Offline Goal-Conditioned Reinforcement Learning for Safety-Critical Tasks with Recovery Policy

Chenyang Cao, Zichen Yan, Renhao Lu, Junbo Tan* and Xueqian Wang*

Abstract— Offline goal-conditioned reinforcement learning (GCRL) aims at solving goal-reaching tasks with sparse rewards from an offline dataset. While prior work has demonstrated various approaches for agents to learn near-optimal policies, these methods encounter limitations when dealing with diverse constraints in complex environments, such as safety constraints. Some of these approaches prioritize goal attainment without considering safety, while others excessively focus on safety at the expense of training efficiency. In this paper, we study the problem of constrained offline GCRL and propose a new method called Recovery-based Supervised Learning (RbSL) to accomplish safety-critical tasks with various goals. To evaluate the method performance, we build a benchmark based on the robot-fetching environment with a randomly positioned obstacle and use expert or random policies to generate an offline dataset. We compare RbSL with three offline GCRL algorithms and one offline safe RL algorithm. As a result, our method outperforms the existing state-of-the-art methods to a large extent. Furthermore, we validate the practicality and effectiveness of RbSL by deploying it on a real Panda manipulator. Code is available at https://github.com/Sunlighted/RbSL.git.

I. INTRODUCTION

Reinforcement Learning (RL) enables robots to learn different skills automatically and has achieved great success in many robotic tasks such as navigation, manipulation, locomotion, etc [1]–[5]. Goal-conditioned reinforcement learning is a significant branch of RL, which aims at learning skills to reach distinct goals by adding goal information into the state [6]–[8]. Recently, offline goal-conditioned reinforcement learning has garnered attention among researchers. It utilizes a supervised learning approach to train general-purpose policies from an offline dataset [9]. Thus, robots can avoid incurring real-world training costs without interacting with the environment. However, actions generated by such methods are unconstrained, which potentially results in dangerous behaviors, such as collision with surrounding objects, or even causing damage to the robot itself. For example, a robot attempts to manipulate an object to a specified location while a glass bottle blocks the path. If it ignores the obstacle, it may break the glass and cause a mess. Therefore, it is imperative for the agent to acquire a safe policy that prevents violations of constraints.

Prior works in offline GCRL [9]–[14] focus on reaching the goals along the shortest path by interacting with an offline dataset. Since the agent cannot further explore the environment to get the desired goal, imitation learning is used

to solve the sparse reward and out-of-distribution (OOD) problems [9]. However, it probably ignores security during policy training due to violations of constraints [15]. In our experiments, constraint violations tend to impede task completion, so planning a path to bypass the regions beyond the constraints will lead to better performance. To achieve this goal, we propose enhancing offline goal-conditioned reinforcement learning methods by incorporating mechanisms for constraint management.

By defining a cost function, the RL problem with constraints is called a constrained Markov decision process (CMDP) [16]. Many prior works integrate constraint satisfaction into the optimization problem and set a Lagrange multiplier to formulate an unconstrained problem [17]. In the offline setting, several recent works try to bridge the gap between offline RL and safe RL, such as penalizing unseen actions [18] or constructing an optimization problem with a stationary distribution correction technique [19]. The above-mentioned methods are limited in two aspects. Firstly, the training efficiency is hindered by the conflict between seeking a more optimal policy and adhering to cost constraints. Secondly, there exist estimation errors in the value functions that prevent effective cost reduction below the constraint threshold.

In this paper, we propose Recovery-based Supervised Learning (RbSL) to overcome the above shortcomings. It can be divided into two policies: the goal-conditioned policy and the recovery policy. Utilizing the advantages of imitation learning, we first train an offline policy with hindsight relabeling [7] and OOD action detection [10] to learn goal-reaching tasks. Next, we propose a recovery policy to keep the running trajectory away from the unsafe area [20]. Such structure design helps balance the targets of goal attainment and constraint satisfaction.

In addition, the quality of the dataset also has a significant effect on training results, which is rarely discussed in prior works. Hence, in order to achieve better performance of the recovery policy, we filter out unsafe trajectories from the dataset and keep successful ones to guide safe actions at the boundary of constraints. Besides, we undertake a cost-value relabeling strategy to improve the training efficiency of the recovery policy.

The main contributions are summarized as follows:
- Supervised learning and reinforcement learning are integrated to solve the offline problem. Besides, the OOD action detection technique is proposed to further enhance the policy performance.
- Separate policies are used to ensure safety for the task

All of the authors are with the Center for Intelligent Control and Telescience, Tsinghua Shenzhen International Graduate School, Shenzhen, China. ccy22@mails.tsinghua.edu.cn, tjblql@sz.tsinghua.edu.cn, wang.xq@sz.tsinghua.edu.cn
*Corresponding author: *Junbo Tan* and *Xueqian Wang*

and constraints without a substantial reduction in the success rate.
- The recovery dataset is reshaped to improve the safety of the recovery policy.

II. RELATED WORK

A. Goal-Conditioned Reinforcement Learning

Goal-conditioned reinforcement learning tries to solve problems with sparse rewards and multiple goals, which has been studied in many works [6]–[8], [21], [22]. Hindsight experience relabeling (HER) [7] is a method that relabels the failure trajectories to successful experiences to maximize training efficiency. HPG [6] combines HER with policy gradient to achieve a stable estimation of the goal-conditioned value function. GCSL [8] uses HER and imitation learning to learn an optimal policy.

Many works extend the above-mentioned methods to an offline setting, where agents do not interact with the environment. WGCSL [9] extends GCSL to the offline setting by using weighted supervised learning. GoFAR [11] utilizes value iteration over a dual form of GCRL. Other methods like Actionable Model [10] are based on Q-learning and attempt to address OOD errors by reducing the selection of actions outside the dataset. DQAPG [23] utilizes the method of goal-swapping for data augmentation, which improved offline algorithms from another perspective. To solve long-horizon problem, Li et al. [24] propose a hierarchical trajectory-level diffusion (HDMI) to discovery subgoals and generate actions. Our work is inspired by WGCSL and concentrates on improving the ability to ensure security.

B. Offline Safe Reinforcement Learning

Safe RL algorithms are used to solve constrained optimization problems [16], [17], [25]–[28]. Lagrangian method [17] is the most common way to deal with constraints among these algorithms, which adds a multiplier to penalize the constraint item. However, Lagrangian-based policy faces difficulties in balancing the trade-off between safety and exploration. What's more, in the offline setting, it performs poorly in limiting the cost value especially when the expert data is not sufficient. Several prior works have explored methods for minimizing constraint violations during offline training [18], [19], [29], [30]. They incorporate offline RL ideas [31], [32] into safe RL, such as using the stationary distribution correction estimation technique to optimize the policy with a cost upper bound [19]. CPQ [18] is a value-based method that labels the OOD actions unsafe to reduce the estimation error of cost value.

Another type of safe RL is called Recovery RL [20], which is composed of a task policy and a recovery policy. Its main idea is to use a pre-trained policy to shield the agent from hazardous areas. Guided by a risk Q-value, the recovery policy is designed to adapt unsafe actions to ensure safety [33]. Building upon Recovery RL, our method exploits its potential in an offline setting, a domain that has received limited attention in prior research.

III. PRELIMINARIES

A. Constrained Goal-augmented Markov Decision Process

We propose a Constrained Goal-augmented Markov Decision Process (CGMDP) as a variant of a Goal-augmented Markov Decision Process (GMDP). A CGMDP can be expressed as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{G}, \mathcal{P}, \mathcal{R}, \gamma, \mathcal{C})$. $\mathcal{S}, \mathcal{A}, \mathcal{G}, \mathcal{P}$ denotes state, action, goal space and transition probability, respectively. γ is a discount factor and $r(s, a, g) = 1[\|\phi(s) - g\|_2^2 \leq \delta] : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \to \mathcal{R}$ is a goal-conditioned binary reward related to a threshold δ. ϕ means a state-to-goal mapping. The cost function is $c(s, a) : \mathcal{S} \times \mathcal{A} \to \mathcal{C}$. The cost value V_C^π, cost action-value Q_C^π and cost advantage functions A_C^π are defined by

$$
\begin{aligned}
V_C^\pi(s, g) &:= \mathbb{E}_\pi \Big[\sum_{t=0}^\infty \gamma^t c(s_t, a_t) | s_0 = s, g \Big], \\
Q_C^\pi(s, a, g) &:= \mathbb{E}_\pi \Big[\sum_{t=0}^\infty \gamma^t c(s_t, a_t) | s_0 = s, a_0 = a, g \Big], \\
A_C^\pi(s, a, g) &:= Q_C^\pi(s, a, g) - V_C^\pi(s, g).
\end{aligned}
\tag{1}
$$

In the CGMDP setting, a policy $\pi : \mathcal{S} \times \mathcal{G} \to \mathcal{A}$ aims to maximize the expected return with constraints:

$$
\begin{aligned}
J(\pi) = \max_\pi \ \mathbb{E}_{g \sim p(g), a_t \sim \pi(\cdot|s_t, g)} \Big[\sum_{t=0}^\infty \gamma^t r(s_t, a_t, g) \Big], \\
\text{s.t. } J_C(\pi) \leq l,
\end{aligned}
\tag{2}
$$

where $J_C(\pi) = \mathbb{E}_{a_t \sim \pi(\cdot|s_t, g)} \big[\sum_{t=0}^\infty \gamma^t c(s_t, a_t) \big]$ denotes the expectation of discounted cost return and l is the constraint bound. This constrained optimization problem can be modified into a Lagrangian form:

$$
\min_{\lambda \geq 0} \max_\pi \ \mathbb{E} \Big[\sum_{t=0}^\infty \gamma^t (r(s_t, a_t, g) - \lambda c(s_t, a_t)) + \lambda l \Big], \tag{3}
$$

where λ is a Lagrange multiplier.

B. Offline Constrained GCRL

In an offline setting, the training trajectory is sampled from a static dataset of logged transitions $\mathcal{D} := \{\tau_i\}_{i=1}^N$ in a form of $\tau_i = (s_0^{(i)}, a_0^{(i)}, r_0^{(i)}, c_0^{(i)} \ldots ; g^{(i)})$. The optimization problem of constrained GCRL can be described as:

$$
\begin{aligned}
J(\pi) = \max_\pi \ \mathbb{E}_{(s_t, s_{t+1}, g) \sim \mathcal{D}, a_t \sim \pi} \Big[\sum_{t=0}^\infty \gamma^t r(s_t, a_t, g) \Big], \\
\text{s.t. } \mathbb{E}_{(s_t, s_{t+1}) \sim \mathcal{D}, a_t \sim \pi} \Big[\sum_{t=0}^\infty \gamma^t c(s_t, a_t) \Big] \leq l.
\end{aligned}
\tag{4}
$$

IV. METHOD

The proposed method simplifies the problem (4) into two parts. The safe goal-reaching problem is addressed through a supervised learning-based method, while constraints are managed by a recovery policy. The combination of these two policies significantly improves training efficiency.
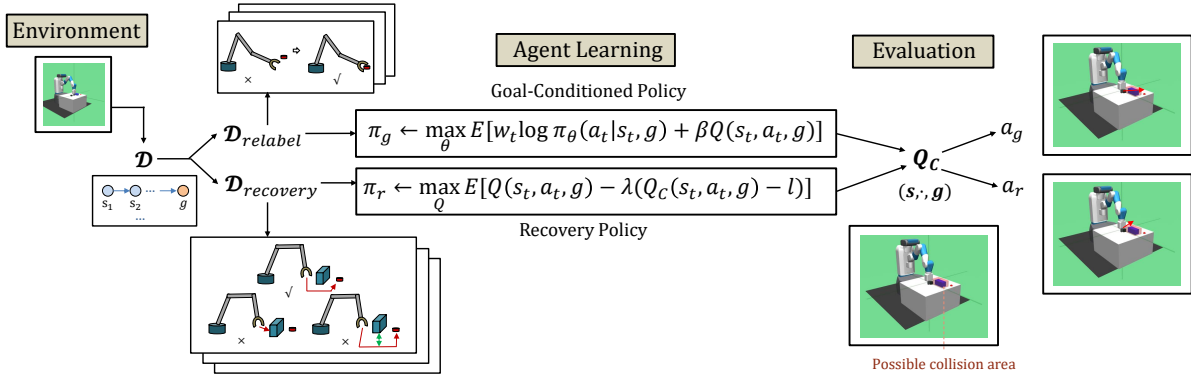
Fig. 1: An overview of Recovery-based Supervised Learning (RbSL). RbSL first samples data from the environment and processes them into two datasets. Then, the agent learns a goal-conditioned policy and a recovery policy. In evaluation, we use the cost Q-value to predict an unsafe state and decide which policy to use.

A. Overview

In order to reach goals safely, we train a cost Q-network to predict the likelihood of violations, which determines the switching between two policies:

$$\pi(a_t|s_t, g) = \begin{cases} \pi_g(a_t|s_t, g), & Q_C(s_t, a_t, g) \le l, \\ \pi_r(a_t|s_t, g), & Q_C(s_t, a_t, g) > l, \end{cases} \quad (5)$$

where π_g indicates the goal-conditioned policy and π_r denotes the recovery policy. The goal-conditioned policy is primarily oriented toward reaching the goal. When it breaches the constraint, the recovery policy will try to modify the action in the direction of decreasing the cost Q-value. Our proposed method is illustrated in Figure 1. Next, in sections B and C, we will introduce more details of the two policies and discuss why they behave well in the offline setting.

B. Supervised Learning

In offline GCRL, imitation learning methods like GCSL [8] mimic actions to ensure that the learned policy is consistent with the data distribution. However, it may perform sub-optimal behavior in the offline setting as it only focuses on the success of each task at the end of the episode while neglecting optimization at each individual step. To overcome this disadvantage, prior works have tried to estimate the optimal policy by advantage-weighted regression [34], [35]. Weighted goal-conditioned supervised learning (WGCSL) can determine the shortest path in offline goal-reaching tasks. Therefore, we use it as our goal-conditioned policy.

First, we relabel the offline dataset $\mathcal{D}$ for supervised learning. Suppose that $\mathcal{D} = \{s_t, a_t, r_t, g\}, t \in [1, T]$, we relabel the goal with a future state $g' = \phi(s_{t'}), t' \ge t$, where ϕ is a mapping from state to goal. In an environment with constraints, we add the information of constraints as a part of the state into the dataset:

$$\mathcal{D}_{\text{relabel}} = \begin{cases} \{s_t, a_t, r_{t'}, g' = \phi(s_{t'}), o\}, & s_T \notin o, \\ \{s_t, a_t, r_t, g, o\}, & \text{otherwise}, \end{cases} \quad (6)$$

where o indicates an unsafe area. The relabeling reduces the experience that violates the constraints and increases successful trajectories to improve learning efficiency.

Next, we use a composed weight to guarantee an optimal policy in the offline setting. The policy can be trained by maximizing the following BC-based expectations [36]:

$$J_{\text{wgcsl}}(\pi) = \mathbb{E}_{(s_t, a_t, g') \sim \mathcal{D}_{\text{relabel}}} \big[w_{t,t'} \cdot \log \pi_\theta(a_t|s_t, g') \big], \quad (7)$$

where the weight $w_{t,t'} = \gamma^{t'-t} \cdot \exp(A(s_t, a_t, g')) \cdot \epsilon(A(s_t, a_t, g'))$. The first part $\gamma^{t'-t}$ assigns smaller weight to sub-optimal trajectories. The second part $\exp(A(s_t, a_t, g'))$ restricts the learned policy to avoid OOD actions. The last part equals one if $A(s_t, a_t, \phi(s_i)) > \hat{A}$ and equals to ϵ otherwise, where $\hat{A}$ is a threshold and ϵ is a small value. It leads the policy to the highest return.

To get better performance than imitation learning on goal-reaching tasks, we consider optimizing the Q-value to improve the behavior policy. Therefore, we rewrite the policy improvement (7) following TD3+BC [37] approach:

$$\begin{aligned} J(\pi) = \mathbb{E}_{(s_t, a_t, g') \sim \mathcal{D}_{\text{relabel}}} \big[& w_{t,t'} \cdot \log \pi_\theta(a_t|s_t, g') \\ & + \alpha' Q_g(s_t, a_t, g') \big], \\ \text{and} \quad \alpha' = \frac{\alpha}{\frac{1}{N} \sum_{(s_i, a_i)} |Q_g(s_i, a_i, g')|}, \end{aligned} \quad (8)$$

where α is a hyper-parameter to balance maximizing Q-value and minimizing the BC term. Since α is highly influenced by the scale of Q-value, we add a normalization term $\frac{1}{N} \sum_{(s_i, a_i)} |Q_g(s_i, a_i, g')|$ into it.

C. Recovery Policy

The recovery policy is responsible for action correction as shown in Figure 2. We first learn a constraint Q-value $Q_C(s_t, a_t, g)$ to estimate the future probability of violation:

$$\begin{aligned} Q_C(s_t, a_t, g) &= \mathbb{E}\big[\sum_{t'=t}^{\infty} \gamma^{t'-t} c_{t'} | s_t, a_t \big] \\ &= c_t + (1 - c_t)\gamma \mathbb{E}\big[Q_C(s_{t+1}, a_{t+1}, g) | s_t, a_t \big], \end{aligned} \quad (9)$$

where c_t indicates the cost function. Recovery RL [20] pretrains the cost Q-value on the historical trajectories that contain constraint violations. In contrast, our method trains the cost Q-value and the goal-conditioned Q-value simultaneously as the trajectories contain information about a complete
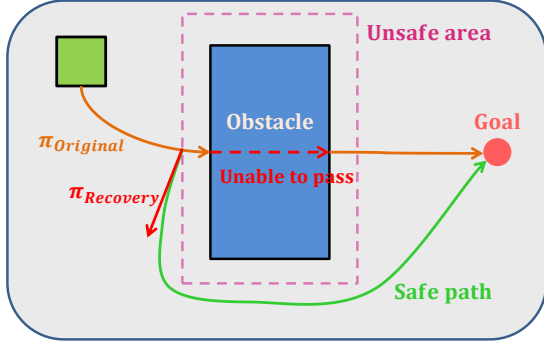
Fig. 2: Recovery policy: we illustrate the recovery policy on a robotic push task. The original policy will choose the shortest path to reach the goal while the obstacle blocks its way and results in failure. In contrast, the recovery policy will correct the action of entering into an unsafe area and plan a safe path to the goal.

path. It is necessary for the agent to learn to reach the goal so that the recovery policy can be effectively updated. To this end, we consider policy improvement as follows:

$$\pi_r = \arg\min_{\pi} \mathbb{E}_{(s,g)\sim\mathcal{D}, a\sim\pi(\cdot|s,g)}\big[Q_r(s,a,g) \\ - \lambda(Q_C(s,a,g)-l)\big], \quad (10)$$

where λ represents a Lagrange multiplier in an optimization problem, chosen to be sufficiently large to constrain the cost Q-value below l.

In the offline setting, there exists a problem of overestimating Q-values caused by OOD actions for the recovery policy. Hence, we use $\mathcal{A}'(s,g)$ to denote an unseen action set that can not be reached when the agent tries to achieve the goal g from the state s in the dataset $\mathcal{D}$. Next, let $\mathcal{G}'(g)$ be a set of negative goals different from the desired goal g. We can assume that any action $a' \sim \mathcal{A}'(s,g)$ finally leads to a negative goal $g' \sim \mathcal{G}'(g)$. Therefore, we can minimize the Q-value for these negative actions a' using relabeling. We first sample a negative action a' from the soft-max distribution $\frac{exp(Q_r(s,a',g))}{Z}$ where Z is a normalization factor. Then we set the target Q-value to zero as $Q_{target}(s,a',g) \leftarrow 0$ so that we alleviate the high value of OOD action and get the correct recovery policy.

D. Data Processing

In the offline setting, we should collect an offline dataset $\mathcal{D}$ containing data from random policies and experts first. The task policy and recovery policy are both trained on $\mathcal{D}$. However, the recovery policy is so sensitive to random data that safe actions cannot be learned effectively. Additionally, some trajectories with zero cost return are useless for training recovery policy. Therefore, we filter the dataset $\mathcal{D}$ in two steps: 1) filter out random trajectories τ_i if $R(\tau_i) = \sum_{t=0}^T \gamma^t r_i(s_t, a_t, g) = 0$ to get an expert dataset $\mathcal{D}_e$; 2) filter out some expert trajectories τ_i' if $C(\tau_i') = \sum_{t=0}^T \gamma^t c_i(s_t, a_t) = 0$ from $\mathcal{D}_e$. Finally, we get a recovery dataset $\mathcal{D}_{rec}$.

Algorithm 1 Recovery-based Supervised Learning

Require: $\mathcal{D}_{\text{offline}}$, number of episodes N, number of episodes N_r for recovery training, a Lagrangian multiplier λ

1: Randomly initialize policy π_g, π_r, and value function Q_g, Q_r, Q_C.
2: Filter the recovery dataset and get $\mathcal{D}_{\text{goal}} \leftarrow \mathcal{D}_{\text{offline}}$, $\mathcal{D}_{\text{recovery}} \leftarrow \mathcal{D}_{\text{filter}}$
3: **for** $i \in \{1, \ldots N\}$ **do**
4:      Sample a minibatch $\{(s_t^i, a_t^i, g^i, s_{t+1}^i, r_t^i)\}_{i=1}^M$ from the offline dataset $\mathcal{D}_{\text{goal}}$
5:      Relabel the minibatch by using Eq.6: $\{(s_t^i, a_t^i, \phi(s_j^i), s_{t+1}^i, r_t^i(\cdot, \cdot, \phi(s_j^i)))\}_{i=1}^M, j \geq t$
6:      Update Q_g and estimate $A(s_t^i, a_t^i, \phi(s_j^i))$
7:      Get K percentile $\hat{A}$ and update policy π_g by Eq.8
8: **end for**
9: **for** $i \in \{1, \ldots N_r\}$ **do**
10:      Sample a minibatch $\{(s_t^i, a_t^i, g^i, s_{t+1}^i, r_t^i, c_t^i)\}_{i=1}^M$ from the offline dataset $\mathcal{D}_{\text{recovery}}$
11:      Relabel dataset using Eq.6 and do cost shaping Eq.11
12:      Update Q_r by TD-error with negative action penalty
13:      Update Q_C by Eq.9
14:      Update recovery policy π_r by Eq.10
15: **end for**

To train a recovery policy, we consider cost shaping to $\mathcal{D}_{\text{rec}}$ as follows:

$$c'(s_t, a_t) = \begin{cases} c(s_t, a_t) - 1, & c(s_t, a_t) = 1 \text{ and} \\ & c(s_{t+1}, a_{t+1}) = 0, \\ c(s_t, a_t), & \text{otherwise.} \end{cases} \quad (11)$$

We reduce the cost by learning actions that prevent the policy from violating constraints. The cost shaping facilitates the recovery policy in learning such safe actions.

To sum up, two policies are trained on different datasets independently. During the evaluation phase, we use both policies for decision-making to ensure both the task performance and security. We illustrate our complete method in Algorithm 1.

V. EXPERIMENTS

Our experiments are carried out in Gym-Robotics environments [38] with four classic manipulation tasks. Besides, Panda-Gym [39] is used for applying RbSL to the sim-to-real experiments, as shown in Figure 3. In this section, we aim to answer the following questions:

1) Can RbSL consistently outperform prior methods across various tasks on datasets of different quality?

2) Can RbSL reach goals with fewer constraint violations compared to other offline goal-conditioned algorithms?

3) Can RbSL successfully transition from a simulated environment to real-world applications?

Moreover, in order to verify our method's robustness, we change the mixing ratio of expert and random data in the offline dataset to test the algorithm's performance. In the

TABLE I: Main Result (averaged over 5 seeds)

Task	Mixture of dataset (expert–random)	AM-lag		GCSL		WGCSL		GoFAR		RbSL(Ours)	
		Success rate	Cost return	Success rate	Cost return	Success rate	Cost return	Success rate	Cost return	Success rate	Cost return
ReachObstacle	0.1-0.9	**1**	**$0.016_{\pm0.000}$**	0.997	$3.15_{\pm0.036}$	**1**	$1.646_{\pm0.004}$	**1**	$1.886_{\pm0.000}$	**1**	$0.123_{\pm0.000}$
	0-1	0.983	$0.217_{\pm0.071}$	0.773	$17.65_{\pm1.144}$	0.99	$6.326_{\pm1.971}$	0.873	$7.996_{\pm0.684}$	**1**	**$0.21_{\pm0.000}$**
PickAndPlaceObstacle	1-0	0.58	$1.223_{\pm0.521}$	0.719	$6.336_{\pm0.859}$	0.916	$4.003_{\pm1.21}$	0.916	$2.966_{\pm0.089}$	**0.946**	**$0.87_{\pm0.276}$**
	0.5-0.5	0.447	$1.902_{\pm0.391}$	0.68	$7.65_{\pm1.266}$	0.86	$4.253_{\pm0.841}$	0.91	$4.53_{\pm1.353}$	**0.936**	**$1.41_{\pm0.083}$**
	0.2-0.8	0.337	$2.654_{\pm1.833}$	0.623	$7.816_{\pm0.592}$	0.81	$5.28_{\pm0.053}$	0.836	$4.47_{\pm0.405}$	**0.87**	**$2.14_{\pm0.018}$**
	0.1-0.9	0.206	$2.947_{\pm4.172}$	0.553	$9.236_{\pm3.295}$	0.733	$9.11_{\pm1.411}$	0.793	$6.336_{\pm0.341}$	**0.843**	**$2.516_{\pm0.75}$**
PushObstacle	1-0	0.639	$2.473_{\pm0.363}$	0.71	$4.353_{\pm0.285}$	0.813	$3.74_{\pm1.162}$	0.723	$4.026_{\pm1.843}$	**0.896**	**$0.566_{\pm0.009}$**
	0.5-0.5	0.57	$3.2_{\pm1.786}$	0.667	$5.233_{\pm2.437}$	0.756	$4.633_{\pm0.991}$	0.553	$3.96_{\pm0.204}$	**0.883**	**$0.553_{\pm0.015}$**
	0.2-0.8	0.413	$4.766_{\pm1.103}$	0.523	$6.066_{\pm1.703}$	0.646	$4.71_{\pm1.251}$	0.41	$5.23_{\pm0.504}$	**0.667**	**$2.113_{\pm0.464}$**
	0.1-0.9	0.083	**$1.65_{\pm2.63}$**	0.393	$5.4_{\pm0.557}$	0.526	$4.31_{\pm0.275}$	0.343	$5.64_{\pm2.39}$	**0.603**	$2.43_{\pm0.569}$
SlideObstacle	1-0	0.336	$2.483_{\pm0.135}$	0.313	$3.436_{\pm0.7}$	0.33	$3.02_{\pm0.326}$	0.523	$3.239_{\pm0.128}$	**0.556**	**$0.356_{\pm0.012}$**
	0.5-0.5	0.153	$2.81_{\pm0.948}$	0.28	$4.749_{\pm1.11}$	0.313	$3.25_{\pm0.268}$	**0.516**	$3.023_{\pm0.042}$	0.459	**$0.77_{\pm0.141}$**
	0.2-0.8	0.05	$1.5_{\pm0.551}$	0.216	$3.953_{\pm1.233}$	0.24	$2.846_{\pm0.264}$	**0.353**	$2.153_{\pm0.239}$	0.286	**$1.21_{\pm0.058}$**
	0.1-0.9	0.02	**$0.85_{\pm1.156}$**	0.176	$4.373_{\pm2.040}$	0.15	$2.803_{\pm0.375}$	**0.293**	$2.656_{\pm0.285}$	0.243	$1.75_{\pm0.299}$
Average	-	0.416	2.049	0.545	6.386	0.649	4.281	0.646	4.151	**0.726**	**1.216**

rest of this section, the practical significance of the proposed method is verified.

A. Environment and Experiment Setup

We modify the four manipulation tasks in MoJoCo [40] Gym-Robotics by adding obstacles: *ReachObstacle, PickAndPlaceObstacle, PushObstacle, SlideObstacle*. These tasks require the robot or object to reach the goal as quickly as possible without collision. The best case is to find the shortest path away from the obstacle to attain the goal.

The state space for *ReachObstacle* is $\mathcal{S} \in \mathbb{R}^{19}$ consists of a 10-dim vector about basic environment information and another 9-dim vector representing obstacle position and rotation. The rest three tasks have a (25+12)-dim state by adding the object information and relative positions. The action space $\mathcal{A} \in \mathbb{R}^4$ contains the increments of the end-effector in the Cartesian space and control of grippers.

The cost function is defined as

$$c(s_t) = 1(s_t \notin O), \qquad (12)$$

where $O = \{\mathbf{o} = (x, y, z)|d(\mathbf{o}_i, \mathbf{s}_i) - L_i \leq \epsilon, i = x, y, z\}$ is a box region and $\mathbf{s}$ is the object (or grasp) coordinate. L_i is the sum of object and obstacle sizes in orientation i. ϵ is a threshold (0.05 in experiments).

In all tasks, the rewards are sparse and binary: the agent receives +1 if it achieves a desired goal and 0 otherwise. We collect data with two different policies for offline training, like the way in [9]. The random dataset is collected by a uniform random policy. And the expert chooses an online trained data-augment TD3+HER policy [21], [41] with Gaussian noise. Each dataset has 2×10^6 transitions, enough for offline algorithms to learn a qualified policy. In our experiments, we mix the random data and expert data in different proportions to test the policy's robustness. The main results are presented in Table I and Figure 4.

For the real-world experiment, we employ a Franka Emika Panda robot for manipulation and an Intel RealSense camera which is set beyond the robotic arm to capture object coordinates. Its setup is visualized in Figure 3(c).
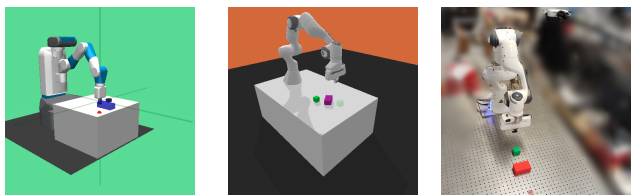
(a) Gym-Robotics (b) Panda-Gym (c) Franka Emika Panda

Fig. 3: Goal-conditioned environments: (a) MuJoCo Gym-Robotics, (b) Panda-Gym, a simulation environment for the real-world experiment, (c) An experiment environment for the Franka Emika Panda robotic arm.

B. Baselines

We compare our method with prior offline GCRL and offline safe RL algorithms. GCSL [8] is an imitation learning method which shows good performance in goal-reaching tasks. WGCSL [9] solves the offline GCRL problem by improving GCSL in the offline setting using AWR [34]. GoFAR [11] is a state-of-the-art offline GCRL method that is practical in various challenging problem settings. Since GCRL methods can learn policies to reach goals, they may cause collisions with obstacles. Therefore, we use AM-lag [10], an offline safe GCRL, to compare the safety with our algorithm. It is built on a conservative Q-Learning [42] method using goal-chaining and simply adds a Lagrangian multiplier to penalize constraint violations.

C. Evaluations and Results

We evaluate our method in four tasks and record the success rate and cost return in Table I. The constraint limit is set as 1.5 in all tasks, which means the average constraint violation rate is 1.5 every episode. The Lagrangian approach uses adaptive PID-based Lagrangian multipliers [43] and the initial value of the lagrangian multiplier is set as 10 which should be set as large as possible. As we can see, it becomes evident that RbSL consistently performs well across all settings, achieving the highest average success rate and the lowest average cost return. In other words,
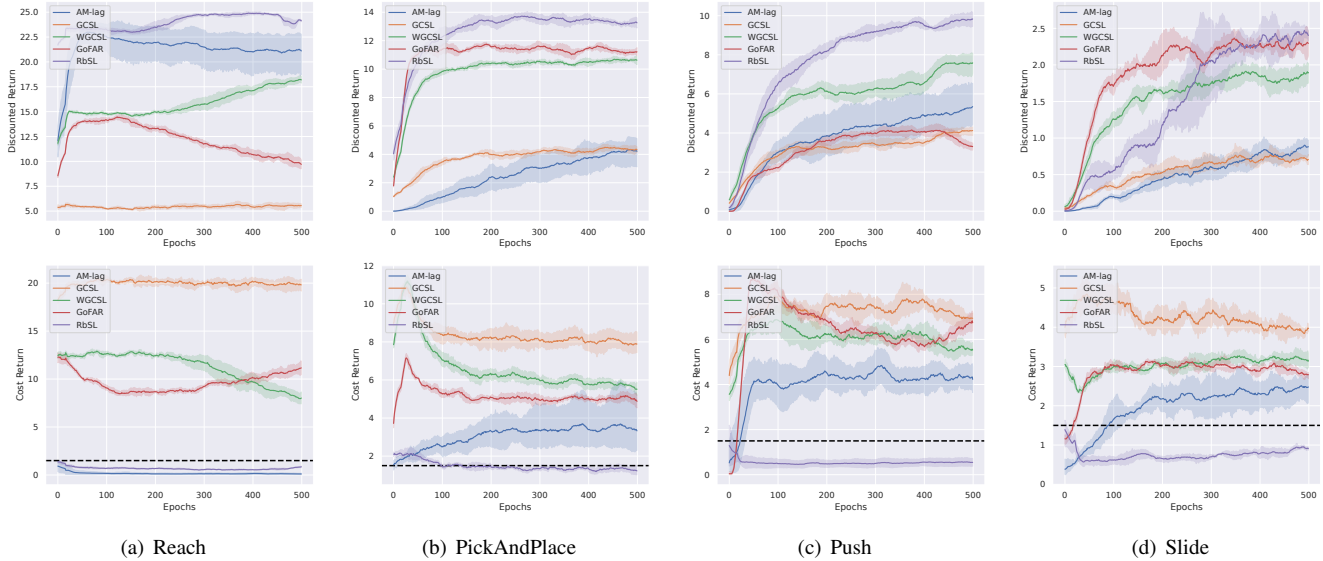
(a) Reach   (b) PickAndPlace   (c) Push   (d) Slide

Fig. 4: Training curves of *PushObstacle, PickAndPlaceObstacle, SlideObstacle* in the 0.5-0.5 setting and *ReachObstacle* in the 0-1 setting. In each figure, the sub-figures show the average number of discounted returns and cost returns per epoch, where the black dotted line shows the safe constraint limit.

RbSL can reach goals precisely without encountering constraint violations. However, we observed that our method underperforms compared to GoFAR in *SlideObstacle*, primarily because the recovery policy faces challenges when handling complex tasks that are less safety-critical. The previous GCRL methods like GoFAR often lead to more collisions with obstacles, even though they exhibit good task performance. Conversely, offline safe RL methods encounter difficulties in increasing the success rate when attempting to violate the constraints. Nevertheless, RbSL consistently demonstrates robust and outstanding performance across datasets of various quality. To better visualize the effects of the algorithm, we show training curves in Figure 4. As evident, there is a significant gap between RbSL and the baselines, especially the cost return in challenging tasks like *PushObstacle, SlideObstacle*. Besides, RbSL exhibits faster convergence compared to the others. Overall, RbSL achieves superior task performance by effectively avoiding obstacles, thereby reducing the likelihood of constraint violations.

D. Real World Deployment

To apply our algorithm to a real robot, we built a simulation environment in panda-gym [39]. The agent receives joint positions and velocities of the end-effector as a robotic state and it is controlled by a 3-dim vector describing the end-effector's position in simulation. We compare RbSL with WGCSL for 100 epochs of training. The simulation result is shown in Table II.

Finally, we apply the offline trained model on the real manipulator and use joint position control without sim-to-real fine-tuning. We use HSV color space filtering to obtain the positions of objects and obstacles in the real world. The task is pushing the green object to the goal without colliding
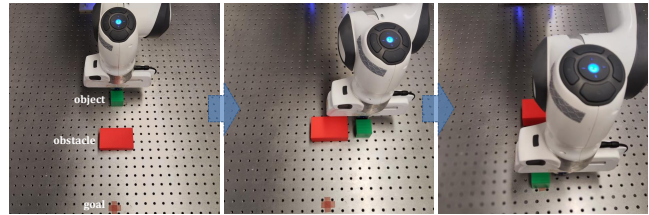

Fig. 5: Leveraging RbSL, we display the example execution trajectory of push, which involves navigating around an obstacle to reach the goal.

TABLE II: Simulation Result

Algorithm	Success Rate	Discounted Return	Cost Return
WGCSL [9]	78.3%	11.862	1.493
RbSL(Ours)	**96.7%**	**17.056**	**0.978**

with the red obstacle. The results are shown in Figure 5 and more details can be found in the supplementary video.

VI. CONCLUSIONS

This paper proposes a novel offline GCRL method to solve constraints in complex environments by combining a goal-conditioned policy and a recovery policy. In our experiment, it is proved that the proposed RbSL can accomplish goal-reaching tasks with lower cost return and it outperforms the prior methods. Additionally, RbSL is tested in a real robot and achieves accomplished returns.

ACKNOWLEDGMENT

REFERENCES

[1] L. Smith, J. C. Kew, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Legged robots that keep on learning: Fine-tuning locomotion policies in the real world," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 1593–1599, IEEE, 2022.

[2] M. Wilson and T. Hermans, "Learning to manipulate object collections using grounded state representations," in *Conference on Robot Learning*, pp. 490–502, PMLR, 2020.

[3] Z. Fu, X. Cheng, and D. Pathak, "Deep whole-body control: learning a unified policy for manipulation and locomotion," in *Conference on Robot Learning*, pp. 138–149, PMLR, 2023.

[4] S. W. Abeyruwan, L. Graesser, D. B. D'Ambrosio, A. Singh, A. Shankar, A. Bewley, D. Jain, K. M. Choromanski, and P. R. Sanketi, "i-sim2real: Reinforcement learning of robotic policies in tight human-robot interaction loops," in *Conference on Robot Learning*, pp. 212–224, PMLR, 2023.

[5] S. Kumar, J. Zamora, N. Hansen, R. Jangir, and X. Wang, "Graph inverse reinforcement learning from diverse videos," in *Conference on Robot Learning*, pp. 55–66, PMLR, 2023.

[6] P. Rauber, A. Ummadisingu, F. Mutz, and J. Schmidhuber, "Hindsight policy gradients," in *International Conference on Learning Representations*, 2018.

[7] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, vol. 30, 2017.

[8] D. Ghosh, A. Gupta, A. Reddy, J. Fu, C. M. Devin, B. Eysenbach, and S. Levine, "Learning to reach goals via iterated supervised learning," in *International Conference on Learning Representations*, 2020.

[9] R. Yang, Y. Lu, W. Li, H. Sun, M. Fang, Y. Du, X. Li, L. Han, and C. Zhang, "Rethinking goal-conditioned supervised learning and its connection to offline rl," in *International Conference on Learning Representations*, 2021.

[10] Y. Chebotar, K. Hausman, Y. Lu, T. Xiao, D. Kalashnikov, J. Varley, A. Irpan, B. Eysenbach, R. C. Julian, C. Finn, *et al.*, "Actionable models: Unsupervised offline reinforcement learning of robotic skills," in *International Conference on Machine Learning*, pp. 1518–1528, PMLR, 2021.

[11] J. Y. Ma, J. Yan, D. Jayaraman, and O. Bastani, "Offline goal-conditioned reinforcement learning via f-advantage regression," *Advances in Neural Information Processing Systems*, vol. 35, pp. 310–323, 2022.

[12] J. Hejna, J. Gao, and D. Sadigh, "Distance weighted supervised learning for offline interaction data," *arXiv preprint arXiv:2304.13774*, 2023.

[13] K. Fang, P. Yin, A. Nair, H. R. Walke, G. Yan, and S. Levine, "Generalization with lossy affordances: Leveraging broad offline data for learning visuomotor tasks," in *Conference on Robot Learning*, pp. 106–117, PMLR, 2023.

[14] L. Mezghani, S. Sukhbaatar, P. Bojanowski, A. Lazaric, and K. Alahari, "Learning goal-conditioned policies offline with self-supervised reward shaping," in *Conference on Robot Learning*, pp. 1401–1410, PMLR, 2023.

[15] L. Brunke, M. Greeff, A. W. Hall, Z. Yuan, S. Zhou, J. Panerati, and A. P. Schoellig, "Safe learning in robotics: From learning-based control to safe reinforcement learning," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, pp. 411–444, 2022.

[16] E. Altman, *Constrained Markov Decision Processes*. PhD thesis, INRIA, 1995.

[17] Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone, "Risk-constrained reinforcement learning with percentile risk criteria," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6070–6120, 2017.

[18] H. Xu, X. Zhan, and X. Zhu, "Constraints penalized q-learning for safe offline reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 8753–8760, 2022.

[19] J. Lee, C. Paduraru, D. J. Mankowitz, N. Heess, D. Precup, K.-E. Kim, and A. Guez, "Coptidice: Offline constrained reinforcement learning via stationary distribution correction estimation," in *International Conference on Learning Representations*, 2021.

[20] B. Thananjeyan, A. Balakrishna, S. Nair, M. Luo, K. Srinivasan, M. Hwang, J. E. Gonzalez, J. Ibarz, C. Finn, and K. Goldberg, "Recovery rl: Safe reinforcement learning with learned recovery zones," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4915–4922, 2021.

[21] Y. Lin, J. Huang, M. Zimmer, Y. Guan, J. Rojas, and P. Weng, "Invariant transform experience replay: Data augmentation for deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6615–6622, 2020.

[22] N. Dengler, D. Großklaus, and M. Bennewitz, "Learning goal-oriented non-prehensile pushing in cluttered scenes," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1116–1122, IEEE, 2022.

[23] W. Yang, H. Wang, D. Cai, J. Pajarinen, and J.-K. Kämäräinen, "Swapped goal-conditioned offline reinforcement learning," *arXiv preprint arXiv:2302.08865*, 2023.

[24] W. Li, X. Wang, B. Jin, and H. Zha, "Hierarchical diffusion for offline decision making," in *International Conference on Machine Learning*, pp. 20035–20064, PMLR, 2023.

[25] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International conference on machine learning*, pp. 22–31, PMLR, 2017.

[26] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," in *International Conference on Learning Representations*, 2018.

[27] L. Zhang, Z. Yan, L. Shen, S. Li, X. Wang, and D. Tao, "Safety correction from baseline: Towards the risk-aware policy in robotics via dual-agent reinforcement learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 9027–9033, IEEE, 2022.

[28] Q. Lin, B. Tang, Z. Wu, C. Yu, S. Mao, Q. Xie, X. Wang, and D. Wang, "Safe offline reinforcement learning with real-time budget constraints," *arXiv preprint arXiv:2306.00603*, 2023.

[29] H. Le, C. Voloshin, and Y. Yue, "Batch policy learning under constraints," in *International Conference on Machine Learning*, pp. 3703–3712, PMLR, 2019.

[30] Z. Liu, Z. Guo, Y. Yao, Z. Cen, W. Yu, T. Zhang, and D. Zhao, "Constrained decision transformer for offline safe reinforcement learning," *arXiv preprint arXiv:2302.07351*, 2023.

[31] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[32] R. F. Prudencio, M. R. Maximo, and E. L. Colombini, "A survey on offline reinforcement learning: Taxonomy, review, and open problems," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[33] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[34] X. B. Peng, A. Kumar, G. Zhang, and S. Levine, "Advantage-weighted regression: Simple and scalable off-policy reinforcement learning," *arXiv preprint arXiv:1910.00177*, 2019.

[35] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," in *International Conference on Learning Representations*, 2021.

[36] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," *arXiv preprint arXiv:1805.01954*, 2018.

[37] S. Fujimoto and S. S. Gu, "A minimalist approach to offline reinforcement learning," *Advances in neural information processing systems*, vol. 34, pp. 20132–20145, 2021.

[38] M. Plappert, M. Andrychowicz, A. Ray, B. McGrew, B. Baker, G. Powell, J. Schneider, J. Tobin, M. Chociej, P. Welinder, *et al.*, "Multi-goal reinforcement learning: Challenging robotics environments and request for research," *arXiv preprint arXiv:1802.09464*, 2018.

[39] Q. Gallouédec, N. Cazin, E. Dellandréa, and L. Chen, "panda-gym: Open-source goal-conditioned environments for robotic learning," *arXiv preprint arXiv:2106.13687*, 2021.

[40] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033, IEEE, 2012.

[41] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*, pp. 1587–1596, PMLR, 2018.

[42] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1179–1191, 2020.

[43] A. Stooke, J. Achiam, and P. Abbeel, "Responsive safety in reinforcement learning by pid lagrangian methods," in *International Conference on Machine Learning*, pp. 9133–9143, PMLR, 2020.