of the three real-world robotic manipulation tasks.

2 PRELIMINARIES

Goal-Conditioned Reinforcement Learning (GCRL). We formalize our problem within the Goal-Conditioned Markov Decision Process (Puterman, 1990) (GCRL) framework, which augments the standard MDP with a goal space $\mathcal{G}$. It is defined by the tuple: $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R, \gamma, \mathcal{G})$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(\cdot \mid s, a)$ is the transition probability function, $\gamma \in [0, 1)$ is the discount factor, and $R(s, a, g)$ is the goal-conditioned reward function. At each timestep, the agent receives a state s_t and a goal g, and selects an action from its policy $\pi_\phi(a_t \mid s_t, g)$. The objective is to learn a policy that maximizes the expected discounted return:$\mathbb{E}_{\pi_\phi}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, g)\right]$.

Goal Relabeling. A primary challenge in GCRL is reward sparsity (Liu et al., 2023), where meaningful learning signals are rare. *Hindsight Experience Replay* (HER) (Andrychowicz et al., 2017) addresses this by relabeling failed trajectories with achieved goals, thereby transforming failures into useful learning signals. Given a trajectory $\tau = (s_0, a_0, \ldots, s_T)$, HER samples a time

index t' and defines a relabeled goal $\hat{g} = s_{t'}$. The transition (s_t, a_t, s_{t+1}) enters the replay buffer with relabeled reward $\hat{r}_t = R(s_t, a_t, \hat{g})$. While effective, standard HER treats all achieved goals as equally informative, regardless of whether they were reached via concise, purposeful behaviors or through indirect and inefficient paths. This uniform treatment may reinforce suboptimal strategies.

3 METHODOLOGY

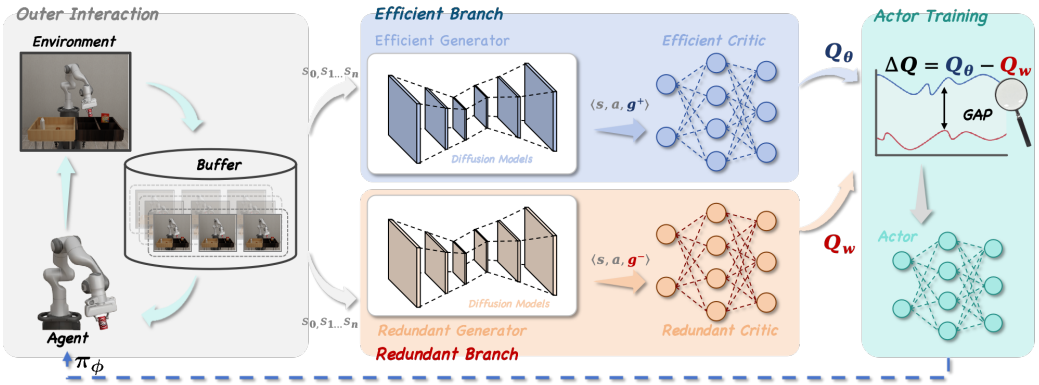

Figure 2: Overview of GEAR. Trajectories are selected from the same replay buffer according to each generator's sampling weights, and states s_n are sampled to train two goal generators: G_ψ^+ for efficient goals g^+ and G_ψ^- for redundant goals g^-. Dual critics evaluate the same transition (s, a) under both goals to compute the efficiency gap $\Delta Q(s, a)$, which serves as a contrastive learning signal to encourage efficient action selection.

We now present the GEAR framework, which transforms goal-conditioned reinforcement learning into an efficiency-aware process via a self-supervised contrastive curriculum. While conventional relabeling strategies such as HER improve sample efficiency, they fail to account for whether achieved goals stem from purposeful or inefficient behaviors, often reinforcing suboptimal strategies. In contrast, GEAR incorporates a generative mechanism to model goal-reaching quality and leverages a contrastive signal to guide the policy toward efficient behaviors.

Motivation. Our approach is driven by a key insight: excelling in goal-conditioned tasks requires understanding not only ***which*** goal was achieved, but also ***how*** it was achieved. An agent should distinguish concise and intentional trajectories from indirect, redundant, or accidental ones, yet existing methods provide limited mechanisms for making this distinction. To address this, we use the efficiency gap as an **auxiliary contrastive signal** for policy learning. Rather than ranking trajectories by absolute efficiency, the gap compares the learned value of the same state-action pair conditioned on efficient and redundant goals. A larger gap indicates that, according to the current critics, the action is more compatible with efficient goals than redundant ones, thereby providing targeted supervision for actor optimization. This signal plays two roles in GEAR. First, it prioritizes actions that are more aligned with efficient goals. Second, by contrasting them against redundant goals, it exposes the policy to increasingly informative distractors. Together, these effects induce a self-paced curriculum in which goal disambiguation becomes harder as the agent improves, enabled by the dual-generator and dual-critic architecture described next.

3.1 EFFICIENCY SPECTRUM MODELING

To support contrastive value learning, GEAR compares the value estimates of the same state-action pair under efficient and redundant goal conditions and uses the resulting value gap to guide the actor's action preferences. Maximizing this gap encourages the actor to favor actions that, according to the current critics, better support efficient goals while providing less support for redundant goals. We therefore introduce two asymmetric goal generators: an efficient generator G_ψ^+ and a redundant generator G_ψ^-, both implemented as conditional denoising diffusion probabilistic models (DDPMs) (Ho et al., 2020). G_ψ^+ uses weighted sampling to emphasize the distribution of goals from trajectories that successfully reach the goal in fewer steps, whereas G_ψ^- emphasizes the distribution of

goals from longer successful trajectories and failed trajectories and is trained adversarially to generate plausible hard negatives. With a single mixed diffusion model, adversarial updates for redundant goals could affect the modeling of efficient goals through shared parameters, even with different labels. We therefore use independent networks to prevent these updates from directly changing the efficient generator's parameters. Since both generators operate in the goal space without extra environment interaction, the added cost is limited to goal generation and diffusion updates. Table 2 supports this design through ablations.

Dataset Preparation. Using weighted sampling from the replay buffer, we construct training sets $\mathcal{D}^+$ and $\mathcal{D}^-$ for the efficient and redundant branches, respectively. For generator training, a successful trajectory $\tau = (s_0, \ldots, s_T)$ ends when the task success criterion is first met; a failed trajectory ends when data collection stops. $T(\tau) = T$ denotes its number of steps, and c_τ is 1 for success and 0 otherwise. This segmentation constructs generator training data without changing the environment's episode termination rule. To construct $\mathcal{D}^+$, we assign nonzero sampling weights only to successful trajectories, with higher weights for trajectories that successfully reach the goal in fewer steps:

$$w_\tau^+ = c_\tau \left[\delta_w + (1 - 2\delta_w)\sigma\left(\frac{T_{\max} - T(\tau)}{h} \right) \right] \tag{1}$$

To construct $\mathcal{D}^-$, we use $w_\tau^- = 1 - w_\tau^+$, assigning higher sampling weights to longer successful trajectories and failed trajectories. For failed trajectories, $w_\tau^+ = 0$ and $w_\tau^- = 1$. Here, σ is the sigmoid function, $T_{\max}$ is a reference step count, $h > 0$ controls the smoothness of the weight transition, and $0 < \delta_w < \frac{1}{2}$ is the lower bound on successful-trajectory weights in both branches. Let $p_0(\tau)$ denote the base replay sampling probability of a trajectory. Two branches sample according to:

$$p^b(\tau) = \frac{w_\tau^b p_0(\tau)}{\sum_{\tau'} w_{\tau'}^b p_0(\tau')} \tag{2}$$

where $b \in \{+, -\}$ indexes the efficient and redundant branches, respectively. After sampling trajectories according to p^+ and p^-, we pair current states with goals extracted from later states in the selected trajectories to construct $\mathcal{D}^+$ and $\mathcal{D}^-$. Both branches use the same within-trajectory sampling rule, detailed in Appendix C.4. Through weighted sampling, the two generators learn goal distributions with different emphases. Under the generated goal conditions, the dual critics construct a gap signal that helps the policy distinguish concise successful behavior from redundant or misleading behavior. The efficient goal generator G_ψ^+ and the redundant goal generator G_ψ^- are trained on the training sets $\mathcal{D}^+$ and $\mathcal{D}^-$, respectively. The former focuses on efficient-goal modeling, while the latter uses adversarial training to generate misleading but plausible goals. We next describe the training procedure for each generator.

Unified Diffusion Loss for Goal Generators. Both the efficient goal generator G_ψ^+ and the redundant goal generator G_ψ^- are implemented as conditional denoising diffusion probabilistic models (DDPMs) (Ho et al., 2020), trained with a shared mean-squared denoising loss. The training objective for each generator, indexed by $i \in \{+, -\}$, is defined as:

$$\mathcal{L}_{\text{DDPM}}^i = \mathbb{E}_{\substack{\tau \sim p^i,\ (s_t, g) \sim \rho(\cdot|\tau) \\ \epsilon \sim \mathcal{N}(0, I)}} \left[\left\| \epsilon - \epsilon_\psi^i(z_\ell, \ell \mid s_t) \right\|^2 \right] \tag{3}$$

Here, p^i is the branch sampling distribution in Eq. (2), ρ is the shared within-trajectory state–goal pairing rule, and ϵ_ψ^i is the learned noise predictor. The noisy target z_ℓ is obtained by applying forward diffusion $q(z_\ell \mid g, \ell)$ to the sampled goal g. The expectation also averages over diffusion time ℓ under the DDPM training procedure. Trajectories determine the training pairs and sampling weights, while the generator conditions on the current state. At inference time, the generator conditions on the current state s_t, and starts from Gaussian noise $z_K \sim \mathcal{N}(0, I)$. It then performs reverse denoising over K steps using ϵ_ψ^i, where K is the total number of diffusion steps, ultimately generating a synthetic goal $\hat{g} \sim p_\psi(g \mid s_t)$. The efficient goal generator G_ψ^+ learns to synthesize efficient goals in $\mathcal{D}^+$. In contrast, the redundant goal generator G_ψ^- produces redundant goals from $\mathcal{D}^-$, helping the agent distinguish between efficient and misleading strategies.

Efficient Goal Generator (G_ψ^+). Building on the unified diffusion loss, the generator G_ψ^+ is implemented as a conditional denoising diffusion probabilistic model (DDPM) trained on the training set $\mathcal{D}^+$ constructed through weighted sampling. During training, we sample a state–goal pair (s_t, g^+) from $\mathcal{D}^+$ and apply forward diffusion noise to g^+. Conditioned on s_t, the model is trained to predict the added noise by minimizing $\mathcal{L}_{G^+} = \mathcal{L}^+_{\text{DDPM}}$ (Eq. (3)), thereby learning to recover the clean goal. The weighted sampling described above makes the generator focus on the goal distribution in trajectories that successfully reach the goal in fewer steps. Given these conditions, the trained generator produces efficient goals through reverse denoising, providing positive goal conditions for subsequent contrastive value learning.

Redundant Goal Generator (G_ψ^-). The redundant goal generator G_ψ^- is implemented as a conditional denoising diffusion probabilistic model (DDPM), trained on the training set $\mathcal{D}^-$. The purpose of G_ψ^- is to synthesize goals that reflect behaviorally suboptimal or misleading intent. It aims to generate both inefficiently reached goals and structurally plausible distractors. We define the unified loss function in Eq. (3), using $\mathcal{D}^-$ as the training set and ϵ_ψ^- as the noise predictor, with the base loss given by $\mathcal{L}^-_{\text{DDPM}}$. The model is trained to denoise corrupted embeddings of redundant goals, conditioned on the current state s_t, thereby modeling the corresponding conditional goal distribution. To ensure that the generated goals are not only structurally valid but also semantically misleading, we introduce an **adversarial sharpness loss** to encourage G_ψ^- to synthesize hard negative goals:

$$\mathcal{L}^-_{\text{adv}} = \mathbb{E}_{(s,a)} \left[\left(Q_\theta(s, a \mid g^+) - Q_\theta(s, a \mid g^-) \right)^2 \right] \tag{4}$$

where $g^+ \sim G_\psi^+(s_t)$ and $g^- \sim G_\psi^-(s_t)$. Both goals are evaluated under the efficient critic Q_θ to directly enforce value-space confusion. This design compels G_ψ^- to generate harder negatives by minimizing their distinction from g^+ under a shared value model. Using the same Q_θ avoids introducing differences between the two critics' estimates into this adversarial comparison. The final training objective for G_ψ^- is a curriculum-weighted combination:

$$\mathcal{L}_{G^-} = \mathcal{L}^-_{\text{DDPM}} + \beta_t \cdot \lambda_{\text{adv}} \cdot \mathcal{L}^-_{\text{adv}} \tag{5}$$

where $\beta_t \in [0, 1]$ is a time-dependent weight that gradually strengthens adversarial supervision during training. Together with G_ψ^+, it forms the core of the GEAR contrastive curriculum. Given the generated efficient goal g^+ and redundant goal g^-, a key question arises: *How can we enable the agent to quantify the difference between efficient and redundant behaviors?* To this end, we introduce the notion of the **efficiency gap**, a contrastive value signal that enables the agent to distinguish efficient behaviors from redundant ones.

3.2 Efficiency Gap as an Auxiliary Learning Signal

We emphasize that the efficiency gap is not intended as a task-independent definition of trajectory efficiency. It is a critic-dependent auxiliary signal used to shape policy updates by contrasting efficient and redundant goal conditions. To convert this contrast between concise goal-reaching behavior and redundant or misleading goal conditions into a usable policy-learning signal, we use the *efficiency gap* as an auxiliary value-based contrast. Rather than assigning an absolute efficiency score to a trajectory, this gap compares how the same state-action pair is evaluated under two goal conditions sampled from the asymmetric generators. Given a transition (s, a), we sample an efficient goal $g^+ \sim G_\psi^+(s)$ and a redundant goal $g^- \sim G_\psi^-(s)$, and compute

$$\Delta Q(s, a) = Q_\theta(s, a \mid g^+) - Q_\omega(s, a \mid g^-) \tag{6}$$

Here, Q_θ evaluates the state-action pair under the efficient goal, while Q_ω evaluates the same state-action pair under the redundant goal. A larger ΔQ indicates that, under the current learned critics, the action is more compatible with the efficient-goal distribution than with the redundant-goal distribution. This makes ΔQ useful for policy learning: maximizing it encourages the actor to favor actions that support efficient goals while suppressing actions that remain highly valued under redundant or misleading goals.

This gap provides a learning signal to help the agent differentiate efficient behaviors from inefficient ones. A larger $\Delta Q(s, a)$ indicates stronger relative support for the efficient goal under the current

critics, while a near-zero gap indicates similar value estimates. To ensure the reliability of this signal, we train two separate critics: $Q_\theta(s, a \mid g^+)$ evaluates the behavior under efficient goals, and $Q_\omega(s, a \mid g^-)$ evaluates the same behavior under redundant goals. Both critics predict a *soft value*, incorporating immediate rewards, expected future rewards, and an entropy bonus for exploration. This setup follows the Soft Actor-Critic (SAC) framework. The target values for training the critics are computed via soft Bellman backups.

$$y^+ = r(s, a, g^+) + \gamma\, \mathbb{E}_{a' \sim \pi(\cdot \mid s', g^+)}[Q_\theta(s', a' \mid g^+) - \alpha \log \pi(a' \mid s', g^+)] \tag{7}$$

$$y^- = r(s, a, g^-) + \gamma\, \mathbb{E}_{a' \sim \pi(\cdot \mid s', g^-)}[Q_\omega(s', a' \mid g^-) - \alpha \log \pi(a' \mid s', g^-)] \tag{8}$$

We then update the critics by minimizing the squared temporal difference (TD) error between the predicted values and their targets:

$$\mathcal{L}_{Q_\theta} = \mathbb{E}_{(s,a,s') \sim \mathcal{D}} \left[\left(Q_\theta(s, a \mid g^+) - y^+ \right)^2 \right] \tag{9}$$

$$\mathcal{L}_{Q_\omega} = \mathbb{E}_{(s,a,s') \sim \mathcal{D}} \left[\left(Q_\omega(s, a \mid g^-) - y^- \right)^2 \right] \tag{10}$$

The resulting efficiency gap, $\Delta Q(s, a)$, serves as an auxiliary supervision signal for training the actor. This gap encourages actions that favor efficient goals over redundant ones under the current critics.

3.3 Contrastive Value-Gap Policy Optimization

To leverage the efficiency gap in policy learning, we optimize the goal-conditioned actor π_ϕ using this contrastive learning signal. As introduced in Section 3.2, the efficiency gap quantifies the relative value of an action a under an efficient goal g^+ compared to a redundant goal g^-. To guide the policy toward concise and reliable behaviors, we define the contrastive actor objective:

$$\mathcal{J}_{\text{gap}}(\phi) = \mathbb{E}_{s \sim \mathcal{D},\, g^+,\, g^-}[\, Q_\theta(s, \pi_\phi(s, g^+) \mid g^+) - Q_\omega(s, \pi_\phi(s, g^+) \mid g^-)] \tag{11}$$

where $g^+ \sim G_\psi^+(s)$ and $g^- \sim G_\psi^-(s)$ are sampled from the efficient and redundant goal generators, respectively. We now formalize the optimality condition for this contrastive objective under fixed critics and goal distributions:

Theorem 1 (Contrastive Gap Optimality)**.** *Let Q_θ and Q_ω be fixed soft Q-functions, with fixed goal and state sampling distributions. If the policy class can realize the pointwise maximizers, the greedy policy induced by $\mathcal{J}_{gap}$ in Equation* (11) *is given by: (Proof in Appendix A.2)*

$$\pi^\star(s, g^+) \in \arg\max_{a \in \mathcal{A}} \left[Q_\theta(s, a \mid g^+) - \mathbb{E}_{g^- \sim G_\psi^-(s)} Q_\omega(s, a \mid g^-) \right] \tag{12}$$

Based on Theorem 1, when the critics and generated goal distributions are fixed, the pointwise maximizer of the Q-gap defines a greedy action-selection rule for $\mathcal{J}_{\text{gap}}$. This theorem characterizes the optimization behavior of the gap-based training objective, rather than providing a global optimality guarantee for the original RL task.

Mutual Information Regularization. While the gap objective $\mathcal{J}_{\text{gap}}$ effectively encourages the policy to prefer actions that lead to efficient goals over redundant ones, it alone may fail to ensure strong goal-conditioning, especially under sparse reward or ambiguous goal distributions. To address this, we incorporate a mutual information (MI) regularizer that maximizes $I_\phi(a; g^+ \mid s)$: the dependency between actions and goals given the current state.

$$\mathcal{J}_{\text{MI}}(\phi) = \beta \cdot \mathbb{E}_s \left[I_\phi(a; g^+ \mid s) \right] \tag{13}$$

where β is a tunable coefficient. This term encourages the policy to use goal conditioning and adapt its actions to goal changes. We implement this regularizer using a conditional InfoNCE lower bound, with sampling and optimization details provided in Appendix C.6. Combining the two terms, we obtain the final actor objective:

$$\mathcal{J}_{\text{gear}}(\phi) = \mathcal{J}_{\text{gap}}(\phi) + \mathcal{J}_{\text{MI}}(\phi) \tag{14}$$

As we show next, under the stated alignment condition, the joint gradient remains an ascent direction for $\mathcal{J}_{\text{gap}}$ and has a norm no smaller than the gap gradient.

Theorem 2 (MI-Augmented Gap Objective)**.** *Let $c > 0$ such that, at the current iterate ϕ, $\nabla_\phi \mathcal{J}_{\mathrm{gap}}(\phi) \neq 0$ and the MI gradient satisfies the following alignment condition:*

$$\left\langle \nabla_\phi \mathbb{E}_{s\sim\mathcal{D}}\left[I_\phi(a; g^+ \mid s)\right], \nabla_\phi \mathcal{J}_{\mathrm{gap}}(\phi)\right\rangle \geq c \left\|\nabla_\phi \mathbb{E}_{s\sim\mathcal{D}}\left[I_\phi(a; g^+ \mid s)\right]\right\|^2 \tag{15}$$

*If $0 \leq \beta \leq \beta^\star$ for some threshold $\beta^\star > 0$ (*a conservative sufficient condition*; see Appendix A.3), then the update direction $\nabla_\phi \mathcal{J}_{\mathrm{gear}}(\phi)$ used in $\phi \leftarrow \phi + \eta \cdot \nabla_\phi \mathcal{J}_{\mathrm{gear}}(\phi)$ satisfies the following: (i)* Ascent compatibility with $\mathcal{J}_{\mathrm{gap}}$*: the joint direction $\nabla_\phi \mathcal{J}_{\mathrm{gear}}(\phi)$ is an ascent direction for $\mathcal{J}_{\mathrm{gap}}$, i.e., $\langle \nabla_\phi \mathcal{J}_{\mathrm{gear}}(\phi), \nabla_\phi \mathcal{J}_{\mathrm{gap}}(\phi)\rangle > 0$. (ii)* Non-decreasing gradient norm*: the joint gradient has no norm smaller than the gap gradient: $\|\nabla_\phi \mathcal{J}_{\mathrm{gear}}(\phi)\| \geq \|\nabla_\phi \mathcal{J}_{\mathrm{gap}}(\phi)\|$.*

Remark. These two theorems establish a concrete foundation for our contrastive learning objective. Specifically, Theorem 1 characterizes greedy action selection under fixed critics and goal distributions. Theorem 2 further shows that incorporating mutual information (MI) regularization preserves an ascent direction for $\mathcal{J}_{\mathrm{gap}}$ under the stated gradient alignment condition. Building on these insights, we integrate the contrastive objective into a closed-loop training framework. The redundant goal generator G_ψ^- is updated to produce challenging distractor goals, encouraging the actor to better discriminate between efficient and redundant behaviors. This dynamic interplay forms the core of GEAR's adaptive contrastive mechanism, which we further analyze from a game-theoretic perspective in Appendix A. The complete training loop is provided in Appendix A.1.

4 Experiments

We design experiments to evaluate GEAR and validate the four core claims: **(C1):** GEAR outperforms prior goal-conditioned methods in task performance and sample efficiency in simulation, with additional success-rate evaluation on real-world tasks. **(C2):** Key components, such as the dual-critic architecture, gap-guided actor optimization, and co-adaptive pseudo-goal generation, are essential for GEAR's performance. **(C3):** GEAR introduces only a modest computational overhead relative to diffusion-based baselines in practical use. **(C4):** GEAR enables a structured learning curriculum, where redundant goals are generated adversarially to challenge the policy, facilitating skill acquisition. We evaluate GEAR on goal-conditioned tasks from DeepMind Control, ManiSkill2, Habitat and a real-world robotic manipulation. Our analysis includes quantitative results, ablation studies, and curriculum dynamics.

4.1 Environments and Baselines

We evaluate GEAR on three simulated goal-conditioned benchmarks and one real-world robotic manipulation benchmark: **DMC** (Tassa et al., 2018) tests sample efficiency in sparse-reward low-dimensional control; **ManiSkill2** (Gu et al., 2023) evaluates visual goal disambiguation in manipulation tasks with complex object interactions; **Habitat** (Savva et al., 2019; Szot et al., 2021; Puig et al., 2023) evaluates visual robustness and generalization in indoor HM3D navigation scenes built from real-world 3D scans (Ramakrishnan et al., 2021); **Real-World Robotic Control** evaluates physical manipulation tasks with an Elephant Robotics manipulator. Full details are provided in Appendix B.

Baselines. We compare GEAR with several strong and diverse baselines to highlight its advantages: **PPO** (Schulman et al., 2017): An on-policy baseline that contrasts the benefits of experience replay and goal relabeling, providing a strong reference for comparison. **SAC** (Haarnoja et al., 2018): An off-policy actor–critic baseline. **HER** (Andrychowicz et al., 2017): A standard hindsight experience relabeling method with off-policy training, commonly used in GCRL tasks. **RIG** (Nair et al., 2018): A latent goal generation method that uses a variational autoencoder to generate goals. **Decision Diffuser (DD)** (Ajay et al., 2023): A diffusion-based policy model conditioned on return, enabling offline RL without relying on dynamic programming. **QVPO** (Ding et al., 2024): A diffusion-based method using Q-weighted variational policy optimization. All baselines use the same random seeds for fair comparison. Full details are provided in Appendix C.

4.2 Main Results

Overall Performance (C1). Table 1, Table 2, and Figure 3 report the performance of GEAR and baseline methods. For DMC, we report episodic return; for ManiSkill2 and Habitat, we use success

rate. GEAR outperforms all baselines, validating C1. GEAR accelerates learning via contrastive supervision and gap-guided actor optimization, outperforming recent methods like QVPO and DD. It also performs well in high-dimensional visual tasks like ManiSkill2 and Habitat, with generalization evaluated across different Habitat training and validation scenes. Additional results in Appendix B.

Table 1: Goal-conditioned control on DMC: **raw** episodic return at 4M steps (mean ± std, 5 seeds).

Method	`Acrobo.`	`Cartpol.`	`Cheetah-Run`	`Finger-Spin`	`Reacher-hard`	`Walker-Walk`
SAC	43.8 ± 21.4	802.1 ± 42.7	596.3 ± 17.9	803.5 ± 45.2	923.0 ± 13.3	902.7 ± 16.6
PPO	8.5 ± 2.1	240.6 ± 21.8	82.9 ± 13.4	18.2 ± 4.7	288.3 ± 26.0	64.4 ± 12.7
HER	78.7 ± 34.0	816.2 ± 31.5	610.8 ± 36.1	817.3 ± 41.9	935.6 ± 17.4	917.9 ± 9.3
RIG	64.2 ± 18.6	824.4 ± 26.8	629.1 ± 32.3	829.7 ± 31.0	927.5 ± 14.8	924.1 ± 14.2
DD	143.5 ± 26.9	826.0 ± 23.1	654.6 ± 21.7	833.8 ± 29.2	938.4 ± 21.1	936.7 ± 11.8
QVPO	162.9 ± 34.5	835.3 ± 25.4	663.2 ± 28.1	836.6 ± 37.0	943.5 ± 15.2	945.8 ± 12.3
GEAR	**192.6 ± 28.3**	**843.9 ± 27.4**	**680.1 ± 27.5**	**852.1 ± 11.6**	**951.2 ± 7.9**	**956.0 ± 8.5**

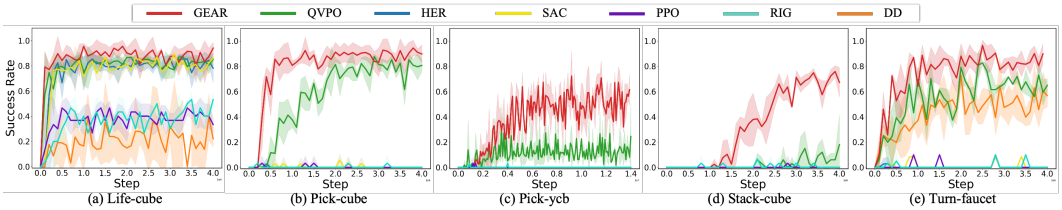

Figure 3: The success rate of ManiSkill2's five tasks [Lift, Pick-cube, Pick-ycb, Stack, Turn]

Table 2: GEAR performance and ablations on Habitat (success rate (%), mean ± std, 5 seeds).

Method	**Residential**	**Office**	**Commercial**	**Method**	**Residential**	**Office**	**Commercial**
SAC	41.2 ± 2.9	48.5 ± 2.4	44.8 ± 2.7	**GEAR**	$\mathbf{61.7 \pm 2.1}$	$\mathbf{78.9 \pm 3.2}$	$\mathbf{64.8 \pm 2.9}$
PPO	38.7 ± 3.1	43.1 ± 2.6	40.2 ± 2.8	w/o Red.	52.4 ± 3.6	55.1 ± 4.1	46.9 ± 3.9
HER	46.8 ± 3.2	52.4 ± 2.8	49.3 ± 3.0	w/o Dual	51.3 ± 3.3	62.2 ± 3.9	53.6 ± 2.9
RIG	50.1 ± 2.8	58.2 ± 2.2	54.7 ± 2.5	w/o Gap.	54.7 ± 3.1	58.4 ± 3.7	51.8 ± 3.4
DD	53.6 ± 3.3	64.3 ± 2.6	58.1 ± 2.9	w/o MI	59.6 ± 2.6	76.3 ± 3.4	62.2 ± 3.2
QVPO	57.2 ± 2.4	68.4 ± 3.1	60.9 ± 2.3	w/o Diff.	45.2 ± 2.1	53.7 ± 3.6	46.1 ± 2.4

Ablation of GEAR Components (C2). To validate C2, we ablate GEAR components on Habitat (Table 2). **(1) w/o G_ψ^-.** Removing the redundant goal generator eliminates the negative side of the efficient–redundant contrast, weakening the value-gap signal and causing a clear performance drop. **(2) w/o Dual Critics.** Using a shared critic for both goal types may blur their value distinction, making ΔQ less reliable for policy learning. **(3) w/o Actor Update via ΔQ.** Without gap-guided actor optimization, the policy can still follow efficient goals but cannot explicitly suppress actions favored under redundant goals, reducing task success rates. **(4) w/o MI Regularization.** Removing MI may weaken goal sensitivity and increases the reported standard deviations; final performance remains competitive. **(5) w/o Diffusion.** Replacing diffusion-based generation may reduce goal diversity and adaptability to unseen states, consistent with the observed performance drop.

Table 3: Real-world robotic manipulation on Elephant Robotics: successes in 20 trials per task.

Method	**Object Lift**	**Block Stack**	**Target Grasp**	**Method**	**Object Lift**	**Block Stack**	**Target Grasp**
HER	14/20 (70%)	9/20 (45%)	9/20 (45%)	RIG	15/20 (75%)	10/20 (50%)	11/20 (55%)
QVPO	17/20 (85%)	13/20 (65%)	14/20 (70%)	**GEAR**	**18/20 (90%)**	**15/20 (75%)**	**16/20 (80%)**

Overhead and Deployment Feasibility (C3). We further analyze practical aspects of GEAR on Habitat `Office`. Table 4 reports wall-clock training time for 2M training steps on an RTX 3090, with GEAR taking 816 minutes, 5.4% longer than QVPO and 27.5% longer than SAC. Additional scalability results in Appendix B.2 show that GEAR remains stable when increasing actor–critic capacity and enlarging the achieved-goal pool on Habitat `Office`. To further assess deployment feasibility, Table 3 reports a real-world robotic manipulation study on an Elephant Robotics manipulator across three tasks. Full results in Appendix B.1.

Table 4: Training time on Habitat `Office` for 2M vision-based training steps on an RTX 3090.

Method	SAC	PPO	HER	RIG	QVPO	DD	**GEAR**	GEAR w/o Diff.
Time (min)	640	685	690	732	774	786	**816**	652

Emergent Curriculum (C4). Figure 4 reports the normalized average Q-gap and redundant-goal entropy on Habitat `Office`. As training proceeds, the Q-gap generally increases, reflecting stronger value contrast between efficient and redundant goal conditions under the current critics. At the same time, redundant-goal entropy rises, indicating that the generator produces increasingly diverse redundant goals rather than collapsing to repetitive negatives. These coupled trends are consistent with GEAR's emergent curriculum: the generator provides diverse redundant goals and is trained adversarially to challenge the policy. This self-adjusting behavior can be interpreted through the Stackelberg interaction between the generator and the policy, where the Q-gap guides policy updates, enabling GEAR to challenge the learner without manually specifying a sequence of distractor goals.

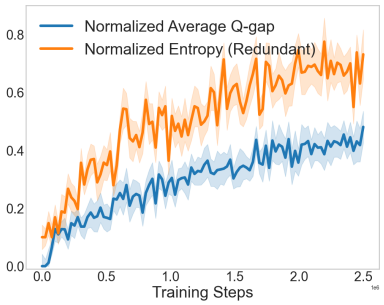

Figure 4: Q-Gap and Entropy Evolution

5 Related Work

Goal Generation in GCRL. Recent GCRL advances explore diverse generative paradigms for goal sampling. GOAL-GAN (Florensa et al., 2018) uses adversarial training to generate goals near the agent's capability frontier. SkewFit (Pong et al., 2020) prioritizes low-density goals via inverse-density sampling. Latent models like RIG (Nair et al., 2018) and GoPlan (Wang et al., 2024a) sample structured goals from learned latent spaces. Curriculum-based methods such as C-planning (Zhang et al., 2021) and PLR (Jiang et al., 2021) adaptively schedule goals based on learning progress or regret. Diffusion models,including Diffuser (Janner et al., 2022), Decision Diffuser (Ajay et al., 2023), and Latent Diffuser (Li, 2023),generate trajectory-level plans in pixel or latent space, while CPQL (Chen et al., 2024), CEP (Lu et al., 2023a), and QVPO (Ding et al., 2024) integrate diffusion into policy learning.

Experience Replay in GCRL. Experience Replay (Kapturowski et al., 2019), initially explored by (Lin, 1992) and widely adopted in Deep Q-Networks (DQN) by (Mnih et al., 2013), stabilizes reinforcement learning by decorrelating sequential data. Prioritized Experience Replay (PER) (Schaul et al., 2015) enhances learning efficiency by favoring informative samples, with variants such as actor-critic PER (Saglam et al., 2022) and new priority criteria (Sujit et al., 2023; Hong et al., 2022; Yu et al., 2024a). Crucially for Goal-Conditioned RL, Hindsight Experience Replay (HER) (Andrychowicz et al., 2017) converts failed episodes into valuable learning by relabeling goals, and it integrates well with curricula (Fang et al., 2019) and multi-goal learning (Plappert et al., 2018). Recent advances include Contact Energy-Based Prioritization (Sayar et al., 2024), distributed ER architectures like Ape-X (Horgan et al., 2018), as well as Prioritized Generative Replay (PGR) (Wang et al., 2025) and SynthER (Lu et al., 2023b), which employ generative models to synthesize or prioritize replay data.

6 Conclusion

We presented **GEAR**, a generative framework for goal-conditioned reinforcement learning that exploits a contrastive value gap between goals. GEAR uses dual conditional diffusion generators to produce efficient and redundant goals, together with dual critics to estimate their value gap. This contrastive signal guides policy optimization by promoting purposeful behaviors and suppressing inefficient ones, inducing a self-adaptive curriculum aligned with agent progress. Experiments across benchmarks show that GEAR improves task performance and sample efficiency over prior methods, demonstrating the value of contrastive efficiency learning.

REFERENCES

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making?, 2023. URL https://arxiv.org/abs/2211.15657. First posted 2022-11-28; revised 2023-07-10.

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 *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/hash/453fadbd8a1a3af50a9df4df899537b5-Abstract.html.

Hongye Cao, Fan Feng, Tianpei Yang, Jing Huo, and Yang Gao. Causal information prioritization for efficient reinforcement learning, 2025. URL https://arxiv.org/abs/2502.10097.

Yuhui Chen, Haoran Li, and Dongbin Zhao. Boosting continuous control with consistency policy, 2024. URL https://arxiv.org/abs/2310.06343. First posted 2023-10-10; revised 2024-01-24.

Ching-An Cheng, Andrey Kolobov, and Adith Swaminathan. Heuristic-guided reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 34, pp. 13550–13563, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/70d31b87bd021441e5e6bf23eb84a306-Abstract.html.

Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via q-weighted variational policy optimization. In *Advances in Neural Information Processing Systems*, volume 37, pp. 53945–53968, 2024. URL https://arxiv.org/abs/2405.16173.

Yiming Ding, Carlos Florensa, Pieter Abbeel, and Mariano Phielipp. Goal-conditioned imitation learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/c8d3a760ebab631565f8509d84b3b3f1-Abstract.html.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Russ R Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 35603–35620. Curran Associates, Inc., 2022. doi: 10.52202/068431-2580. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/e7663e974c4ee7a2b475a4775201ce1f-Paper-Conference.pdf.

Meng Fang, Tianyi Zhou, Yali Du, Lei Han, and Zhengyou Zhang. Curriculum-guided hindsight experience replay. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/83715fd4755b33f9c3958e1a9ee221e1-Abstract.html.

Guhao Feng and Han Zhong. Rethinking model-based, policy-based, and value-based reinforcement learning via the lens of representation complexity. In *Advances in Neural Information Processing Systems*, volume 37, pp. 8569–8611, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/0fe6a18be9491139fb759e2f645374b1-Abstract-Conference.html.

Carlos Florensa, David Held, Xinyang Geng, and Pieter Abbeel. Automatic goal generation for reinforcement learning agents, 2018. URL https://arxiv.org/abs/1705.06366. First posted 2017-05-17; revised 2018-07-23.

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. Maniskill2: A unified benchmark for generalizable manipulation skills, 2023. URL https://arxiv.org/abs/2302.04659.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Kihyuk Hong, Yuhang Li, and Ambuj Tewari. A primal-dual-critic algorithm for offline constrained reinforcement learning. In *International conference on artificial intelligence and statistics*, pp. 280–288. PMLR, 2024. URL https://proceedings.mlr.press/v238/hong24a.html.

Zhang-Wei Hong, Tao Chen, Yen-Chen Lin, Joni Pajarinen, and Pulkit Agrawal. Topological experience replay. *arXiv preprint arXiv:2203.15845*, 2022. URL https://arxiv.org/abs/2203.15845.

Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. Distributed prioritized experience replay. *arXiv preprint arXiv:1803.00933*, 2018. URL https://arxiv.org/abs/1803.00933.

Peide Huang, Mengdi Xu, Fei Fang, and Ding Zhao. Robust reinforcement learning as a stackelberg game via adaptively-regularized adversarial training, 2022. URL https://arxiv.org/abs/2202.09514.

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

Minqi Jiang, Edward Grefenstette, and Tim Rocktäschel. Prioritized level replay, 2021. URL https://arxiv.org/abs/2010.03934. First posted 2020-10-08; revised 2021-06-12.

Can Jin, Tong Che, Hongwu Peng, Yiyuan Li, Dimitris N. Metaxas, and Marco Pavone. Learning from teaching regularization: Generalizable correlations should be easy to imitate. In *Advances in Neural Information Processing Systems*, volume 37, pp. 966–994, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/01ce1ae7f94d139e4917f9e4425a4f38-Abstract-Conference.html.

Steven Kapturowski, Georg Ostrovski, John Quan, Rémi Munos, and Will Dabney. Recurrent experience replay in distributed reinforcement learning. In *International Conference on Learning Representations*, 2019. URL https://iclr.cc/virtual/2019/poster/648.

Wenhao Li. Efficient planning with latent diffusion, 2023. URL https://arxiv.org/abs/2310.00311.

Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321, 1992. doi: 10.1007/bf00992699. URL https://link.springer.com/article/10.1007/BF00992699.

Bo Liu, Yihao Feng, Qiang Liu, and Peter Stone. Metric residual networks for sample efficient goal-conditioned reinforcement learning, 2023. URL https://arxiv.org/abs/2208.08133.

Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning, 2023a. URL https://arxiv.org/abs/2304.12824.

Cong Lu, Philip J. Ball, Yee Whye Teh, and Jack Parker-Holder. Synthetic experience replay, 2023b. URL https://arxiv.org/abs/2303.06614.

Yu Luo, Tianying Ji, Fuchun Sun, Huaping Liu, Jianwei Zhang, Mingxuan Jing, and Wenbing Huang. Goal-conditioned hierarchical reinforcement learning with high-level model approximation. *IEEE Transactions on Neural Networks and Learning Systems*, 36(2):2705–2719, 2025. doi: 10.1109/tnnls.2024.3354061. URL https://doi.org/10.1109/TNNLS.2024.3354061.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

Gaia Molinaro, Cédric Colas, Pierre-Yves Oudeyer, and Anne G. E. Collins. Latent learning progress drives autonomous goal selection in human reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 32251–32280, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/38c5feed4b72c96f6cf925ccc9832ecf-Abstract-Conference.html.

Ashvin 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, 2018. URL https://arxiv.org/abs/1807.04742.

Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *Journal of Machine Learning Research*, 21(181):1–50, 2020. URL http://jmlr.org/papers/v21/20-212.html.

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, 2018. URL https://arxiv.org/abs/1802.09464.

Vitchyr H. Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning, 2020. URL https://arxiv.org/abs/1903.03698. First posted 2019-03-08; revised 2020-08-04.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5171–5180. PMLR, 2019. URL https://proceedings.mlr.press/v97/poole19a.html.

Xavier Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander William Clegg, Michal Hlavac, So Yeon Min, Vladimír Vondruš, Theophile Gervet, Vincent-Pierre Berges, John M. Turner, Oleksandr Maksymets, Zsolt Kira, Mrinal Kalakrishnan, Jitendra Malik, Devendra Singh Chaplot, Unnat Jain, Dhruv Batra, Akshara Rai, and Roozbeh Mottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots, 2023. URL https://arxiv.org/abs/2310.13724.

Martin L. Puterman. Markov decision processes. In D. P. Heyman and M. J. Sobel (eds.), *Stochastic Models*, volume 2 of *Handbooks in Operations Research and Management Science*, chapter 8, pp. 331–434. Elsevier, 1990. doi: 10.1016/s0927-0507(05)80172-0. URL https://doi.org/10.1016/S0927-0507(05)80172-0.

Ravi Raj and Andrzej Kos. Intelligent mobile robot navigation in unknown and complex environment using reinforcement learning technique. *Scientific Reports*, 14(1):22852, 2024. doi: 10.1038/s41598-024-72857-3. URL https://www.nature.com/articles/s41598-024-72857-3.

Santhosh K. Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X. Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai, 2021. URL https://arxiv.org/abs/2109.08238.

Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal-conditioned imitation learning using score-based diffusion policies. *arXiv preprint arXiv:2304.02532*, 2023. URL https://arxiv.org/abs/2304.02532.

Baturay Saglam, Furkan B. Mutlu, Dogan C. Cicek, and Suleyman S. Kozat. Actor prioritized experience replay. *arXiv preprint arXiv:2209.00532*, 2022. URL https://arxiv.org/abs/2209.00532.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied ai research. In *Proceedings of the*

IEEE/CVF International Conference on Computer Vision (ICCV), pp. 9339–9347, 2019. URL https://openaccess.thecvf.com/content_ICCV_2019/html/Savva_Habitat_A_Platform_for_Embodied_AI_Research_ICCV_2019_paper.html.

Erdi Sayar, Zhenshan Bing, Carlo D'Eramo, Ozgur S Oguz, and Alois Knoll. Contact energy based hindsight experience prioritization. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5434–5440. IEEE, 2024. doi: 10.1109/icra57147.2024.10610910. URL https://arxiv.org/abs/2312.02677.

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015. URL https://arxiv.org/abs/1511.05952.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

Shivakanth Sujit, Somjit Nath, Pedro Braga, and Samira Ebrahimi Kahou. Prioritizing samples in reinforcement learning with reducible loss. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 23237–23258. Curran Associates, Inc., 2023. doi: 10.52202/075280-1008. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/48726631f87322012c6be38e00c72a47-Paper-Conference.pdf.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimír Vondruš, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems*, volume 34, pp. 251–266, 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/021bbc7ee20b71134d53e20206bd6feb-Abstract.html.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018. URL https://arxiv.org/abs/1801.00690.

Mianchu Wang, Rui Yang, Xi Chen, Hao Sun, Meng Fang, and Giovanni Montana. Goplan: Goal-conditioned offline reinforcement learning by planning with learned models, 2024a. URL https://arxiv.org/abs/2310.20025. First posted 2023-10-30; revised 2024-05-16.

Renhao Wang, Kevin Frans, Pieter Abbeel, Sergey Levine, and Alexei A. Efros. Prioritized generative replay, 2025. URL https://arxiv.org/abs/2410.18082.

Yiming Wang, Kaiyan Zhao, Furui Liu, and Leong Hou U. Rethinking exploration in reinforcement learning with effective metric-based exploration bonus. In *Advances in Neural Information Processing Systems*, volume 37, pp. 57765–57792, 2024b. URL https://openreview.net/forum?id=QpKWFLtZKi.

Beyazit Yalcinkaya, Niklas Lauffer, Marcell Vazquez-Chanlatte, and Sanjit A. Seshia. Compositional automata embeddings for goal-conditioned reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 37, pp. 72933–72963, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/858fc542b70d3b39067f7d3b1cd77635-Abstract-Conference.html.

Jingyun Yang, Max Sobol Mark, Brandon Vu, Archit Sharma, Jeannette Bohg, and Chelsea Finn. Robot fine-tuning made easy: Pre-training rewards and policies for autonomous real-world reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4804–4811. IEEE, 2024. doi: 10.1109/icra57147.2024.10610421. URL https://arxiv.org/abs/2310.15145.

Rui Yang, Yong Lin, Xiaoteng Ma, Hao Hu, Chongjie Zhang, and Tong Zhang. What is essential for unseen goal generalization of offline goal-conditioned rl?, 2023. URL https://arxiv.org/abs/2305.18882.

Jiayu Yu, Jingyao Li, Shuai Lü, and Shuai Han. Mixed experience sampling for off-policy reinforcement learning. *Expert Systems with Applications*, 251:124017, 2024a. doi: 10.1016/j.eswa.2024.124017. URL https://doi.org/10.1016/j.eswa.2024.124017.

Xinyao Yu, Sixian Zhang, Xinhang Song, Xiaorong Qin, and Shuqiang Jiang. Trajectory diffusion for objectgoal navigation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 110388–110411, 2024b. URL https://papers.neurips.cc/paper_files/paper/2024/hash/c72861451d6fa9dfa64831102b9bb71a-Abstract-Conference.html.

Kaidong Zhang, Pengzhen Ren, Bingqian Lin, Junfan Lin, Shikui Ma, Hang Xu, and Xiaodan Liang. Pivot-r: Primitive-driven waypoint-aware world model for robotic manipulation, 2024. URL https://arxiv.org/abs/2410.10394.

Tianjun Zhang, Benjamin Eysenbach, Ruslan Salakhutdinov, Sergey Levine, and Joseph E. Gonzalez. C-planning: An automatic curriculum for learning goal-reaching tasks, 2021. URL https://arxiv.org/abs/2110.12080.

A Theoretical Analysis and Proofs

A.1 Training Algorithm

Algorithm 1 GEAR Training Loop

Require: Replay buffer $\mathcal{D}$, actor π_ϕ, critics Q_θ, Q_ω, generators G_ψ^+, G_ψ^-

for each training step **do**
  Sample transition $(s_t, a_t, s_{t+1}) \sim \mathcal{D}$
  Sample efficient goal $g^+ \sim G_\psi^+(s_t)$
  Sample redundant goal $g^- \sim G_\psi^-(s_t)$
  Compute Bellman targets y^+ and y^- for g^+ and g^-
  Update critic $Q_\theta \leftarrow \text{MSE}(Q_\theta(s_t, a_t \mid g^+), y^+)$
  Update critic $Q_\omega \leftarrow \text{MSE}(Q_\omega(s_t, a_t \mid g^-), y^-)$
  Update actor π_ϕ using contrastive objective $\mathcal{J}_{\text{gear}}(\phi)$
  Sample trajectories for the generator training batches according to p^+ and p^-
  Update G_ψ^+ using diffusion loss $\mathcal{L}_{G^+} = \mathcal{L}_{\text{DDPM}}^+$
  Update G_ψ^- using combined loss: $\mathcal{L}_{G^-} = \mathcal{L}_{\text{DDPM}}^- + \beta_t \cdot \lambda_{\text{adv}} \cdot \mathcal{L}_{\text{adv}}^-$
end for

Algorithm 1 provides the complete training procedure of GEAR. At each training step, the agent first samples transitions from the replay buffer and then generates two goal types conditioned on the current state: an efficient goal g^+ and a redundant goal g^-. The two goal-conditioned critics are updated separately, allowing the model to estimate the value difference between efficient and redundant behaviors. The actor is then optimized using the contrastive value-gap objective, which encourages actions that are valuable under efficient goals while suppressing redundant goal-conditioned behaviors. Finally, using the corresponding training data obtained through weighted sampling, the efficient generator is trained with a standard diffusion loss, whereas the redundant generator is further shaped by an adversarial term to produce informative and nontrivial redundant goals.

A.2 Proof of Contrastive Gap Optimality

The actor is conditioned only on g^+ in the contrastive objective. The redundant goal g^- is not an execution-time input to the policy. In GEAR, g^- is generated by the redundant goal generator only during training as an auxiliary contrastive signal for penalizing inefficient or redundant behavior. It appears in the critic evaluation and in the actor loss, where it shapes the policy gradient by discouraging actions that are also valuable under redundant goals. In the contrastive actor objective, the same action $\pi(s, g^+)$ is evaluated under both goals; g^- is not an additional policy input. Therefore, the actor is defined as a standard goal-conditioned policy $\pi(s, g^+)$, while g^- is used only inside the training objective and is marginalized out in the greedy action-selection rule. For Theorem 1, recall the contrastive gap objective:

$$\mathcal{J}_{\text{gap}}(\pi) = \mathbb{E}_{\substack{s\sim\mathcal{D} \\ g^+\sim G_\psi^+(s) \\ g^-\sim G_\psi^-(s)}} \left[Q_\theta\big(s, \pi(s, g^+) \mid g^+\big) - Q_\omega\big(s, \pi(s, g^+) \mid g^-\big) \right]$$

Following this information structure, for any fixed (s, g^+), we define the marginalized gap score

$$A_{\text{gap}}(s, a, g^+) = Q_\theta(s, a \mid g^+) - \mathbb{E}_{g^-\sim G_\psi^-(s)} \left[Q_\omega(s, a \mid g^-) \right]$$

Using this definition, the objective can be rewritten as

$$\mathcal{J}_{\text{gap}}(\pi) = \mathbb{E}_{\substack{s\sim\mathcal{D} \\ g^+\sim G_\psi^+(s)}} \left[A_{\text{gap}}\big(s, \pi(s, g^+), g^+\big) \right]$$

Lemma 1 (Marginalized Gap Decomposition). When Q_θ, Q_ω, G_ψ^+, and G_ψ^- are fixed, the state sampling distribution is fixed, and the policy class can realize the pointwise maximizers, maximizing

$\mathcal{J}_{\mathrm{gap}}$ over deterministic goal-conditioned policies decomposes into pointwise maximization of the marginalized gap score for each pair (s, g^+):

$$\pi^\star(s, g^+) \in \arg\max_{a \in \mathcal{A}} A_{\mathrm{gap}}(s, a, g^+)$$

Proof. The sampling distributions over s, g^+, and g^- are fixed, and the critics are fixed. Since the policy selects an action based only on (s, g^+), the random variable g^- can be marginalized inside the objective. Thus, for each (s, g^+), the contribution of the policy to $\mathcal{J}_{\mathrm{gap}}$ is fully determined by $A_{\mathrm{gap}}(s, \pi(s, g^+), g^+)$. Therefore, the greedy action that maximizes this marginalized gap score provides the pointwise maximizer of the local gap-based objective. □

Proof of Theorem 1. By Lemma 1, the contrastive objective can be expressed as an expectation of the marginalized gap score:

$$\mathcal{J}_{\mathrm{gap}}(\pi) = \mathbb{E}_{s, g^+}\left[A_{\mathrm{gap}}\big(s, \pi(s, g^+), g^+\big)\right]$$

Therefore, for fixed critics and fixed generated goal distributions, a greedy policy that maximizes the local gap score for each (s, g^+) is given by

$$\pi^\star(s, g^+) \in \arg\max_{a \in \mathcal{A}} \left[Q_\theta(s, a \mid g^+) - \mathbb{E}_{g^- \sim G_\psi^-(s)} Q_\omega(s, a \mid g^-)\right]$$

This proves that the theorem characterizes the greedy action-selection rule induced by the gap-based objective. The result concerns optimization of the auxiliary contrastive objective under fixed critics and fixed goal generators, and does not imply global optimality for the original reinforcement learning task. □

Intuition. The efficient critic and the redundant critic evaluate the same candidate action from two different goal-conditioned perspectives. The actor is asked to choose an action that receives a high value under the efficient goal while receiving a low expected value under redundant goals. Since redundant goals are averaged out rather than given to the actor as inputs, g^- acts as a training-time contrastive reference, not as an execution-time conditioning variable.

A.3 Proof of MI–Augmented Gap Preservation

We first recall the *gap objective* used in Theorem 2:

$$\mathcal{J}_{\mathrm{gap}}(\phi) = \mathbb{E}_{\substack{s \sim \mathcal{D} \\ g^+ \sim G_\psi^+(s) \\ g^- \sim G_\psi^-(s)}}\left[Q_\theta\big(s, \pi_\phi(s, g^+) \mid g^+\big) - Q_\omega\big(s, \pi_\phi(s, g^+) \mid g^-\big)\right]$$

and the MI regularizer, weighted by $\beta \geq 0$,

$$\mathcal{J}_{\mathrm{MI}}(\phi) = \beta\, \mathbb{E}_{s \sim \mathcal{D}}\big[I_\phi(a; g^+ \mid s)\big]$$

The combined actor objective is

$$\mathcal{J}_{\mathrm{gear}}(\phi) = \mathcal{J}_{\mathrm{gap}}(\phi) + \mathcal{J}_{\mathrm{MI}}(\phi)$$

Gradient notation and assumptions. Define the gradients at the current iterate ϕ as

$$\mathbf{g} := \nabla_\phi \mathcal{J}_{\mathrm{gap}}(\phi) \qquad \mathbf{m} := \nabla_\phi \mathbb{E}_{s \sim \mathcal{D}}\big[I_\phi(a; g^+ \mid s)\big]$$

Then the total gradient direction used by the actor update is

$$\mathbf{u} := \nabla_\phi \mathcal{J}_{\mathrm{gear}}(\phi) = \mathbf{g} + \beta\, \mathbf{m}$$

By the assumptions in Theorem 2, the current iterate is not at a stationary point of the gap objective, i.e., $\mathbf{g} \neq \mathbf{0}$. Furthermore, there exists a constant $c > 0$ such that the MI gradient satisfies the following alignment condition:

$$\langle \mathbf{m}, \mathbf{g} \rangle \geq c\, \|\mathbf{m}\|^2 \tag{16}$$

Since $c > 0$ and $\|\mathbf{m}\|^2 \geq 0$, this alignment condition implies $\langle \mathbf{m}, \mathbf{g} \rangle \geq 0$.

Proof of Theorem 2. We show that for any $\beta \geq 0$, the joint update direction $\mathbf{u} = \mathbf{g} + \beta\mathbf{m}$ (i) preserves ascent on $\mathcal{J}_{\text{gap}}$ and (ii) does not decrease the gradient norm relative to $\mathbf{g}$.

(i) Ascent compatibility with $\mathcal{J}_{\text{gap}}$. The directional derivative of $\mathcal{J}_{\text{gap}}$ along $\mathbf{u}$ is given by the inner product $\langle \mathbf{u}, \mathbf{g} \rangle$. Using $\mathbf{u} = \mathbf{g} + \beta\mathbf{m}$, we have

$$\langle \mathbf{u}, \mathbf{g} \rangle = \langle \mathbf{g} + \beta\mathbf{m}, \mathbf{g} \rangle = \|\mathbf{g}\|^2 + \beta \langle \mathbf{m}, \mathbf{g} \rangle$$

Since $\beta \geq 0$ and $\langle \mathbf{m}, \mathbf{g} \rangle \geq c\|\mathbf{m}\|^2 \geq 0$, we obtain

$$\langle \mathbf{u}, \mathbf{g} \rangle \geq \|\mathbf{g}\|^2$$

Because $\mathbf{g} \neq \mathbf{0}$, it follows that $\|\mathbf{g}\|^2 > 0$. Therefore,

$$\langle \mathbf{u}, \mathbf{g} \rangle > 0$$

meaning $\mathbf{u}$ is a strict ascent direction for $\mathcal{J}_{\text{gap}}$.

(ii) Non-decreasing gradient norm. We compare the squared norms of the gradients:

$$\|\mathbf{u}\|^2 = \|\mathbf{g} + \beta\mathbf{m}\|^2 = \|\mathbf{g}\|^2 + 2\beta\langle \mathbf{m}, \mathbf{g} \rangle + \beta^2\|\mathbf{m}\|^2$$

Using the alignment condition in (16), we get

$$\|\mathbf{u}\|^2 \geq \|\mathbf{g}\|^2 + 2\beta c\|\mathbf{m}\|^2 + \beta^2\|\mathbf{m}\|^2 \geq \|\mathbf{g}\|^2$$

where the last inequality holds because $\beta \geq 0$, $c > 0$, and $\|\mathbf{m}\|^2 \geq 0$. Taking the square root of both sides yields $\|\mathbf{u}\| \geq \|\mathbf{g}\|$.

This proves (i) and (ii) for any $\beta \geq 0$, and thus also for the restricted range $0 \leq \beta \leq \beta^\star$. □

Remark on the conservative threshold $\beta^\star$. As shown above, under the deterministic alignment condition in (16), conclusions (i) and (ii) hold for *any* $\beta \geq 0$. We introduce the threshold condition "$0 \leq \beta \leq \beta^\star$" in the main text as a *conservative sufficient condition*. In practical reinforcement learning implementations, gradients are estimated from stochastic minibatches, so the alignment condition in (16) may only hold in expectation or approximately. A moderately bounded β helps prevent the variance of the MI gradient from overwhelming the gap gradient, thereby helping stabilize optimization.

A.4 Stackelberg Game Interpretation

Motivation. GEAR's training loop can be interpreted as a Stackelberg-style interaction between (i) the redundant goal generator $G^-_{\psi^-}$, which acts as an adversarial leader by producing challenging redundant goals, and (ii) the goal-conditioned actor π_ϕ, which acts as a follower by learning to separate efficient-goal-compatible actions from redundant-goal-compatible ones. This interpretation helps explain the emergent curriculum observed during training: the redundant generator is trained adversarially to synthesize challenging negatives, which in turn encourages the actor to learn a sharper value-based separation.

Role Definitions. We write ψ^+ and ψ^- for the parameters of the efficient and redundant generators, respectively.

Leader ($G^-_{\psi^-}$). For each anchor state s, the redundant generator samples $g^- \sim G^-_{\psi^-}(\cdot \mid s)$. Its goal is not to directly minimize the full actor gap. Instead, it produces hard redundant goals by reducing their value-space distinction from efficient goals under the efficient critic Q_θ.

Follower (π_ϕ). Given an efficient goal $g^+ \sim G^+_{\psi^+}(\cdot \mid s)$ and a redundant goal $g^- \sim G^-_{\psi^-}(\cdot \mid s)$, the actor maximizes the contrastive value gap, together with the MI regularizer that encourages goal-dependent behavior.

Efficiency Gap. For actor optimization, GEAR uses the learned value gap

$$\Delta Q(s, a \mid g^+, g^-) = Q_\theta(s, a \mid g^+) - Q_\omega(s, a \mid g^-)$$

This gap compares the same state-action pair under efficient and redundant goal conditions. It is used as an auxiliary signal for policy learning rather than as an absolute efficiency score.

Actor Objective. Following Eq. (11), the actor-side objective is

$$F(\phi, \psi^-) = \mathbb{E}_{s\sim\mathcal{D}} \mathbb{E}_{\substack{g^+\sim G^+_{\psi^+}(\cdot|s) \\ g^-\sim G^-_{\psi^-}(\cdot|s)}} \left[\Delta Q\big(s, \pi_\phi(s, g^+) \mid g^+, g^-\big)\right]$$

This is the same gap objective $\mathcal{J}_{\text{gap}}(\phi)$ used in the main text, with its dependence on the redundant generator parameters ψ^- made explicit. The same action $\pi_\phi(s, g^+)$ is evaluated under both goal conditions.

The follower then maximizes

$$\textbf{Follower (Actor):} \quad \max_\phi \ \left[F(\phi, \psi^-) + \mathcal{J}_{\text{MI}}(\phi)\right]$$

where, as defined in Eq. (13)

$$\mathcal{J}_{\text{MI}}(\phi) = \beta\, \mathbb{E}_{s\sim\mathcal{D}} \left[I_\phi(a; g^+ \mid s)\right]$$

Generator Objective. The redundant generator follows the asymmetric objective used in the main method. It is trained with the standard diffusion loss on redundant goals and an additional critic-guided adversarial loss:

$$\textbf{Leader (Generator):} \quad \min_{\psi^-} \ \mathcal{L}^-_{\text{DDPM}}(\psi^-) + \beta_t \lambda_{\text{adv}}\, \mathcal{L}^-_{\text{adv}}(\psi^-)$$

where

$$\mathcal{L}^-_{\text{adv}}(\psi^-) = \mathbb{E}_{(s,a)} \left[\big(Q_\theta(s, a \mid g^+) - Q_\theta(s, a \mid g^-)\big)^2\right]$$
$$g^+ \sim G^+_{\psi^+}(\cdot \mid s), \qquad g^- \sim G^-_{\psi^-}(\cdot \mid s)$$

This objective differs from directly minimizing the full actor gap $Q_\theta(s, a \mid g^+) - Q_\omega(s, a \mid g^-)$. Both goals are evaluated under the same efficient critic Q_θ so that $G^-_{\psi^-}$ learns to generate plausible hard negatives that are difficult to distinguish from efficient goals in value space.

Alternating Updates. The resulting interaction can be summarized by the alternating updates

$$\psi^- \leftarrow \psi^- - \eta_G \nabla_{\psi^-} \left[\mathcal{L}^-_{\text{DDPM}}(\psi^-) + \beta_t \lambda_{\text{adv}} \mathcal{L}^-_{\text{adv}}(\psi^-)\right]$$
$$\phi \leftarrow \phi + \eta_\pi \nabla_\phi \left[F(\phi, \psi^-) + \mathcal{J}_{\text{MI}}(\phi)\right]$$

The generator update makes redundant goals more competitive by reducing their value-space separation from efficient goals under Q_θ. The actor update optimizes the contrastive gap together with the MI regularizer, as in Eq. (14), encouraging the policy to recover separation under these adversarially generated negatives.

Curriculum Dynamics. This alternating pressure supports a self-paced curriculum. At the generator side, $G^-_{\psi^-}$ is trained adversarially to generate challenging redundant goals. At the actor side, π_ϕ learns to choose actions whose value is high under efficient goals but low under redundant goals. Thus, the generator challenges the actor with adversarially generated goals, while the actor is encouraged to reopen the value gap. Over training, this interaction is consistent with the increasing Q-gap and redundant-goal entropy observed in the curriculum analysis.

Interpretation Rather Than a Convergence Guarantee. The Stackelberg view is intended as a qualitative interpretation of the interaction between the redundant generator and the actor. It should not be read as a formal convergence guarantee for the nonconvex training dynamics. In particular, $G^-_{\psi^-}$ is not optimized as a direct minimizer of the actor objective $F(\phi, \psi^-)$. Instead, it is optimized through the asymmetric adversarial loss above, which produces hard negatives for the actor's gap-based policy optimization.

Conclusion.

- $G^{-}_{\psi^{-}}$ plays the leader-like role by generating challenging redundant goals through a critic-guided adversarial objective.
- π_{ϕ} plays the follower-like role by maximizing the learned efficiency gap under these adversarially generated negatives.
- Their interaction can support an emergent difficulty-adaptive curriculum aligned with learning progress.

B Additional Experiments

This section presents extended empirical evaluations to further validate the effectiveness, robustness, and deployment feasibility of GEAR. We begin with a real-world robotic manipulation study on an Elephant Robotics desktop manipulator, covering object lifting, block stacking, and target grasping tasks. We then report complete training curves across all DMC tasks and provide additional ablation studies to quantify the contributions of key components in our framework.

B.1 Real-World Robotic Manipulation

To examine whether the main components of GEAR remain effective under physical deployment, we conduct a real-world robotic manipulation study on an Elephant Robotics 280 desktop manipulator. This study is intended as a lightweight feasibility check rather than a full real-robot benchmark.

Platform. We deploy GEAR on an Elephant Robotics 280 desktop manipulator with 6-DoF control and a two-finger parallel gripper. The arm is base-mounted over a flat tabletop workspace. Perception uses an *eye-in-hand* RGB camera mounted on the wrist, while proprioceptive states, including joint positions, joint velocities, and gripper state, are always available. The policy outputs operational-space end-effector deltas, including translation, rotation, and gripper open/close commands, which are tracked by the vendor controller. Episodes terminate upon task success or when a fixed horizon is reached. Standard safety interlocks, including velocity limits and an emergency stop, are enabled during all physical trials.

Tasks. We evaluate three real-world manipulation tasks with increasing difficulty. In **Object Lifting**, the robot needs to grasp and lift a target object from the table. This task mainly tests basic goal-conditioned grasping under real visual observations and contact dynamics. In **Block Stacking**, the robot needs to grasp the purple block and place it on top of another object, requiring more accurate spatial control and contact stability. In **Target Grasping**, the robot needs to grasp the specified target object among multiple distractor objects. This task directly tests whether the policy can follow goal-conditioned visual cues instead of being distracted by nearby objects.



Figure 5: Real-world manipulation tasks on the Elephant Robotics. The figure provides the physical task setup.

Compared methods and ablations. We compare GEAR with the main baselines used in the simulation experiments, including SAC, PPO, HER, RIG, DD, and QVPO. These methods cover standard off-policy and on-policy reinforcement learning, hindsight goal relabeling, representation-based goal-conditioned learning, diffusion-based decision modeling, and value-guided policy optimization. To further isolate the contribution of GEAR's core components, we also evaluate two ablated variants. **GEAR w/o Actor Update via ΔQ.** This variant removes the gap-guided actor update and trains the policy only with the efficient-goal branch. It tests whether the value-gap objective is necessary for suppressing actions favored under redundant goals in real-world manipulation. **GEAR w/o**

Dual Critics. This variant replaces the dual-critic architecture with a single shared critic for both efficient and redundant goals. It tests whether separate value estimation is important for preserving a meaningful efficient–redundant value gap. All methods use the same robot platform, observation space, action interface, training budget, and evaluation protocol.

Training and evaluation. For each task, we collect a small set of real-world trajectories on the physical robot and use them to construct the goal-reaching dataset for training and adaptation. The two generator training sets are constructed by weighted sampling from these trajectories, following Section 3.1. For GEAR, these trajectories are used to train the efficient goal generator, redundant goal generator, dual critics, and goal-conditioned actor following the same objective used in simulation. For all baselines, we follow their corresponding training objectives while keeping the available robot data and training budget fixed. At test time, each method is deployed on the physical robot without additional online fine-tuning.

For each task and each method, we conduct 20 physical trials with randomized initial object positions. A trial is counted as successful if the robot completes the task within the maximum horizon. For **Object Lifting**, success requires the target object to be lifted from the table and remain stable for a short duration. For **Block Stacking**, success requires the purple block to be placed on the target object and remain stable after release. For **Target Grasping**, success requires the robot to grasp the specified target object rather than a distractor.

Table 5: Real-world robotic manipulation results on the Elephant Robotics platform. Success rate is computed over 20 physical trials for each task.

Method	Object Lifting	Block Stacking	Target Grasping
SAC	12/20 (60%)	7/20 (35%)	6/20 (30%)
PPO	11/20 (55%)	6/20 (30%)	5/20 (25%)
HER	14/20 (70%)	9/20 (45%)	9/20 (45%)
RIG	15/20 (75%)	10/20 (50%)	11/20 (55%)
DD	16/20 (80%)	11/20 (55%)	13/20 (65%)
QVPO	17/20 (85%)	13/20 (65%)	14/20 (70%)
GEAR w/o Actor Update via ΔQ	15/20 (75%)	11/20 (55%)	10/20 (50%)
GEAR w/o Dual Critics	16/20 (80%)	12/20 (60%)	12/20 (60%)
GEAR	18/20 (90%)	15/20 (75%)	16/20 (80%)

Failure analysis on target grasping. For Target Grasping, we further record whether a failure is caused by grasping a distractor object or by execution errors such as missing, dropping, or unstable grasping. This metric is particularly relevant because the task contains multiple visually plausible but incorrect objects.

Table 6: Failure analysis on Target Grasping over 20 physical trials. The wrong-target rate measures how often the robot grasps a distractor object.

Method	Wrong-Target Rate	Drop/Execution Failure
SAC	8/20 (40%)	6/20 (30%)
PPO	9/20 (45%)	6/20 (30%)
HER	6/20 (30%)	5/20 (25%)
RIG	5/20 (25%)	4/20 (20%)
DD	4/20 (20%)	3/20 (15%)
QVPO	3/20 (15%)	3/20 (15%)
GEAR w/o Actor Update via ΔQ	6/20 (30%)	4/20 (20%)
GEAR w/o Dual Critics	5/20 (25%)	3/20 (15%)
GEAR	1/20 (5%)	3/20 (15%)

As shown in Table 5, the full GEAR model achieves higher success rates than both ablated variants across all three real-world manipulation tasks. The improvement is relatively modest on Object Lifting, where the task mainly requires basic grasping. The gap becomes larger on Block Stacking, where precise spatial placement and contact stability are required. The largest improvement appears on Target Grasping, where distractor objects create ambiguous goal conditions. The failure analysis in Table 6 further shows that GEAR reduces wrong-target grasping, suggesting that the value-gap objective and dual-critic design help the policy distinguish task-relevant goals from redundant or misleading alternatives under real visual observations.

B.2 Scalability Analysis on Habitat



Figure 6: The visual observation of Habitat.

To further examine whether GEAR remains practical under larger model capacity and expanded goal spaces, we conduct a scalability analysis on Habitat Office. Habitat is a photorealistic embodied AI simulation platform built on real-world indoor 3D reconstructions, and the HM3D scenes used in our experiments provide building-scale scanned environments rather than simplified synthetic layouts. Thus, Habitat Office offers a realistic visual navigation setting for testing whether the proposed dual-generator and value-gap learning scheme can scale beyond low-dimensional control tasks. We vary the actor–critic capacity and the size of the achieved-goal pool used for training the diffusion goal generators, while keeping the training steps and evaluation protocol unchanged.

Table 7: Scalability analysis of GEAR on Habitat Office under increasing network capacity and goal-pool size. Results are reported over 5 seeds.

Setting	**Actor–Critic Width**	**Goal Pool Size**	**Success Rate**	**Peak Memory**
Base	256	10K	78.9 ± 3.2	10.8 GB
Larger Network	512	10K	79.6 ± 2.9	13.7 GB
Larger Goal Pool	256	50K	80.1 ± 2.7	12.1 GB
Larger Network + Goal Pool	512	50K	80.4 ± 2.8	15.6 GB
Largest Goal Pool	512	100K	79.8 ± 3.1	17.3 GB

As shown in Table 7, increasing the actor–critic capacity or enlarging the goal pool does not lead to severe performance degradation. The success rate remains close to the base setting, while memory usage grows moderately. These results indicate that GEAR is not restricted to shallow policies or small goal sets, and can remain practical in visually complex goal-conditioned environments.

B.3 Full Learning Curves

Figure 7 reports additional learning curves on eight DMC benchmark tasks beyond the six DMC tasks summarized in the main text. These additional results are included to provide a broader evaluation of GEAR across more diverse control settings. Across these tasks, GEAR generally achieves faster convergence and higher asymptotic returns, demonstrating strong sample efficiency and policy stability. The performance gap becomes more significant in complex tasks such as `humanoid-run`, where goal misalignment and behavioral inefficiency are more prominent. These results further support the advantage of contrastive efficiency modeling and structured goal generation.

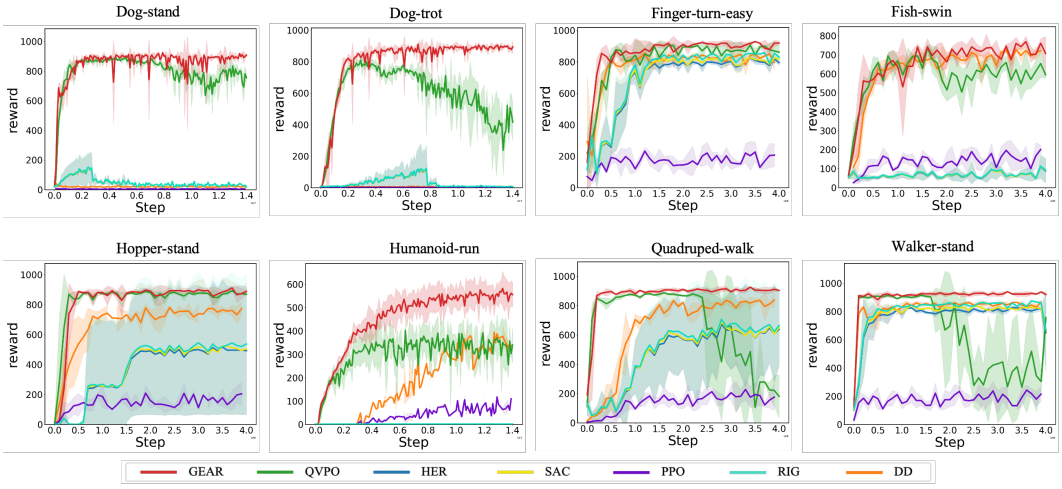

Figure 7: Additional learning curves of GEAR on eight DMC tasks beyond the six DMC tasks reported in the main text. Shaded areas denote standard deviation over five seeds.

B.4 Ablation Experiments

To analyze the design choices in GEAR, we perform systematic ablations by removing one component at a time. Table 8 reports the performance on five representative DMC tasks, showing a consistent drop in performance for all variants. Below we summarize the role and effect of each module:

- **w/o Red.**: Removing the redundant goal generator significantly reduces the efficiency contrast, weakening the policy's ability to distinguish between high- and low-value goals. This leads to less targeted learning signals and lower returns.
- **w/o Dua.**: Without the dual critic architecture, the model loses the capability to separately estimate values for efficient and redundant goals. As a result, the contrastive learning signal becomes noisy, degrading policy quality.
- **w/o Gap.**: Removing the gap-guided actor update leaves the policy without explicit contrastive supervision against actions favored under redundant goals, reducing episodic returns.
- **w/o MI**: Without mutual information regularization between goals and behaviors, the learned policy becomes less goal-consistent. This affects goal-reaching reliability, though the drop is relatively minor compared to structural modules.
- **w/o Diff.**: Replacing the diffusion-based generator with simpler goal sampling severely harms performance. The lack of diverse and behaviorally grounded goals results in inefficient exploration and less effective supervision.

In summary, each component contributes meaningfully to the final performance, with the diffusion generator and redundant goals being particularly critical. This highlights the necessity of contrastive supervision and structured goal synthesis in complex, long-horizon settings.

Method	Acrobo.	Cartpole-Swin	Cheetah-Run	Finger-Spin	Reacher-h.
GEAR	**192.6 ± 28.3**	**843.9 ± 27.4**	**680.1 ± 27.5**	**852.1 ± 11.6**	**951.2 ± 7.9**
w/o Red.	132.4 ± 30.1	772.1 ± 34.2	645.3 ± 28.7	810.5 ± 17.3	927.4 ± 11.5
w/o Dua.	120.3 ± 25.8	784.7 ± 29.6	638.9 ± 31.2	798.9 ± 16.4	915.6 ± 10.2
w/o Gap.	141.6 ± 23.7	801.6 ± 27.3	654.4 ± 29.1	819.3 ± 14.7	933.1 ± 8.4
w/o MI	180.2 ± 25.5	832.2 ± 24.8	676.0 ± 34.3	840.8 ± 10.3	945.5 ± 6.9
w/o Diff.	108.7 ± 21.9	751.3 ± 30.1	611.7 ± 25.5	785.1 ± 13.2	901.3 ± 9.7

Table 8: GEAR Ablation Study on DMC Environments (Episodic Return @4M steps, Mean ± Std over 5 Seeds)

C Experimental Implementation Details

This section provides additional implementation details to ensure the reproducibility of our experiments.

Component	Structure	Output	#Params
Actor $\pi_\phi(a \mid s, g)$	3-layer MLP, width 256	action mean + std	$\sim$322K
Efficient Critic $Q_\theta(s, a \mid g)$	3-layer MLP, width 256	scalar Q-value	$\sim$310K
Redundant Critic $Q_\omega(s, a \mid g)$	3-layer MLP, width 256	scalar Q-value	$\sim$310K
State Encoder $f_\eta(s)$	2-layer MLP, width 256	256-d state feature	$\sim$174K
Efficient Generator G_ψ^+	residual MLP, 6 layers, width 1024	predicted noise / sampled goal	$\sim$6.0M
Redundant Generator G_ψ^-	residual MLP, 6 layers, width 1024	predicted noise / sampled goal	$\sim$6.0M

Table 9: Network architectures and approximate parameter sizes used in GEAR. Parameter counts are reported for the default feature-based implementation and may vary slightly with state, action, and goal dimensions. For visual environments, raw observations are first encoded into compact state features before being passed to the actor, critics, and goal generators.

C.1 Network Architectures

We provide the architectural specifications and approximate parameter counts for the main neural components used in GEAR, including the actor, dual critics, state encoder, and goal generators. Unless otherwise specified, the actor and critics use ReLU activations and Xavier uniform initialization. For visual tasks, raw observations are first encoded into compact state features, and the actor, critics, and goal generators operate on these feature representations.

The actor and critics follow standard SAC-style MLP backbones. The efficient and redundant goal generators are implemented as conditional denoising diffusion models with the same residual-MLP backbone but different training objectives. Specifically, G_ψ^+ is trained on goals sampled with a preference for trajectories that successfully reach the goal in fewer steps, while G_ψ^- is trained on goals sampled with a preference for longer successful and failed trajectories and further shaped by the adversarial value-gap loss. Each generator uses random Fourier timestep embeddings, LayerNorm, and SiLU activations. During training, the denoising network predicts the injected noise; during inference, goals are produced through the reverse sampling process.

C.2 Training Hyperparameters

We summarize the training hyperparameters used in GEAR across all components, including the actor, two critic networks, and dual goal generators. Unless otherwise noted, we apply separate optimizers and learning rates for each network group. All experiments are conducted with 5 random seeds, and the best hyperparameter settings are selected based on the average validation return.

All critic networks (Q_θ, Q_ω) are trained independently using separate MLPs and separate Adam optimizers. This allows the redundant critic to model divergent value distributions when trained under distracting or adversarial goals g^-. All generators use a cosine learning rate schedule during offline pretraining. For all modules, weight initialization follows Xavier uniform. The β_t curriculum slope governs the weight of the adversarial value gap loss $\mathcal{L}_{\text{adv}}$. We use a simple linear schedule over the course of training:

$$\beta_t = \min\left(1.0, \beta_0 + \frac{t}{T_\beta}\right), \quad \beta_0 = 0.1,\ T_\beta = 10^6$$

This schedule ensures early-stage robustness while progressively increasing adversarial pressure. We evaluate each configuration over 5 seeds per task and report the best hyperparameter set based on mean success rate or return on validation tasks. Random seeds are fixed across all baselines and ablations.

Parameter	Search Space
Actor learning rate η_π	{1e-4, 3e-4, 1e-3}
Critic Q_θ learning rate η_{Q_1}	{1e-4, 3e-4, 1e-3}
Redundant Critic Q_ω learning rate η_{Q_2}	{1e-4, 3e-4, 1e-3}
Goal generator learning rate η_G	{1e-5, 1e-4, 3e-4}
Batch size	{128, 256, 512}
Discount factor γ	{0.98, 0.99, 0.995}
Target update rate τ	{0.005, 0.01}
Replay buffer size	{1e5, 1e6}
Reverse-sampling steps (N_{rev})	{4, 8, 16}
Adversarial weight λ_{adv}	{0.0, 2.5, 5.0, 10.0}
MI coefficient β	{0.0, 0.05, 0.1, 0.2}
Curriculum schedule slope (β_t)	Linear growth: $\beta_t = \min(1.0, 0.1 + t/10^6)$
Optimizer	Adam (separate for each module)
DDPM LR schedule	Cosine annealing
Training steps (actor/critic)	1-4M per run (14M for selected tasks)
Training steps (goal generators)	100K offline steps

Table 10: Hyperparameter search space.

C.3 Benchmark Environments

We benchmark GEAR across three families of goal-conditioned environments: *DMC* for low-dimensional sparse control, *ManiSkill2* for high-dimensional robotic manipulation, and *Habitat* for photorealistic indoor navigation. Each environment class is selected to stress-test a different capability of GEAR, including sparse exploration, semantic goal grounding, and generalization under perceptual noise and spatial complexity, as illustrated in Figure 8.

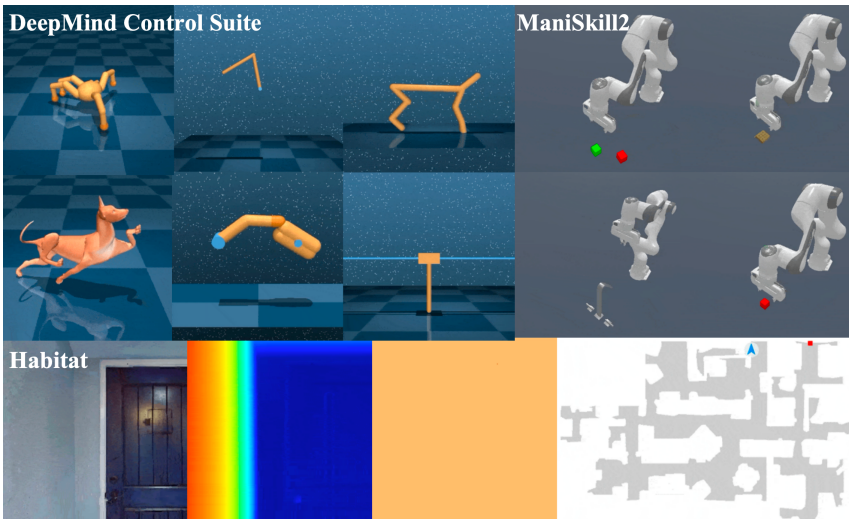

Figure 8: Illustration of the three benchmark environments: DMC, ManiSkill2, and Habitat, used to evaluate GEAR across control, manipulation, and navigation tasks.

DeepMind Control Suite (DMC). The DMC benchmark (Tassa et al., 2018) provides a standardized set of continuous control tasks with known reward structures. We select six sparse-reward tasks from DMC to test GEAR's ability to learn under minimal extrinsic feedback and over long horizons. Tasks vary in actuation complexity and dynamics:

- **Acrobot-Swingup** (obs: 6, act: 1) – a classic underactuated two-link task with delayed rewards.

- **Cartpole-Swingup** (obs: 5, act: 1) – control a cart to swing and balance a pole upright.
- **Cheetah-Run** (obs: 17, act: 6) – maximize forward velocity via periodic locomotion.
- **Finger-Spin** (obs: 9, act: 2, sparse) – maintain continuous rotation of a floating spinner.
- **Reacher-Hard** (obs: 6, act: 2, sparse) – reach distant targets in a planar workspace.
- **Walker-Walk** (obs: 24, act: 6) – learn coordinated bipedal gait and locomotion.

Goals are specified as target state vectors (e.g., joint angles or end-effector position). Rewards are binary or sparse-continuous, triggering only when the agent is sufficiently close to the goal. These tasks pose a significant challenge for exploration-based methods (e.g., PPO, SAC), which often fail to escape the initial low-reward region. GEAR addresses this through dual-critic supervision and pseudo-goal disambiguation. Each episode has a maximum horizon of 1,000 steps. We use fixed seeds $\{10, 20, 30, 40, 50\}$ and standardized random initialization.

ManiSkill2 Visual Manipulation. ManiSkill2 (Gu et al., 2023) contains a suite of realistic robotic manipulation tasks using RGB observations and continuous 6-DoF control. It evaluates an agent's ability to perceive, interpret, and interact with deformable or articulated objects. We consider five representative tasks:

- **Lift Cube** (obs: 42, act: 4) – lift a static cube vertically from a table surface.
- **Pick Cube** (obs: 51, act: 4) – grasp and lift a small cube with variable location and orientation.
- **Pick YCB** (obs: 51, act: 7) – pick an object from the YCB dataset with diverse geometries.
- **Stack Cube** (obs: 55, act: 4) – perform a two-step stack operation with coarse contact precision.
- **Turn Faucet** (obs: 40, act: 7) – rotate an articulated faucet by grasping its tip and turning to a goal angle.

These tasks test visual goal representation, temporal credit assignment, and robustness to semantic distractors. Goals are defined in visual or semantic embedding space (e.g., object mask centroids or task-specific pose states). Visual observations are provided as 84×84 RGB images, with high intra-task variability in object shape, color, and location. We use the `maniskill2==0.4.0` release with deterministic resets and 500-step episode limits.

Habitat Navigation. We adopt goal-conditioned navigation tasks from the Habitat 2.0 platform (Szot et al., 2021) using the HM3D dataset (Ramakrishnan et al., 2021). Agents are deployed in 3D indoor environments rendered from real-world building scans, receiving RGB-D input (256×256) and goal coordinates. The tasks require navigating from random start points to visually specified goals, without access to maps or global localization. We evaluate in three distinct environment types:

- **Residential:** indoor homes with tight corridors, clutter, and occlusions.
- **Office:** multi-room buildings with long hallways and repetitive structures.
- **Commercial:** wide open spaces such as malls or stores with diverse lighting and layout.

The agent must issue a `STOP` action when within 0.2 meters of the goal, under a 1,000-step cap. Goals are egocentric 3D vectors relative to the agent's initial pose. These tasks stress-test generalization under realistic perceptual noise and layout variability. Unlike DMC or ManiSkill2, the scenes vary across training and validation, making offline overfitting more likely for diffusion-based policies. GEAR leverages contrastive supervision and curriculum from redundant goals to mitigate such overfitting.

C.4 Diffusion Goal Generator Training Details

We employ conditional denoising diffusion probabilistic models (DDPMs) to parameterize the *Efficient-Goal Generator* G_ψ^+ and the *Redundant-Goal Generator* G_ψ^-. Each generator is trained on

the corresponding training set $\mathcal{D}^+$ or $\mathcal{D}^-$ constructed by weighted sampling from the replay buffer (Section 3.1), so that the two models later provide a contrastive curriculum.

The base distribution p_0 samples uniformly from valid trajectory records for each task; branch probabilities follow Eq. (2). For a sampled trajectory, we draw t uniformly from $\{0, \dots, T-1\}$ and then draw j uniformly from $\{t+1, \dots, T\}$. We pair s_t with the task's goal representation of s_j. This defines the shared pairing rule ρ. Only trajectories that admit such pairs are used, and positive-branch updates start once successful trajectories are available.

Architecture. Both G_ψ^+ and G_ψ^- are **6-layer residual MLPs** with width 1024, matching the specification in Table 9. The input is a corrupted goal vector $x_t \in \mathbb{R}^{d_g}$ at diffusion timestep t. The generator conditions on the current state representation, denoted by cond below, as in Eq. (3). The local diffusion index t corresponds to ℓ in that equation. The network uses (i) the current state embedding $f_\eta(s_t) \in \mathbb{R}^{256}$ and (ii) a timestep embedding $e(t) \in \mathbb{R}^{256}$ obtained via **random Fourier features**. Each hidden layer follows the pre-activation layout `LayerNorm` $\rightarrow$ `SiLU` $\rightarrow$ `Linear`, and a residual skip is added after every layer. This design follows the residual-MLP diffusion setup of (Lu et al., 2023b) and the training protocol of (Ding et al., 2024), resulting in $\sim$6 M parameters per generator.

Diffusion process. During offline training we use the standard DDPM forward process with a linear noise schedule $\beta_t \in [10^{-4}, 0.02]$ over a fixed $T = 1000$ steps (Ho et al., 2020). The generator is trained to predict the added noise ϵ via

$$q(x_t \mid x_0) = \mathcal{N}\big(x_t; \sqrt{\bar{\alpha}_t}\, x_0,\, (1-\bar{\alpha}_t)\mathbf{I}\big) \qquad \mathcal{L}_{\mathrm{DDPM}}(\psi) = \mathbb{E}_{x_0,\epsilon,t}\big[\|\epsilon - \epsilon_\psi(x_t, t, \mathrm{cond})\|_2^2\big]$$

where $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$.

Reverse-sampling steps. At action-time the reverse process is *sub-sampled* to $N_{\text{rev}} \in \{4, 8, 16\}$ steps (see Table 10); we adopt a DDIM-style uniform stride through $\{T, \dots, 0\}$. The default choice is N_{rev}=8, yielding $\approx$1.2 ms per goal on an RTX-3090.

Curriculum sampling. We use a two-stage curriculum. During the first 20% of training ($\beta_t < 0.5$) we draw goals from large timesteps $t > 700$ *and* use the maximal N_{rev}=16, promoting high-entropy exploration. Afterwards we linearly anneal both the timestep window and N_{rev} toward the value chosen from the hyper-parameter search (default 8), providing sharper, task-relevant goals. The adversarial weight β_t follows the linear schedule defined in C.2.

Training details. Each generator is trained offline for 1×10^5 **iterations** using Adam (lr $= 1\times10^{-4}$, batch $= 256$) and a cosine learning-rate schedule. Inputs are normalised to zero mean and unit variance; early stopping is triggered by reconstruction loss on a held-out validation split.

Sampling during policy training. For every actor update we sample 20 efficient and 20 redundant goals with ancestral sampling (classifier-free guidance weight 1.5) using the current N_{rev}. The resulting value gap drives the efficiency-contrastive actor update (Eq. (11)) and provides a dynamically paced curriculum throughout learning.

C.5 Implementation and Reproducibility

We release the full implementation of GEAR upon publication, including the training pipeline, goal generation modules, evaluation scripts, configuration files, and logging utilities. Our implementation is built on top of widely used public frameworks to ensure compatibility with prior work and reproducibility across domains. These codes can be found on GitHub. Below we list key dependencies and their roles:

- **RLKit** (rail-berkeley/rlkit) Used as the backbone for goal-conditioned SAC, experience replay, and policy optimization modules.
- **Decision Diffuser (DD)** (anuragajay/decision-diffuser) Provides the architectural and training framework.

- **QVPO** (wadx2019/qvpo) Integrated as a diffusion-based baseline using Q-weighted variational policy optimization. We align training protocols and evaluation metrics with QVPO for fair comparison.
- **Stable-Baselines3 (SB3)** (DLR-RM/stable-baselines3) Employed for on-policy baselines such as PPO and for unified Gym-compatible environment wrappers across domains.
- **DeepMind Control Suite** (deepmind/dm_control) Utilized for benchmarking continuous control algorithms on a standard set of physics-based tasks powered by the MuJoCo engine.
- **ManiSkill2** (haosulab/ManiSkill2) A large-scale benchmark for generalizable robotic manipulation skills. We use it to evaluate our agent's performance on complex, long-horizon manipulation tasks.
- **Habitat** (facebookresearch/habitat-lab) Used for embodied AI tasks such as point-goal navigation and object-goal navigation in realistic 3D environments. Our experiments leverage both Habitat Lab for task definition and Habitat Sim for high-performance simulation.

All experiments were conducted on a shared compute cluster with NVIDIA RTX 3090 and A100 GPUs. Each run used a single GPU, with accompanying hardware including Intel Xeon Gold CPUs and 128GB RAM. Training GEAR on a single task for 2-4 million environment steps takes 3–4 hours on DMC, 8–10 hours on ManiSkill2, and 12–14 hours on Habitat. All results are averaged over 5 seeds. Total training across benchmarks consumed roughly 540 GPU-hours, with an additional 40 GPU-hours used for evaluation and curriculum tracking. Diffusion-based goal generation adds modest overhead in the full training pipeline, but remains practical due to efficient batch-parallel sampling. We fix all random seeds and disable non-deterministic CUDA kernels across PyTorch, NumPy, and simulation environments to ensure determinism. Detailed instructions for deterministic reproduction, figure generation, and log inspection are included in the supplementary repository, along with a `requirements.txt`, and per-task YAML configuration files.

C.6 Mutual Information Regularization

We implement the mutual information regularizer in Eq. (13) using a conditional InfoNCE lower bound. For each state s sampled from the replay buffer, we independently draw M goals $g_i^+ \sim G_\psi^+(\cdot \mid s)$ and sample a corresponding action $a_i \sim \pi_\phi(\cdot \mid s, g_i^+)$ for each goal. Since the stochastic actor provides a tractable action density, we use its log-density to score each action–goal pair:

$$\ell_{ij} = \log \pi_\phi(a_i \mid s, g_j^+)$$

The resulting contrastive loss is

$$\mathcal{L}_{\mathrm{NCE}}(\phi) = -\mathbb{E}\left[\frac{1}{M}\sum_{i=1}^{M} \log \frac{\exp(\ell_{ii})}{\sum_{j=1}^{M}\exp(\ell_{ij})}\right]$$

where the expectation covers the sampling of states, goals, and actions. For each action a_i, the matched goal is g_i^+, while the remaining goals sampled at the same state serve as contrastive candidates. Following the tractable-conditional InfoNCE formulation of Poole et al. (2019), this sampling procedure yields

$$I_\phi(A; G^+ \mid S) \geq \log M - \mathcal{L}_{\mathrm{NCE}}(\phi)$$

We therefore optimize the actor by minimizing

$$\mathcal{L}_{\mathrm{actor}}(\phi) = -\mathcal{J}_{\mathrm{gap}}(\phi) + \beta \mathcal{L}_{\mathrm{NCE}}(\phi)$$

where the constant term involving $\log M$ is omitted. During actor updates, the goal generators are held fixed, and gradients are propagated through reparameterized action samples and their log-densities. This implementation requires no additional MI estimation network. All contrastive goal candidates are sampled from G_ψ^+; the redundant goals from G_ψ^- are used separately in the value-gap objective. Theorem 2 characterizes the ideal MI objective. Its gradient argument also applies to the implemented lower-bound objective when the corresponding surrogate gradient satisfies the same form of alignment condition.

D USE OF LLM

Large language models (LLMs) were used as assistive tools for text editing and improving the clarity of exposition. They were not involved in the design of algorithms, implementation, or experimental analysis. All technical content, theoretical results, and experimental findings were produced and verified by the authors. We take full responsibility for the content of this paper. We only used LLMs for language polishing.