Guide to Control: Offline Hierarchical Reinforcement Learning Using Subgoal Generation for Long-Horizon and Sparse-Reward Tasks

Wonchul Shin and **Yusung Kim***
Sungkyunkwan University
{swc0406, yskim525}@skku.edu

Abstract

Reinforcement learning (RL) has achieved considerable success in many fields, but applying it to real-world problems can be costly and risky because it requires a lot of online interaction. Recently, offline RL has shown the possibility of extracting a solution through existing logged data without online interaction. In this work, we propose an offline hierarchical RL method, Guider (Guide to Control), that can efficiently solve long-horizon and sparse-reward tasks from offline data. The high-level policy sequentially generates a subgoal that can guide the agent to arrive at the final goal, and the lower-level policy learns how to reach each given guided subgoal. In the process of learning from offline data, the key is to make the low-level policy reachable to the generated subgoals. We show that high-quality subgoal generation is possible through pre-training a latent subgoal prior model. The well-regulated subgoal generation improves performance while avoiding distributional shifts in offline RL by breaking down long, complex tasks into shorter, easier ones. For evaluations, Guider outperforms prior offline RL methods in long-horizon robot navigation and complex manipulation benchmarks. Our code is available at https://github.com/gckor/Guider.

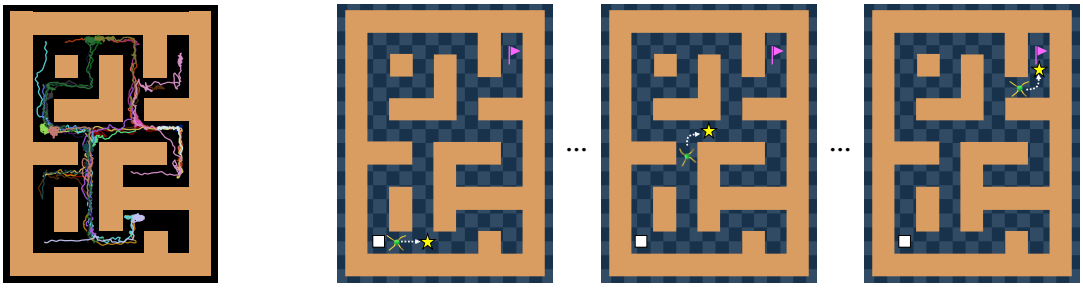
1 Introduction

Deep reinforcement learning (RL) has achieved remarkable success in a range of domains, such as robotics [Kalashnikov *et al.*, 2018], games [Silver *et al.*, 2016], and autonomous driving [Balaji *et al.*, 2019]. However, these works require a large amount of online interaction with the environment, and in real-world applications, online interaction may be limited due to high risk and cost. To address this problem, offline RL methods have emerged, which use previously logged data without online interaction. Unlike online RL, the value overestimation problem on out-of-distribution actions can be fatal because correcting the overestimation is impossible in an offline setting[Fujimoto *et al.*, 2019]. Recent studies have produced promising results via various regularization techniques which conservatively mitigate the distributional shift problems [Wu *et al.*, 2019; Kumar *et al.*, 2020; Kostrikov *et al.*, 2021].

In the real world, however, we face difficulties of complex and long-horizon tasks with sparse rewards which are still challenging to solve with conventional offline RL algorithms. Previous studies have shown the effectiveness of breaking down these long and complex problems into simpler subtasks with a hierarchical structure for online RL [Nachum *et al.*, 2018; Zhang *et al.*, 2020; Bagaria *et al.*, 2021]. Among the various design of hierarchy, goal-conditioned hierarchical RL allows a high-level policy to sequentially generate subgoals at regular intervals and a low-level policy to learn to reach the generated subgoals. A key element of a subgoal-based hierarchical RL is stable and compatible training between the high-level and low-level policies [Wang *et al.*, 2022]. Specifically, a high-level policy should provide reasonable subgoals that a low-level policy can easily reach. In a fully offline setting, it is difficult to determine whether a low-level policy actually reaches a subgoal generated by a high-level policy during the learning process due to the absence of direct validation in the environment. In addition, offline RL learns from fixed datasets which may include a mix of task-agnostic or sub-optimal trajectories [Fu *et al.*, 2020], while online RL can update a policy or a value function with sufficient exploration for a specific task.

In this work, we propose a novel offline hierarchical RL algorithm, Guider, to address the above challenges. Our primary contribution is pre-training a latent variable model to extract the latent distribution of reachable subgoals from an offline dataset. With the pre-trained prior distribution model, we can effectively apply additional regularization while training the subgoal generation policy. Within the constraints of the reachable subgoal distribution in latent space, the high-level policy learns to generate subgoals that ensure a gradual approach to the final goal while maximizing the cumulative rewards. The low-level policy learns to reach a subgoal generated by the higher level through relabeled rewards of hindsight goal relabeling. We benchmark our method on a variety of simulated continuous control tasks including robot locomotion, navigation, and manipulation. Extensive experiments show that our subgoal generation method can successfully break down a complex and long-horizon task into eas-

*Corresponding author



(a) Offline dataset

(b) Illustrative example of subgoal generation

Figure 1: (a) Task-agnostic (random starts and goals) trajectories in the Antmaze-large offline dataset. (b) An example of the subgoal generation scenario. Guider learns hierarchical policies from the offline data. For a new longer trajectory test (unseen in the offline dataset), a high-level policy can sequentially generate a subgoal (the star mark) toward the new final goal (the flag mark) and a low-level policy can focus on reaching each generated subgoal.

ier short-horizon goal-reaching problems. In most evaluation tasks, Guider outperforms prior offline RL baselines.

2 Related Works

2.1 Offline Reinforcement Learning

Offline RL methods mainly tackle the distributional shift issue between a learned policy and a behavior policy that is utilized to collect an offline dataset. Prior works constrain the learned policy to stay close to the offline data distribution via explicit regularization [Wu *et al.*, 2019; Fujimoto *et al.*, 2019], conservative value learning [Kumar *et al.*, 2020], and importance sampling [Nachum *et al.*, 2019]. Several studies employ advantage-weighted supervised learning approach, which implicitly imposes constraints to the offline dataset. Compared with other types of regularization methods, these implicit constraints avoid excessive conservative updates and allow effective online fine-tuning [Nair *et al.*, 2020; Kostrikov *et al.*, 2021]. Recently, [Yang *et al.*, 2022] has shown promising results on multi-goal manipulation tasks by adapting the advantage-weighted method coupled with detailed weighting scheme and hindsight relabeling. Although previous methods effectively address the distributional shift problem, a substantial challenge of accomplishing long-horizon and complex tasks, particularly with sparse reward, still remains. We propose a hierarchical approach in offline RL using subgoal generation which efficiently improves performance on such challenging problems. From the perspective of a high-level policy, we impose additional regularization toward prior distribution that has been pre-trained via unsupervised learning, in order to avoid generating infeasible subgoals. At the same time, a low-level policy learns to reach relatively short-horizon subgoals with an implicit regularization.

2.2 Hierarchical Reinforcement Learning

Hierarchical reinforcement learning (HRL) aims to solve complex long-horizon tasks by breaking them into more tractable subtasks with multi-level policies. In particular, goal-conditioned HRL in which a high-level policy presents subgoals has shown great potential in a variety of sparse reward problems. The effectiveness of goal-conditioned HRL relies on the reasonable subgoal generation, *i.e.*, the high-level policy should provide subgoals that the low-level policy can reach. Recent studies have improved the learning efficiency of goal-conditioned HRL by employing hindsight correction [Nachum *et al.*, 2018; Levy *et al.*, 2019], adversarial learning [Wang *et al.*, 2022], and representation learning of subgoal with contrastive objective [Li *et al.*, 2021] under interaction with the environment. However, these works need extensive online interaction with the environment, which could be expensive and risky in real-world applications. In contrast, our method focuses on training with only offline data including trajectories that may be irrelevant to the evaluation tasks or collected by sub-optimal policies, without additional access to the environment.

Some recent studies proposed offline HRL methods using temporally extended skills [Lynch *et al.*, 2020; Ajay *et al.*, 2021]. These hierarchical skill learning methods train low-level action policies as reconstruction-based decoders of state-action sequences. These works are similar to our method of utilizing unsupervised pre-training and high-level training in latent space. However, we separately train low-level policy via value-based reinforcement learning, which provides a considerable advantage when using sub-optimal mixed data.

3 Preliminaries

We consider a continuous control problem formulated as a goal-conditioned Markov decision process (MDP), denoted by a tuple $(\mathcal{S}, \mathcal{G}, \mathcal{A}, p, r, \gamma)$, where $\mathcal{S}$ is a state space, $\mathcal{G}$ is a goal space, $\mathcal{A}$ is an action space, $p(s'|s,a,g)$ is a transition probability, $r(s,a,g)$ is a reward function, and $\gamma \in (0,1]$ is a discount factor. The objective is to obtain an optimal policy π that maximizes the expected discounted return $\mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_t]$. The goal-conditioned task generally provides sparse reward and the reward function is defined as:

$$r(s_t, a, s_{t+1}, g) = \begin{cases} 1 & \text{if } \|s^{\mathcal{G}}_{t+1} - g\|_2 < \epsilon \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $s^{\mathcal{G}} \in \mathcal{G}$ is a state that is mapped to goal space, and ϵ is a given threshold from the environment. The goal space is defined according to the environment and can be a subset of

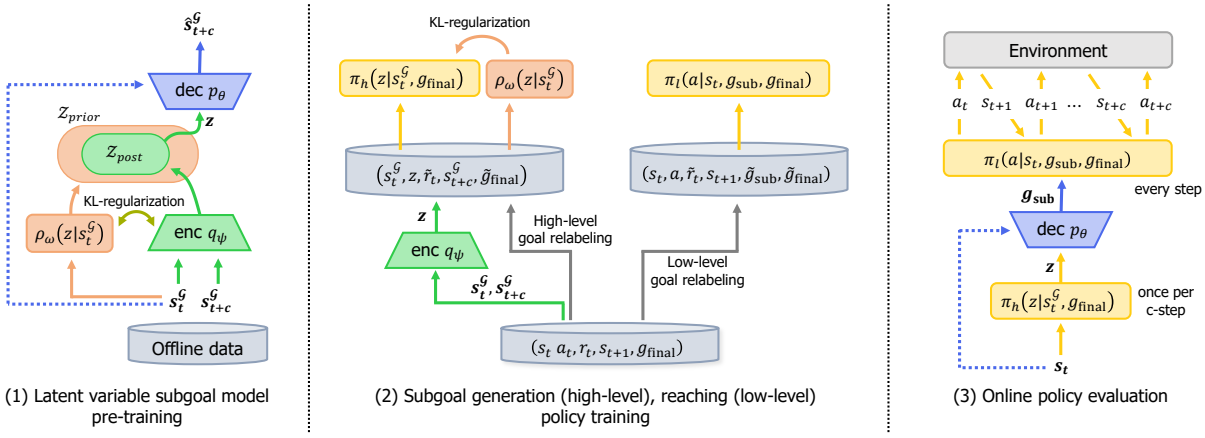


Figure 2: Overall framework of Guider. A latent variable model is pre-trained to embed reachable subgoals into a latent space. A high-level policy is trained to generate a reachable subgoal while regularized with a pre-trained prior model. A low-level policy learns to reach a generated subgoal. In online evaluation, the high-level policy generates a subgoal every c steps to guide the low-level policy.

the state space. For instance, the goal space of a robot navigation task is given as a location of the robot torso, while the state space is composed of tens of proprioceptive features (e.g. coordinates, angles, velocities). For offline RL settings, we assume that the agent uses only previously collected data without any interaction with the environment during the training process.

We consider the goal-conditioned HRL framework consisting of a high-level policy $\pi_h(z|s, g_{\text{final}})$ and a low-level policy $\pi_l(a|s, g_{\text{sub}}, g_{\text{final}})$, where z is a subgoal in a latent space, g_{sub} is a subgoal decoded into a goal space, and g_{final} is the final goal of the downstream task. The high-level policy generates a subgoal every c-steps, and the low-level policy executes an action in each of the steps to reach the generated subgoal. In the following section, we show how to train the latent variable model which captures the reachable subgoals within c-steps from the current state, and utilizes the pre-trained model to regularize the subgoal generation policy.

4 Method

In this section, we describe our method Guider, which trains subgoal generation policy within the constraints of reachable subgoal distribution in latent space. Our objective is to ensure the low-level policy easily reaches a subgoal generated by the high-level policy, while both policies are trained to avoid distributional shifts from the offline dataset without any additional data collection or online interaction. We first introduce how the overall framework is structured, and elaborate on each training part.

4.1 Model Overview

Our model consists of three parts of training: (1) pre-training the latent variable model which embeds reachable subgoals in latent space via unsupervised learning (2) training the high-level policy to generate a subgoal that progressively approaches a given final goal (3) training the low-level policy that reaches the subgoal generated at the high-level. We train the latent variable model comprised of encoder $q_\psi(z|s_t^\mathcal{G}, s_{t+c}^\mathcal{G})$, decoder $p_\theta(s_{t+c}^\mathcal{G}|s_t^\mathcal{G}, z)$, and trainable prior model $\rho_\omega(z|s_t^\mathcal{G})$. We relabel the offline dataset with the pre-trained encoder $q_\psi(z|s_t^\mathcal{G}, s_{t+c}^\mathcal{G})$ for the high-level policy training, and the high-level policy $\pi_h(z|s_t^\mathcal{G}, g_{\text{final}})$ is regularized with the pre-trained prior distribution $\rho_\omega(z|s_t^\mathcal{G})$. The low-level policy $\pi_l(a|s_t, g_{\text{sub}}, g_{\text{final}})$ is trained by relabeled rewards for reaching the subgoals. At test time, the high-level policy generates a subgoal every c-steps, and the low-level policy selects an action for each step toward the generated subgoal. The architecture of our model is illustrated in Figure 2.

4.2 Latent Variable Model for Reachable Subgoals

We train the latent variable model in order to learn the subgoal generation policy in the latent space of subgoals that can be reached within c-steps from the current state, and not in the entire goal space. We sample pairs of random states and c-step next states from the offline dataset and train the encoder $q_\psi(z|s_t^\mathcal{G}, s_{t+c}^\mathcal{G})$ and the decoder $p_\theta(s_{t+c}^\mathcal{G}|s_t^\mathcal{G}, z)$ to reconstruct the next state in a goal space after c-steps, while conditioned on the current states. Additionally, we train subgoal prior model $\rho_\omega(z|s_t^\mathcal{G})$ to obtain the latent distribution when a current state is given. We jointly train the components of the latent variable model by maximizing the following objective function based on evidence lower bound (ELBO):

$$\mathbb{E}_{z\sim q_\psi}[\log p_\theta(s_{t+c}^\mathcal{G}|s_t^\mathcal{G}, z) - \beta D_{\text{KL}}(q_\psi(z|s_t^\mathcal{G}, s_{t+c}^\mathcal{G})\|\rho_\omega(z|s_t^\mathcal{G}))]. \tag{2}$$

The subgoal prior model is trained to cover the latent distributions of reachable subgoals from the current state, and the posterior distribution from the encoder is regularized toward the prior by the KL loss term. β is a weighting parameter for the regularization. We implement a balanced training of both parameterized prior and posterior networks with alternately stopped gradient flow at different learning rates following [Hafner *et al.*, 2020]:

$$D_{\text{KL}}(q_\psi\|\rho_\omega) = \tau D_{\text{KL}}(\text{sg}(q_\psi)\|\rho_\omega) + (1-\tau)D_{\text{KL}}(q_\psi\|\text{sg}(\rho_\omega)) \tag{3}$$

where $\mathrm{sg}(\cdot)$ indicates a stopped gradient flow and τ is the weighting parameter.

4.3 High-level Subgoal Generation Policy

We apply the pre-trained latent variable model to train the high-level subgoal generation policy $\pi_h(z|s^{\mathcal{G}}, g_{\text{final}})$. In the beginning, we relabel the offline dataset using the trained encoder. An action a_t is replaced by an embedded latent subgoal $z \sim q_\psi(\cdot|s_t^{\mathcal{G}}, s_{t+c}^{\mathcal{G}})$. The next state of the high-level transition is relabeled as the state mapped to the goal space after c-step $s_{t+c}^{\mathcal{G}}$. In addition, we relabel the original goal with a random future state after the c-step, employing hindsight relabeling which improves the learning efficiency of a goal-conditioned policy [Andrychowicz *et al.*, 2017]. The relabeled reward $\tilde{r}(s_t^{\mathcal{G}}, z, s_{t+c}^{\mathcal{G}}, \tilde{g}_{\text{final}})$ for relabeled goal $\tilde{g}_{\text{final}}$ is computed by equation 1. Consequently, an original transition tuple $(s_t, a_t, r_t, s_{t+1}, g_{\text{final}})$ in the offline dataset is relabeled to $(s_t^{\mathcal{G}}, z_t, \tilde{r}_t, s_{t+c}^{\mathcal{G}}, \tilde{g}_{\text{final}})$.

We optimize the subgoal generation policy $\pi_h(z|s^{\mathcal{G}}, g_{\text{final}})$ to maximize value function $Q(s^{\mathcal{G}}, z, g_{\text{final}})$ while regularized toward the pre-trained subgoal prior $\rho_\omega(z|s^{\mathcal{G}})$ following the proposed objective:

$$\mathbb{E}_{z\sim q_\psi}[Q(s^{\mathcal{G}}, z, g_{\text{final}}) - D_{\text{KL}}(\pi_h(z|s^{\mathcal{G}}, g_{\text{final}})\|\rho_\omega(z|s^{\mathcal{G}}))]. \tag{4}$$

Minimization of the reverse KL divergence constrains the learned policy to generate subgoals within a learned subgoal space that is reachable in c-steps from the current state. Our regularized policy optimization objective can be applied to an off-the-shelf offline RL algorithm. Here, we opt for Conservative Q-Learning (CQL), an extensively used offline actor-critic algorithm [Kumar *et al.*, 2020]. A maximum entropy regularization term $\min \log \pi_h(z|s, g_{\text{final}})$ of the original CQL policy objective which encourages diverse behaviors is replaced with our KL regularization. This regularization plays an important role in learning hierarchical policies in offline conditions where it is impossible to know whether the low-level policy can reach the generated subgoals. We will show how the additional regularization improves the quality of the generated subgoals and resulting performance in section 5.

4.4 Low-level Subgoal Reaching Policy

Our subgoal generation method can be used in conjunction with any low-level goal-conditioned policy learning algorithm such as goal-conditioned supervised learning (GCSL) [Ghosh *et al.*, 2021] or goal-conditioned version of existing offline RL methods. Since our hierarchical framework decomposes complex tasks into relatively easy short-horizon goal-reaching problems, the lower-level policy can be learned effectively with a simple policy learning algorithm. We introduce a simple advantage-weighted supervised learning method adapted for our goal-conditioned hierarchical framework.

The lower-level policy $\pi_l(a|s, g_{\text{sub}}, g_{\text{final}})$ should learn to choose an optimal action to reach a guided subgoal g_{sub} while avoiding the distributional shift problem of offline RL. Therefore, we update the policy by maximizing advantage $A(s, a, g_{\text{sub}}, g_{\text{final}})$ with KL divergence regularization toward the behavior policy $\pi_\beta(s, a, g_{\text{sub}}, g_{\text{final}})$, using the following optimization formulation:

$$\max \mathbb{E}_{a\sim\pi_l}[A(s, a, g_{\text{sub}}, g_{\text{final}})] \quad \text{s.t.}\, D_{\text{KL}}(\pi_l\|\pi_\beta) < \epsilon \tag{5}$$

where ϵ is a threshold value. The above constraint optimization problem is solved by enforcing the Karush-Kuhn-Tucker conditions following prior works [Peng *et al.*, 2019; Nair *et al.*, 2020]. The implicitly constrained policy optimization objective is derived as:

$$\mathbb{E}_{a\sim\pi_l}[\exp A(s, a, g_{\text{sub}}, g_{\text{final}}) \cdot \log \pi_l(a|s, g_{\text{sub}}, g_{\text{final}})]. \tag{6}$$

We train the low-level policy with goal-relabeled data similar to the high-level policy training. However, the relabeling strategy is slightly different considering our subgoal generation framework. We relabel the subgoal g_{sub} to random future states within c-steps as $\tilde{g}_{\text{sub}} = s_i^{\mathcal{G}}$ for $t < i \leq t + c + 1$, since our high-level policy is trained to generate subgoals reachable within c-steps. This modified relabeling strategy effectively accelerates training the low-level policy. The whole training process of our proposed method is summarized in Algorithm 1.

Algorithm 1 Guider

Require: Dataset $\mathcal{D}$, subgoal generation period c.
Initialize: Encoder q_ψ, decoder p_θ, prior ρ_ω, high-level policy π_h, high-level value function Q_h, low-level policy π_l, low-level value function Q_l.

for M iterations **do**
  Sample mini-batch: $\{(s_t^{\mathcal{G}}, s_{t+c}^{\mathcal{G}})\} \sim \mathcal{D}$
  Update ψ, θ, ω using objective in eq. 2
end for
for N iterations **do**
  # High-level policy training
  Sample mini-batch: $\{(s_t, s_{t+c}, g_{\text{final}})\} \sim \mathcal{D}$
  $z \sim q_\psi(s_t^{\mathcal{G}}, s_{t+c}^{\mathcal{G}})$
  $\tilde{g}_{\text{final}} \sim s_i^{\mathcal{G}} \quad (t + c < i \leq T)$
  Relabel the mini-batch: $\{(s_t^{\mathcal{G}}, z, \tilde{r}_t, s_{t+c}^{\mathcal{G}}, \tilde{g}_{\text{final}})\}$
  Update Q_h to minimize TD error
  Update π_h using objective in eq. 4
  # Low-level policy training
  Sample mini-batch: $\{(s_t, a_t, r_t, s_{t+1}, g_{\text{final}})\} \sim \mathcal{D}$
  $\tilde{g}_{\text{sub}} \sim s_j^{\mathcal{G}} \quad (t < j \leq t + c)$
  Relabel the mini-batch: $\{(s_t, a, \tilde{r}_t, s_{t+1}, \tilde{g}_{\text{sub}}, \tilde{g}_{\text{final}})\}$
  Update Q_l to minimize TD error
  Update π_l using objective in eq. 6
end for
return Trained policies π_h, π_l.

5 Experiments

Through our experiments, we aim to answer the following questions: (1) Can Guider be effectively applied to a variety of long-horizon and sparse-reward tasks? (2) Can Guider learn stitching knowledge from a fixed dataset including task-agnostic trajectories or mixed sub-optimal behaviors? (3) Does our proposed latent subgoal prior model help generate

	OPAL[1]	GCSL	WGCSL	CQL+HER	Guider (Ours)
Antmaze-umaze	-	49.5±5.9	79.5±2.3	60.7±7.6	**88.5±4.4**
Antmaze-medium	81.1±3.1	45.3±13.7	54.0±8.8	28.3±5.3	**87.3±0.4**
Antmaze-large	70.3±2.9	11.8±3.6	17.5±6.7	11.3±8.2	**80.8±4.6**
FetchReach-expert	46.0±0.1	39.8±0.4	46.3±0.2	47.3±0.2	**48.0±0.1**
FetchPush-expert	18.9±0.4	32.8±1.6	38.9±0.4	38.5±1.2	**39.9±1.0**
FetchSlide-expert	0.7±0.1	4.4±1.4	**10.6±0.9**	3.9±1.7	7.6±0.3
FetchPick-expert	24.2±2.6	27.2±0.5	36.1±0.5	36.9±2.3	**40.9±0.6**
FetchReach-mixed	45.2±0.2	34.5±1.0	46.7±0.2	46.6±0.5	**47.0±0.2**
FetchPush-mixed	5.5±2.5	6.1±2.1	30.8±3.2	21.8±3.8	**33.4±2.3**
FetchSlide-mixed	0.7±0.2	1.9±0.7	**5.6±1.0**	3.7±1.2	4.1±0.6
FetchPick-mixed	5.9±3.0	12.0±3.5	22.9±5.9	25.8±6.9	**33.2±2.4**
Kitchen-complete	-	58.6±8.7	57.7±4.7	33.3±5.7	**68.8±7.2**
Kitchen-partial	65.2±2.5	55.0±14.5	59.4±13.3	26.3±4.5	**70.4±7.8**
Kitchen-mixed	64.6±1.8	56.2±5.4	49.6±2.9	23.5±1.1	**67.1±5.8**

Table 1: Performance of Guider and baselines on all tasks. The scores are averaged over 4 random seeds at the end of training. For every seed, we run an evaluation of 100 episodes.

a reasonable subgoal and improve the performance of offline hierarchical RL? To answer the first and second questions, we evaluate the Guider against a variety of benchmarks dealing with stitching and narrow distribution challenges of offline RL. We present ablation studies to answer the third question.

5.1 Environments and Datasets

We evaluate our method on diverse continuous control tasks in three simulated robotic domains: maze navigation, arm manipulation, and kitchen manipulation. All these environments provide sparse reward feedback.

Antmaze

The antmaze domain requires an agent to control a quadruped robot to navigate to a designated goal position. We use *diverse* dataset from the D4RL benchmark [Fu *et al.*, 2020]. The dataset is composed of trajectories from random start locations to random goal positions, which are irrelevant to the target task. Therefore, the agent is required to stitch the parts of suboptimal trajectories to find the optimal path to the evaluation goal. The domain contains three maze layouts (*umaze*, *medium*, *large*). In particular, the long-horizon *large* task is substantially challenging to solve with a conventional offline RL approach.

Fetch

The fetch domain contains four robotic arm manipulation tasks: *FetchPickAndPlace*, *FetchPush*, *FetchSlide*, *FetchReach*. The tasks are to place defined objects at randomly given target goal positions from random initial states. We use *expert* and *mixed* dataset for each task. The *mixed* dataset contains a mixture of trajectories collected by 90% random policy and 10% expert policy provided from [Yang *et al.*, 2022].

Kitchen

The tasks of kitchen domain are to control a robotic arm interacting with several household items to reach a desired goal configuration. D4RL benchmark [Fu *et al.*, 2020] introduces three types of datasets collected by human demonstrations. The *complete* dataset consists of a relatively small number of trajectories performing all subtasks in order. The *partial* dataset contains a mix of successful trajectories and partially performed trajectories. The partially performed trajectories contain a subset of the desired configuration as well as subtasks irrelevant to the target task. The *mixed* dataset consists of only partially performed trajectories.

5.2 Baselines

We compare Guider against prior approaches in goal-conditioned supervised learning, offline goal-conditioned RL, and offline hierarchical RL.

- **GCSL** [Ghosh *et al.*, 2021] is a goal-conditioned supervised learning method using hindsight goal relabeling.
- **WGCSL** [Yang *et al.*, 2022] is a state-of-the-art offline goal-conditioned RL algorithm based on advantage-weighted supervised learning, using hindsight goal relabeling.
- **CQL+HER** [Kumar *et al.*, 2020] is a modified version of offline RL algorithm CQL which learns the out-of-distribution value function conservatively, with hindsight goal relabeling added.
- **OPAL** [Ajay *et al.*, 2021] is a state-of-the-art offline hierarchical RL method based on skill discovery. The high-level policy directs which skill to use, and the low-level policy executes a sequence of actions for the skill.

Implementation details and hyperparameters for our methods and baselines are provided in supplementary material A.

[1]The results for OPAL on antmaze and kitchen tasks are taken from [Ajay *et al.*, 2021]. Since the official source code of OPAL is not available, the results for fetch tasks are from our re-implementation based on the paper.

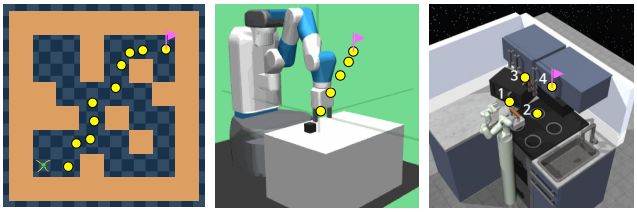


Figure 3: Visualization of sequentially generated subgoals on evaluation for each domain. In antmaze (left), the subgoal suggests a two-dimensional coordinate a quadruped robot should reach. In fetch (middle), the subgoal guides the intermediate positions where the box should be placed. In kitchen (right), the subgoal presents a subtask to be performed.

5.3 Overall Results

The overall results of our experiments are reported in Table 1, and we present the learning curves in Figure 6 in the supplementary material. Our method Guider achieves the best performance on 12 out of 14 tasks. Specifically, the results show that Guider outperforms prior methods by a large margin on challenging offline RL problems such as long-horizon tasks or using mixed datasets.

The antmaze and kitchen are considerably challenging tasks as they require learning policies for reaching long-horizon goals from sparse rewards. Besides, all antmaze datasets and *mixed* dataset of kitchen are composed of task-agnostic datasets, which means that the tasks can not be accomplished by imitating certain trajectories in datasets. The agent must learn to stitch meaningful sub-trajectories from datasets to solve new longer trajectories at testing. Our proposed method effectively addresses these challenges by generating adequate subgoals which are easily reachable. Figure 3 shows the generated subgoals from our high-level policy on evaluations of each task. It is remarkable that Guider shows a success rate of over 80% in *antmaze-large*, where prior methods without the hierarchical approach succeed at less than 20%. We observe that our subgoal-based hierarchical method also outperforms OPAL, the skill-based hierarchical approach. We will discuss the difference between the two approaches through an ablation study in the next subsection.

Another important challenge in offline RL is learning an optimal policy from mixed data including optimal and sub-optimal trajectories. The agent should learn near-optimal policies from the datasets consisting of only a small portion of expert trajectories and the rest of the sub-optimal or random trajectories. In our experiments, the *mixed* datasets of fetch and the *partial* dataset of kitchen tasks deal with the above challenge. Although the fetch-*mixed* dataset contains 10% of expert trajectories and kitchen-partial contains only 3.2% of optimal trajectories, Guider significantly surpasses the average return of the dataset.

5.4 Ablation Studies

Different Low-level Policies

We conduct an ablation study to investigate how our subgoal generation method improves performance on a long-horizon task. We train the unsupervised latent variable model and the high-level policy of our proposed Guider framework on the antmaze-large task, in conjunction with diverse low-level policy learning methods. Figure 4 shows a significant improvement in performance for all types of low-level policy learning when combined with Guider's subgoal generation method. This indicates that Guider takes long-horizon tasks that are difficult to solve with existing methods and turns them into simple tractable problems with guided subgoals.

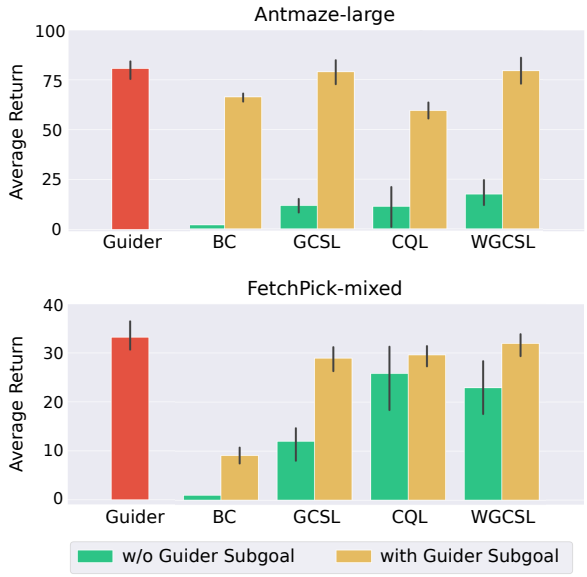


Figure 4: Performance of Guider with various low-level policies compared to individual methods without Guider. All results are averaged over 4 random seeds at the end of training

	Guider	Guider w/o prior	Guider w/o CQL
Antmaze-large	80.8±4.6	57.0±11.3	78.5±11.4
FetchPick-mixed	33.2±2.4	20.2±5.0	28.8±0.9
Kitchen-partial	70.4±7.8	38.9±16.2	59.7±10.2

Table 2: Ablation study on regularization methods. The performance significantly decreases without prior regularization.

Interestingly, we find that Guider combined with behavior cloning (BC) leads to performance similar to OPAL. OPAL learns skill-based hierarchical policies, where the low-level policy extracts a temporally-extended sequence of primitive actions by behavior cloning loss [Ajay *et al.*, 2021]. On the other hand, Guider originally learns low-level goal-reaching policy based on trained value function. This reinforcement learning process at the lower level facilitates additional improvement than imitating actions in the dataset. We conjecture that the different approach in low-level policy learning accounts for the superior performance of Guider to OPAL, although both methods use hierarchical architectures.

Regularization Methods

We also examine the importance of our proposed regularization strategy in subgoal generation and performance. We

compare three different regularization methods while high-level policy training on the antmaze tasks.

- **Guider** is our proposed method implemented on top of CQL with additional regularization toward the prior $\rho_\omega(z|s_t^{\mathcal{G}})$.
- **Guider w/o prior** learns the high-level policy without any additional regularization. It is also implemented on top of CQL.
- **Guider w/o CQL** is implemented on top of Soft Actor-Critic (SAC) [Haarnoja *et al.*, 2017]. SAC is an online actor-critic algorithm without any regularization towards the behavior policy of the offline dataset. We added regularization with the prior $\rho_\omega(z|s_t^{\mathcal{G}})$.

As shown in Table 2, the performance of high-level policy training considerably decreases without prior regularization. We visualize the generated subgoals and the arrived position on evaluation in Figure 7 in the supplementary material. We observe that the agent reaches the generated subgoal in practice only when prior regularization is imposed. Otherwise, the subgoal is generated too far or at an invalid location in relation to the final goal. These infeasible subgoal generations result in decreased success rate.

Subgoal Generation Period

We conduct experiments with varying subgoal generation period c to investigate the influence of this hyperparameter. As shown in Table 3, the subgoal generation period does not critically affect the performance of Guider. However, we understand that too short a subgoal generation period insufficiently benefits from a temporal abstraction of the hierarchical architecture, and too long a period makes it difficult to reach the subgoal with the low-level policy. We chose the appropriate range of the hyperparameter considering the episode length of tasks. In the case of antmaze, it shows the best performance around $c = 50$, which is 1/20 of the episode length of 1000. Additional results on the other tasks are provided in the supplementary material. More sophisticated ways to determine the subgoal generation period can be studied for future work.

Latent Subgoal vs. Reconstructed Subgoal

We investigate the effectiveness of using a decoder to reconstruct a latent subgoal generated by a high-level policy. Our proposed model learns a low-level policy conditioned on the subgoal in goal space, not in latent space. During the evaluation, a generated latent subgoal from the high-level policy is decoded into a goal space before being provided to the low-level policy. However, we do not claim that using a decoder is mandatory for our method. As shown in Figure 5, the difference between Guider with and without a decoder is not significant, although slightly better performance is achieved with the decoder. Our experiments are conducted on environments possessing relatively low-dimensional spatial goal space, which provides clear information. Considering the property of these environments, learning goal-reaching policy in goal space is favorable. On the other hand, one could expect that learning low-level policy conditioned on latent subgoal improves performance on high-dimensional observations such as RGB images, as suggested in prior works [Rafailov *et al.*, 2020; Hafner *et al.*, 2022]. Through these empirical results, we suggest that the general framework of Guider can be extended to diverse high-dimensional observations.

	25	50	75	100
Antmaze-large	77.5	80.8	79.3	73.8
Antmaze-medium	83.3	87.3	85.0	81.5

Table 3: Ablation study on subgoal generation period c. The results are averaged over 4 random seeds.

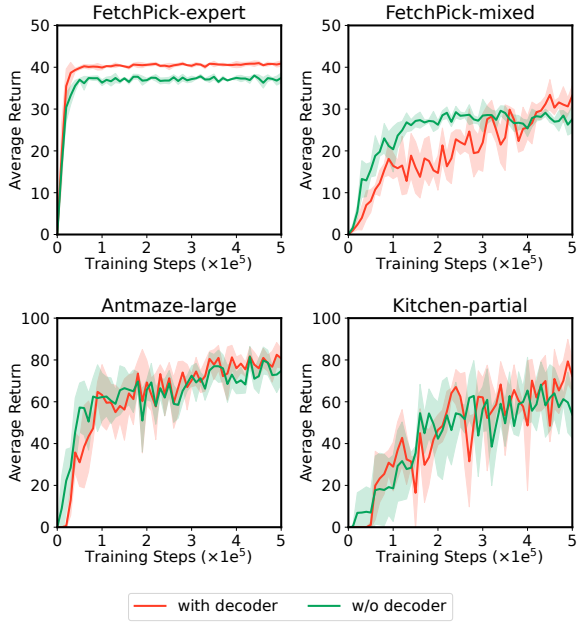


Figure 5: Learning curve of Guider with and without a decoder. Our method suggests decoding a latent subgoal generated from the high-level policy before passing it to the low-level policy. However, the performance does not drastically decrease even without a decoder. For high-dimensional observation and goal space, an approach using latent subgoal-conditioned low-level policy without a decoder can also be considered. All results are averaged over 4 random seeds and the shaded region represents the standard deviation.

6 Conclusion

In this work, we propose Guider, an offline hierarchical reinforcement learning method that learns to generate subgoals at the high level and reach the generated subgoals at the low level. Our unsupervised pre-training of the subgoal prior distribution in latent space can effectively regularize the subgoal generation policy. The generated subgoal can be easily reached by simple low-level policies. Empirical studies show that our proposed method outperforms prior offline RL methods on long-horizon and sparse-reward tasks. An interesting direction of future work would be generating meaningful subgoals from high-dimensional space such as offline RGB images. We also plan to design a flexible subgoal generation model where the generation period can vary depending on the situation.

Acknowledgments

This work was supported partly by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korea Government (MSIT) (No. 2022-0-01045, Self-directed Multi-Modal Intelligence for solving unknown, open domain problems), (No. 2022-0-00688, AI Platform to Fully Adapt and Refect PrivacyPolicy Changes), and (No. 2019-0-00421, Artifcial Intelligence Graduate School Program(Sungkyunkwan University)).

References

[Ajay *et al.*, 2021] Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. In *International Conference on Learning Representations*, 2021.

[Andrychowicz *et al.*, 2017] Marcin Andrychowicz, Dwight Crow, 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*, pages 5048—5058, 2017.

[Bagaria *et al.*, 2021] Akhil Bagaria, Jason Senthil, Matthew Slivinski, and George Konidaris. Robustly learning composable options in deep reinforcement learning. In *International Joint Conference on Artificial Intelligence*, 2021.

[Balaji *et al.*, 2019] Bharathan Balaji, Sunil Mallya, Sahika Genc, Saurabh Gupta, Leo Dirac, Vineet Khare, Gourav Roy, Tao Sun, Yunzhe Tao, Brian Townsend, Eddie Calleja, Sunil Muralidhara, and Dhanasekar Karuppasamy. Deepracer: Educational autonomous racing platform for experimentation with sim2real reinforcement learning. *arXiv preprint arXiv:1911.01562*, 2019.

[Fu *et al.*, 2020] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[Fujimoto *et al.*, 2019] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019.

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

[Haarnoja *et al.*, 2017] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International Conference on Machine Learning*, pages 1352–1361. PMLR, 2017.

[Hafner *et al.*, 2020] Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

[Hafner *et al.*, 2022] Danijar Hafner, Kuang-Huei Lee, Ian Fischer, and Pieter Abbeel. Deep hierarchical planning from pixels. *arXiv preprint arXiv:2206.04114*, 2022.

[Kalashnikov *et al.*, 2018] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, 2018.

[Kostrikov *et al.*, 2021] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[Kumar *et al.*, 2020] Aviral Kumar, Aurick Zhou, George Tucker, and Servey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

[Levy *et al.*, 2019] Andrew Levy, George Konidaris, Robert Platt, and Kate Saenko. Learning multi-level hierarchies with hindsight. In *International Conference on Learning Representations*, 2019.

[Li *et al.*, 2021] Siyuan Li, Lulu Zheng, Jianhao Wang, and Chongjie Zhang. Learning subgoal representations with slow dynamics. In *International Conference on Learning Representations*, 2021.

[Lynch *et al.*, 2020] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonatah Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on Robot Learning*, 2020.

[Nachum *et al.*, 2018] Ofir Nachum, Shixiang Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 3307–3317, 2018.

[Nachum *et al.*, 2019] Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. Algaedice: Policy gradient from arbitrary experience. *arXiv preprint arXiv:1912.02074*, 2019.

[Nair *et al.*, 2020] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

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

[Rafailov *et al.*, 2020] Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. *arXiv preprint arXiv:2012.11547*, 2020.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche1, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game

of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[Wang *et al.*, 2022] Vivienne H. Wang, Joni Pajarinen, Tinghuai Wang, and Joni-Kristian Kamarainen. Hierarchical reinforcement learning with adversarially guided subgoals. *arXiv preprint arXiv:2201.09635*, 2022.

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

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

[Zhang *et al.*, 2020] Jesse Zhang, Haonan Yu, and Wei Xu. Hierarchical reinforcement learning by discovering intrinsic options. In *International Conference on Learning Representations*, 2020.