Safety Optimized Reinforcement Learning via Multi-Objective Policy Optimization

Homayoun Honari, Mehran Ghafarian Tamizi, Homayoun Najjaran[1]

***Abstract*— Safe reinforcement learning (Safe RL) refers to a class of techniques that aim to prevent RL algorithms from violating constraints in the process of decision-making and exploration during trial and error. In this paper, a novel model-free Safe RL algorithm, formulated based on the multi-objective policy optimization framework is introduced where the policy is optimized towards optimality and safety, simultaneously. The optimality is achieved by the environment reward function that is subsequently shaped using a safety critic. The advantage of the *Safety Optimized RL (SORL)* algorithm compared to the traditional Safe RL algorithms is that it omits the need to constrain the policy search space. This allows SORL to find a natural tradeoff between safety and optimality without compromising the performance in terms of either safety or optimality due to strict search space constraints. Through our theoretical analysis of SORL, we propose a condition for SORL's converged policy to guarantee safety and then use it to introduce an aggressiveness parameter that allows for fine-tuning the mentioned tradeoff. The experimental results obtained in seven different robotic environments indicate a considerable reduction in the number of safety violations along with higher, or competitive, policy returns, in comparison to six different state-of-the-art Safe RL methods. The results demonstrate the significant superiority of the proposed SORL algorithm in safety-critical applications.**

I. Introduction

Reinforcement learning (RL) is a class of machine learning methods where an agent learns to make decisions by interacting with an environment to maximize rewards. However, the trial-and-error nature of training RL algorithms makes them challenging to use in safety-critical applications where the execution of some actions might lead to system failure. To tackle this, Safe RL algorithms aim to incorporate safety into the learning process to ensure that the policy learned by the algorithm avoids dangerous states. These algorithms have been applied successfully in various real-world domains such as robotics [1], [2] and autonomous driving [3] showing their great potential to enable the control of real-world systems with a minimized total number of failures.

As reviewed extensively in [4]–[6], most of the Safe RL methods utilize the Constrained Markov Decision Process (CMDP) framework. Algorithms under this framework specify a level of safety that the policy must adhere to while exploring unknown states and improving its reward performance. However, a major disadvantage of this class of algorithms is their susceptibility to converge to a suboptimal policy due to suboptimal tuning of the safety-related hyperparameters.

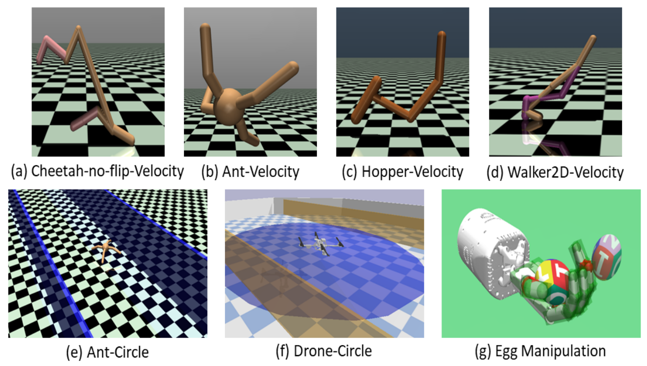

Fig. 1: Visualization of the simulated safety-concerned robotics environments used to evaluate SORL in the context of three main safety topics of system-level safety (a-d), collision avoidance (e,f) and safe manipulation (g). The top row also showcases some possible constraint violation for the system-level safety benchmarks.

To mitigate this, our work presents a novel model-free Safe RL algorithm, named Safety Optimized Reinforcement Learning (SORL), designed to enhance both safety and reward performance of the agent, simultaneously. Unlike conventional methods, we tackle the Safe RL problem as a multi-objective policy optimization problem, which allows us to introduce a reward-shaping technique that encourages the agent to explore the environment safely while striving to achieve better performance. This formulation provides an advantage over other model-free Safe RL approaches by eliminating the need to fine-tune the degree of constraining the policy search space (i.e., ϵ_{safe}). As a result, the algorithm will be able to reach a natural trade-off between performance and safety. Through our analysis of SORL, we guarantee the safety of its converged policy through a condition. This condition motivates the introduction of the concept of aggressiveness in our algorithm which provides an intuitive way to tune the hyperparameters of the proposed algorithm.

Finally, to demonstrate the effectiveness of our proposed algorithm we conduct experiments in seven different safety-concerned simulated robotics problems, which are divided into three main safety topics, namely system-level safety, collision avoidance, and safe manipulation. The tasks are visualized in Fig. 1. The algorithm is compared with six other state-of-the-art model-free Safe RL methods. Our results indicate superior performance in both optimality and safety aspects.

*This work was supported by Apera AI and Mathematics of Information Technology and Complex Systems (MITACS) under IT16412 Mitacs Accelerate.

[1] The authors are all affiliated with the Faculty of Engineering and Computer Science, University of Victoria, Victoria, BC, Canada `hmnhonari@uvic.ca`, `mehranght@uvic.ca`, `najjaran@uvic.ca`

II. Related work

Numerous approaches in the literature address safe reinforcement learning. Altman et al. [7] studied RL algorithms under CMDP framework which aims to train them while satisfying certain constraints. In this regard, Lagrangian methods [8] are widely employed for efficient CMDP resolution. Shen et al. [9] introduced the risk-sensitive policy optimization (RSPO) algorithm which decreases the Lagrangian Relaxation (LR) optimization term to 0 sequentially. Furthermore, Tessler et al. [10] introduced Reward-constrained policy optimization (RCPO), which optimizes policy and Lagrange multiplier via dual gradient descent. Furthermore, Zhang et al. [11] proposed the First-Order Policy Optimization (FOCOPS) method, which identifies the optimal update policy through constrained optimization in a nonparametric policy space and subsequently maps it back into the parametric policy space. Stooke et al. [12] incorporated proportional and derivative control into the Lagrange multiplier updates.

Moreover, another approach in the literature to tackle the problem of Safe RL is through modifying unsafe actions locally, meaning they modify the action when it is identified as leading to an unsafe state. Srinivasan et al. [13] uses LR formulation and rejects unsafe policy actions above a safety threshold. Dalal et al. [14] applies a modification to the reward policy's action using a safety layer that solves its formulation analytically. Furthermore, Yu et al. [15] proposed SEditor that trains a safety editing policy that modified the selected actions considered as unsafe into safe actions. Koller et al. [16] introduce a learning-based model predictive control (MPC) framework with high probabilities of safety constraint satisfactions which is attained through a Gaussian process statistical model. Hsu et al. [17] introduced an unsupervised action planning method that stores the agent's recovery actions for leaving unsafe areas in a dedicated replay buffer, subsequently utilizing it when the agent faces an unsafe state. Safe model-based policy optimization (SMBPO) [18] proactively plans a brief horizon into the future to anticipate and prevents safety violations by applying penalties to unsafe trajectories. Recovery RL [19] balances exploration and safety through the utilization of either a backup policy, employed for ensuring safety, or MPC to determine the optimal action sequence.

Finally, it is noteworthy that most of the aforementioned methods require the specification of the extent to which the policy is limited, whether using CMDP framework or attempting to detect unsafe actions, which necessitates the careful fine-tuning of the safety threshold.

III. Preliminaries

In this section, the basic background concepts and formulations are explained. First, the MDP framework is explained and the safety critic and reward penalty framework are discussed.

A. Markov decision process

Reinforcement learning is trained under Markov Decision Process (MDP) framework which is presented as $< S, A, P, r, \gamma, \mu >$ and is outlined in [20]. The MDP is composed of a state space S, an action space A, and a reward function $r : S \times A \times S \mapsto \mathbb{R}$. The transition function $P : S \times A \times S \mapsto [0, 1]$ determines the probability $P(s'|s, a)$ of transitioning from state s to s' by executing action a. The initial state distribution, $\mu : S \mapsto [0, 1]$, and the discount factor, $\gamma \in [0, 1)$, are also included. Finally, the policy $\pi : S \mapsto \Delta_A$ is the probability distribution over actions, with $\pi(a|s)$ indicating the likelihood of taking action a at state s. The value function of a policy π for a state-action pair (s, a) and the resulting recursive equation, called the Bellman equation, can be written as:

$$\begin{aligned} Q^{\pi}_{r}(s,a) &= \mathbb{E}_{s_t\sim P, a_t\sim\pi}[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)|s_0=s,a_0=a] \\ &= \mathbb{E}_{s'\sim P}[r(s,a)+\gamma V^{\pi}_{r}(s')] \end{aligned} \tag{1}$$

The ultimate objective of an RL algorithm is to maximize the expected discounted cumulative return given the initial state distribution μ:

$$\pi^* = \underset{\pi\in\Pi}{\operatorname{argmax}}\ J^{\pi}_{r} = \underset{\pi\in\Pi}{\operatorname{argmax}}\ \mathbb{E}^{\pi}_{s_0\sim\mu}[\sum_{t=0}^{\infty}\gamma^t r_t] \tag{2}$$

B. Safety critic

The safety critic Q^{π}_{safe} as described in [13], is based on the safety-aware MDP framework, which is represented as $< S, A, P, r, c, \gamma, \gamma_{safe} >$. The safety critic's discount factor is denoted by γ_{safe}, and the safety signal $c(s)$ is used to determine whether a given state s is safe or not:

$$c(s) = \begin{cases} 1 & \text{if } s \in S_{unsafe} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

The main purpose of the safety critic is to estimate the likelihood of a policy failure in the future, based on the expected cumulative discounted probability of failure:

$$\begin{aligned} Q^{\pi}_{safe}(s,a) &= \mathbb{E}_{s_t\sim P, a_t\sim\pi}\big[c(s)+(1-c(s))\sum_{t=1}^{\infty}[\gamma^t_{safe}c(s_t)]\big] \\ &= \Pr[c(s)=1]+\gamma_{safe}\mathbb{E}_{s'\sim P}\big[(1-c(s))V^{\pi}_{safe}(s')\big] \end{aligned} \tag{4}$$

C. Reward penalty framework

Generally, the main purpose of Safe RL algorithms is to identify, and avoid, the set of states that violate the safety constraints, which are refered to as S_{unsafe}. To do so, similar to the previous works on Safe RL [18], [21], we can characterize a subset of the state space that are not considered unsafe but will lead to an unsafe state inevitably:

Definition 1: A state $s \in S$ is considered **Irrecoverable** if for any sequence of actions $a_0, a_1, a_2, ...$, starting from state $s_0 = s$ and following the transition probability $s_{t+1} \sim P(s_t, a_t)$ $\forall t \in \mathbb{N}$, there exists some time step $\bar{t} \in \mathbb{N}$ s.t. $s_{\bar{t}} \in S_{unsafe}$.

Naturally, based on the definition, the safe state space S_{safe} encompasses the subset of the state space which are neither categorized as unsafe nor irrecoverable. Correspondingly, an action is considered safe if executing it leads to a safe state. Furthermore, we can assume a soon occurrence of safety violation after entering into it an irrecoverable state:

Assumption 1: For any state $s_0 \in S_{\text{irrecoverable}}$ and for any sequence of actions $a_0, a_1, \ldots$ starting from s_0 and $s_{t+1} \sim P(s_t, a_t)$ $\forall t \in \mathbb{N}$, there exists $\bar{t} \in \{1, ..., H^*\}$ s.t. $s_{\bar{t}} \in S_{\text{unsafe}}$.

Finally, based on the reward penalty MDP $< S, A, P, \tilde{r}, \gamma >$ which is introduced in [18], the reward function can be modified as:

$$\tilde{r}(s,a) = \begin{cases} r(s,a) & \text{if } s \notin S_{unsafe} \\ -C & \text{otherwise} \end{cases} \tag{5}$$

where C satisfies the following inequality:

$$C > \frac{r_{max} - r_{min}}{\gamma^{H^*}} - r_{max} \tag{6}$$

The terminal state cost $C \in \mathbb{R}$ is used to penalize the RL agent when unsafe trajectories are executed.

IV. Method

In this work, a model-free safe reinforcement learning algorithm is proposed which uses safety signals to avoid unsafe regions. In the training process, the safety critic is used to modify the reward function such that exploration in unsafe regions is prevented.

A. Multi-Objective Policy Optimization

Unlike the conventional Safe RL setting where the algorithm is trained under the Constrained MDP framework, we formulate SORL in a multi-objective policy optimization setting [22], [23]. To this end, we propose *Safety-aware reward penalty MDP* $< S, A, P, \tilde{r}, c, \gamma, \gamma_{safe} >$. Under this setting, in multi-objective policy optimization, there are multiple (often conflicting) objectives and the aim is to optimize the objectives simultaneously. For this purpose, the policy performance is defined as a 2-dimensional vector:

$$J(\pi) = \begin{bmatrix} J_r(\pi) \\ J_c(\pi) \end{bmatrix} = \begin{bmatrix} \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t r_t] \\ \mathbb{E}_{\tau\sim\pi}[\sum_{t=0}^{\infty} \gamma^t c_t] \end{bmatrix} \tag{7}$$

Furthermore, according to the Multi-Objective optimization literature, the dominance of a policy π relative to π' can be defined as:

$$\text{if} \begin{cases} J_r(\pi) \geq J_r(\pi') \\ \quad\textbf{and} \\ J_c(\pi) \leq J_c(\pi') \end{cases} \Rightarrow J(\pi) \succcurlyeq J(\pi') \tag{8}$$

The field of multi-objective deep reinforcement learning is an active area of research that encompasses various methodologies aimed at optimizing the expected return of a set of potentially conflicting reward functions [24]. One widely-used approach for addressing this challenge involves scalarizing the reward vector using a scalarization function, which enables the optimization of the scalarized function and, subsequently, the optimization of all rewards. In the context of SORL, the objective is to devise a reward-shaping scheme ($\hat{r}$) that optimizes both performance functions by optimizing a single policy:

$$J_{\hat{r}}(\pi) \geq J_{\hat{r}}(\pi') \Rightarrow \pi \succcurlyeq \pi' \tag{9}$$

B. Safety Optimized Reward Shaping

To address Eq. 9, the notion of the reward function as a vector consisting of the actual reward function r and the safety signal function c seems plausible. However, it is not possible to use the safety signal directly in the scalarization function as it is a sparse function and its immediate value does not bear much significance. To this end, we propose the augmented reward function, based on the safety estimate of the safety critic, defined as:

$$\tilde{r}(s_t,a_t) = \begin{cases} [1 - \lambda Q^{\pi}_{safe}(s_t,a_t)]r(s_t,a_t), & \text{if } r(s_t,a_t) \geq 0 \\ \lambda Q^{\pi}_{safe}(s_t,a_t) r(s_t,a_t), & \text{otherwise} \end{cases} \tag{10}$$

where $\lambda > 0$ is defined as the safety critic significance factor. Finally, under the safety-aware reward penalty MDP framework, the final reward shaping will be as follows:

$$\hat{r}(s_t,a_t) = \begin{cases} \tilde{r}(s_t,a_t), & \text{if } s_{t+1} \in S_{safe} \\ -C, & \text{otherwise} \end{cases} \tag{11}$$

C. Safety Guarantee

The aim of this section is to guarantee the safety of the converged policy when Eq. 11 reward shaping scheme is used which will help us during the hyperparameter tuning phase. To prove the theorem, the following assumptions are used:

Assumption 2: The reward function r is bounded in the range $[r_{min}, r_{max}]$ where $r_{min} < 0$ and $r_{max} > 0$.

Assumption 3: The environment is deterministic in terms of the safety violations. In other words, for any state $s \in S$, $\Pr[c(s) = 1]$ is either 0 or 1.

In order to provide a safety guarantee we must first study the irrecoverable states discussed in section III-C. Using Assumption 1 allows for further categorizing the irrecoverable states into levels of unsafety.

Lemma 1: In an environment where Assumption 1 holds, for any trajectory $\tau = \{(s_0,a_0), ..., (s_{|\tau|}, a_{|\tau|})\}$ where $s_0 \in S_{irrecoverable}$, $s_t \sim P(s_{t-1}, a_{t-1})$, and $s_{|\tau+1|} \sim P(s_{|\tau|}, a_{|\tau|}) \in S_{unsafe}$, and for any $t \in \{1, ..., |\tau|\}$ we have:

$$Q^{\pi}_{safe}(s_t,a_t) \geq (\gamma_{safe})^{H^*-t} \tag{12}$$

Proof: Recall that from Assumption 1 we know the length of the trajectory is $|\tau| \leq H^*$. Moreover, at any time step t and its corresponding state s_t, consider the space of all the trajectories that start from s_t: $\mathrm{T}_t = \{\tau' : (s'_0 = s_t, a'_0), \ldots, (s'_{|\tau'|}, a'_{|\tau'|}), s'_{|\tau'+1|} \sim P(s_{|\tau'|}, a_{|\tau'|}) \in S_{unsafe}\}$. Suppose there exists a trajectory $\tau' \in \mathrm{T}_t$ with $|\tau'| > H^* - t$. Consequently, the concatenation $\tau(s_0 : s_t) \cup \tau'$ will have a length greater than H^* which will violate Assumption 1 since there will then exist a trajectory from the irrecoverable state s_0 with a length greater than H^*. Therefore, based on the original trajectory τ, we can conclude that $\max_{\tau' \in \mathrm{T}_t} |\tau'| \leq H^* - t$. Finally, by the definition of the safety critic in Eq. 4 and using Assumption 3, the lower bound can be achieved. ■

It is noteworthy that to establish safety bounds for the safety critic's evaluation in safe states, we need to make assumptions about the policy's behavior and its interactions with the environment because in some cases, the policy may choose unsafe actions even in safe states, making the safety critic's output an expectation that includes both safe and unsafe actions. Therefore, to maintain a broad analysis, we avoid making such limiting assumptions.

Theorem 1: Under the safety-aware reward penalty MDP framework, let the following condition for safety critic significance factor λ hold:

$$\frac{r_{max}}{1-\gamma} - \frac{r_{max}+C}{1-\gamma}\gamma^{|\tau^*_{uwc}|} < \Big(\frac{\gamma_{safe}^{H^*} r_{max}}{1-\frac{\gamma}{\gamma_{safe}}} - \frac{\gamma_{safe}^{H^*} r_{max}}{1-\frac{\gamma}{\gamma_{safe}}}\Big(\frac{\gamma}{\gamma_{safe}}\Big)^{|\tau^*_{uwc}|} + \frac{\gamma_{safe} r_{min}}{1-\gamma}\Big)\lambda \tag{13}$$

where C and $|\tau^*_{uwc}|$ follow Eq. 6 and Eq. 16, respectively. Consequently, for any state s we have: $\hat{Q}^*(s,a) > \hat{Q}^*(s,a')$, where action a is safe, a' is unsafe, and $\hat{Q}^*$ is the Q^* value-function following Eq. 11 reward-shaping.

Proof: By Assumption 1, if a' is unsafe, it is going to lead to an unsafe state in at most H^* steps. Therefore, by Assumption 2, the maximum discounted return in the worst-case scenario can be expressed as a function of the length of the trajectory before reaching the unsafe state:

$$R^{\pi}_{wc}(|\tau|) = \sum_{t=0}^{|\tau|-1} \big(\gamma^t \big[1 - \lambda Q^{\pi}_{safe}(s_t,a_t)\big] r_{max}\big) + \sum_{t=|\tau|}^{\infty} \big(\gamma^t(-C)\big) \tag{14}$$

While it is possible to ignore the safety critic term and upper bound the equation, we provide a tighter upper bound in our case. We first use Lemma 1 to upper bound the function (in the following we use the notation x to indicate the variability of $|\tau|$):

$$\begin{aligned} R^{\pi}_{wc}(x) &\leq \sum_{t=0}^{x-1}(\gamma^t\big[1-\lambda\gamma_{safe}^{H^*-t}\big]r_{max}) + \sum_{t=x}^{\infty}(\gamma^t(-C)) \\ &= \Big(\frac{r_{max}}{1-\gamma} - \frac{\lambda\gamma_{safe}^{H^*} r_{max}}{1-\frac{\gamma}{\gamma_{safe}}}\Big) - \frac{r_{max}+C}{1-\gamma}\gamma^x \\ &\quad + \frac{\lambda\gamma_{safe}^{H^*} r_{max}}{1-\frac{\gamma}{\gamma_{safe}}}\Big(\frac{\gamma}{\gamma_{safe}}\Big)^x = R_{uwc}(x) \end{aligned} \tag{15}$$

To find the maximum unsafe return in the domain $x \in [1, H^*]$, we take the derivative with respect to x and set it to zero: $\partial R^{\pi}_{uwc}(x)/\partial x = 0$. By solving this derivative and ensuring that the trajectory length remains within the range $|\tau_{uwc}| \in [1, H^*]$, we can determine the trajectory length with the highest return as:

$$|\tau^*_{uwc}| = \begin{cases} \left\lfloor \dfrac{\ln\Big(\frac{\lambda\gamma_{safe}^{H^*} r_{max}}{1-\frac{\gamma}{\gamma_{safe}}}\ln\frac{\gamma}{\gamma_{safe}}\Big) - \ln\Big(\frac{r_{max}+C}{1-\gamma}\ln\gamma\Big)}{\ln\gamma_{safe}} \right\rfloor & \text{if} \in [1, H^*] \\ \underset{|\tau|\in\{1,H^*\}}{\operatorname{argmax}}\ R_{uwc}(|\tau|) & \text{otherwise} \end{cases} \tag{16}$$

Hence, Eq. 14 can be upper bounded as:

$$R^{\pi}_{wc}(|\tau|) \leq R_{uwc}(|\tau^*_{uwc}|) \tag{17}$$

Furthermore, executing the safe action a leads to a safe state where a safe trajectory can be generated which does not encounter a safety violation. The discounted return with the minimum reward (Assumption 2) can be lower bounded as:

$$\begin{aligned} \sum_{t=0}^{\infty}(\gamma^t\lambda Q^{\pi}_{safe}(s_t,a_t)r_{min}) &= \lambda r_{min}\sum_{t=0}^{\infty}(\gamma^t Q^{\pi}_{safe}(s_t,a_t)) \\ &\geq \lambda r_{min}\gamma_{safe}\sum_{t=0}^{\infty}(\gamma^t) = \frac{\lambda\gamma_{safe} r_{min}}{1-\gamma} \end{aligned} \tag{18}$$

To establish the inequality, we leverage the observation that $Q^{\pi}_{safe}(s_t,a_t) \leq \gamma_{safe}$. This is based on the fact that for any timestep t, execution of the state-action pair (s_{t-1}, a_{t-1}) leads to the safe state s_t. Hence, Assumption 3 ensures us that $\Pr[c(s_t) = 1] = 0$; consequently, the right hand side equation in Eq. 4 can be simplified as $\gamma_{safe}\mathbb{E}_{s'\sim P}[V^{\pi}_{safe}(s')]$. Finally, because the expectation operator outputs a value within the range of zero and one, the observation is justified. Therefore, since the discounted return of staying safe forever must always be higher than a trajectory that has a safety violation, it suffices that:

$$\Delta = \frac{\lambda\gamma_{safe} r_{min}}{1-\gamma} - R_{uwc}(|\tau^*_{uwc}|) > 0 \tag{19}$$

Rearranging Eq. 19 gives us the condition. ■

It is possible to derive bounds similar to Theorem 1 with other variations of Assumption 2. Moreover, intuitively, the value of Δ in Eq. 19 determines the degree of aggressiveness of the algorithm. The main difference between Δ and the threshold in CMDP-based methods is the fact that in CMDP methods, the threshold first specifies the degree with which the policy is constrained; following that, the constrained optimization formulation determines the tradeoff between the reward objective and the constraint objective. However, using the formulation of Δ allows for direct specification of the tradeoff between the two objectives. The closer this value is to zero, the more the algorithm prioritizes optimality and performance over its failure rate. Hence, for each task, we define the level of aggressiveness in SORL by specifying the value of Δ and fine-tuning λ to align it as closely as possible with the specified value.

D. Safety Optimized Reinforcement Learning Algorithm

The training process of the SORL algorithm is presented in Algorithm 1. The proposed algorithm can be built on top

Algorithm 1 Safety Optimized RL

```
Require: Safety critic significance factor $\lambda$ and $\Delta$, horizon $H^*$
1:  Initialize policy $\pi_\theta$, critic $Q_{\phi_1}, Q_{\phi_2}$, replay buffer $\mathcal{D}$, safety critic $Q^{\psi_1}_{safe}, Q^{\psi_2}_{safe}$, replay buffer $D_{safe}$
2:  for $e = 1, ..., E_{max}$ do
3:      $s_1 \leftarrow env.reset()$
4:      for $t = 1, ..., T_{max}$ do
5:          $a_t \sim \pi_\theta(.|s_t)$
6:          $\hat{c}_t \leftarrow \max\{Q^{\psi_1}_{safe}(a_t|s_t), Q^{\psi_2}_{safe}(a_t|s_t)\}$
7:          $s_{t+1}, r_t, c_t, done \leftarrow env.step(a_t)$
8:          compute empirical $r_{min}, r_{max}$ and update $C$
9:          solve and update $\lambda$ (Eq. 13)
10:         if $c_t == 0$ then
11:             if $r_t > 0$ then $\hat{r}_t \leftarrow \big[1 - \lambda\hat{c}_t\big] r_t$
12:             else $\hat{r}_t \leftarrow \lambda\hat{c}_t r_t$
13:         else
14:             $\hat{r}_t \leftarrow -C$
15:             $\mathcal{D}_{safe} \leftarrow \mathcal{D}_{safe} \cup (s_t, a_t, c_t, \hat{r}_t, s_{t+1})$
16:         $\mathcal{D} \leftarrow \mathcal{D} \cup (s_t, a_t, c_t, \hat{r}_t, s_{t+1})$
17:         train $\pi_\theta, Q_{\phi_1}, Q_{\phi_2}$ on $\mathcal{D}$
18:         train $Q^{\psi_1}_{safe}, Q^{\psi_2}_{safe}$ on $\mathcal{D} \cup \mathcal{D}_{safe}$
19:         if done then Break
```

of any Model-Free RL algorithm and uses two safety critic networks to estimate the cumulative discounted probability of failure. Two replay buffers are utilized, one for storing all the transitions and the other one for storing unsafe transitions. In the training process, the range of the reward function is computed empirically and, as discussed in Section IV-C, we update the value of λ to satisfy Eq. 13 and the predefined Δ value in Eq. 19. It should be noted that in the case where $|\tau^*_{uwc}| \in (1, H^*)$, Δ becomes a non-linear equality which makes it challenging to derive a closed-form solution. Therefore, in practice, given an initial value for λ, we find a solution in the locality of it that satisfies the conditions.

V. Experiment

In the following section, the performance of SORL is studied. Particularly, we aim to investigate two questions:

- How does the safety formulation performs compared with the other model-free off-policy Safe RL algorithms?
- How does the value of Δ in Eq. 19 affect the performance of SORL?

A. Benchmarks and Comparison Methods

In order to evaluate the level of safety that the model can achieve, we execute it on three safety-concerned categories of environments:

- **System-level safety:** RL algorithms are often used to optimally control robots while adhering to the system limits. We assess our proposed model using four MuJoco environments: *Cheetah-no-flip-Velocity, Ant-Velocity, Hopper-Velocity,* and *Walker2D-Velocity*. In these environments, the agent must learn to move faster in the x-direction while avoiding actions that cause the robot to fall and fail. Additionally, safety violations occur if the robot exceeds a certain velocity. We obtained the codebase for Hopper-Velocity, Walker2D-Velocity, and Ant-Velocity from [25]. For Cheetah-no-flip-Velocity, the base environment was adopted from [18] and the Velocity constraint was added to it.
- **Collision Avoidance:** Besides the inherent robot limits, additional constraints from the environment can impact the algorithm. Collision avoidance is one such constraint, where the controller aims to control the robot while preventing collisions with obstacles. We evaluate our algorithms using Ant-Circle [26] and Drone-Circle (from Bullet Safety Gym codebase [27]) environments. These assessments involve controlling robots to move in circular paths while staying within a safety region smaller than the circle's radius.
- **Safe Manipulation:** Finally, one of the important applications of Safe RL is safe manipulation. To this end, we adopt a modified version of the in-hand object manipulation from Gymnasium Robotics [28] which uses a dexterous hand to manipulate an egg to achieve a target pose. In this task, if the hand exerts a normal force more than a threshold (20 N), the egg will get crushed and the agent will fail.

The episode ends whenever a safety violation has been incurred. Moreover, in all the environments, the alive bonus has been eliminated to evaluate the performance of the safety algorithms in situations where the original reward shaping does not explicitly encode safety.

Six model-free Safe RL algorithms are used to showcase the performance of SORL. The comparison algorithms include Lagrangian Relaxation (LR), Safety Q-Functions for RL (SQRL) [13], Model-Free Recovery RL (RRL-MF) [19], Risk Sensitive Policy Optimization (RSPO) [9], and Reward Constrained Policy Optimization (RCPO) [10]. Finally, to study the safety performance of SORL reward shaping, SAC+C is executed which uses Eq. 5 reward scheme.

B. Implementation settings

The codebase for the comparison methods are adopted from [19] codebase, and, to have a fair comparison between the algorithms and evaluate their safety, pretraining is disabled for all of the algorithms. To this end, SORL and the comparison methods are built on top of the Soft Actor-Critic algorithm [29], and, for fair comparison, the general and common hyperparameters of all the algorithms are kept the same. In addition to that, with the help from the problem-specific hyperparamter settings discussed in [18], we tune the parameters of the comparison algorithms for each benchmark problem. Moreover, for the hyperparameters relating to SORL, we set $H^* = 10$ for all the cases and tune the proposed algorithm based on the value of Δ. For each task, the value of Δ used to execute SORL is shown in Fig. 2. During our experimentation of various Δ values within the specified environments, we observed significant

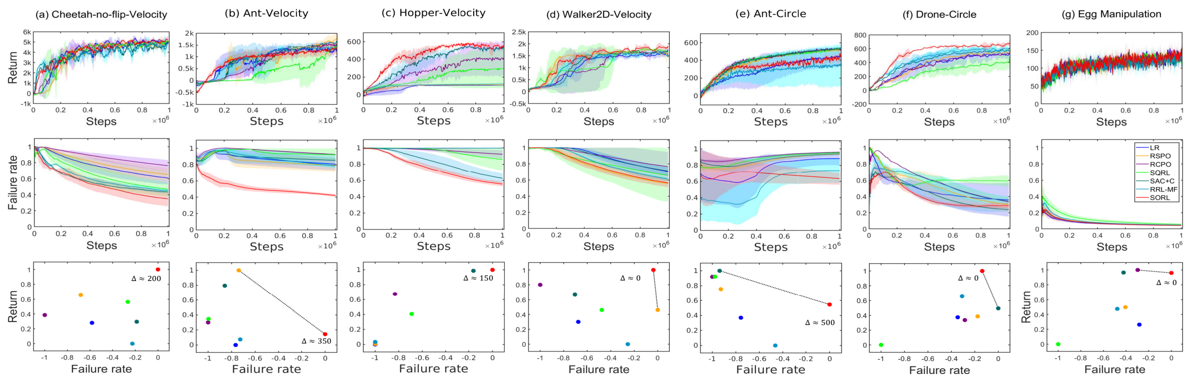

Fig. 2: Benchmark results of SORL compared with six other Safe RL algorithms. **(Top row):** Return values achieved during the training phase (higher is better). **(Middle row):** Episodic failure rates suffered during the training phase (lower is better). **(Bottom row):** Pareto optimality plot corresponding to the return and failure rate values (closer to the top right corner is better). They also display the specific value of Δ employed for executing SORL. For easier comparison, the return and failure rate values are normalized and the failure rate is scaled to lie within -1 and 0. The Pareto optimality solutions are highlighted through the dotted line.

variations in the performance of SORL when its change of value is near the magnitude of 50. The magnitude of change in Δ can be largely attributed to the choice of γ and γ_{safe}. The results illustrate the mean and variance of the execution of the algorithms with independent random seeds.

C. Results

For the performance comparison of the different Safe RL algorithms, as depicted in Fig. 2, we report the reward performance and the failure rate of the algorithms. Based on the learning curves, we plot the pareto optimality of the algorithms based on Eq. 9. Our results show dominant and superior performance of SORL in both aspects in Fig. 2(a) and 2(c). Furthermore, in Fig. 2(b), 2(e), and 2(g) we can see that the proposed algorithm attains significantly better safety performance while achieving comparable returns. Importantly, the suboptimality of the converged policies of the comparison methods in Fig. 2(b) and 2(c) in one or both aspects of performance can be seen. Finally, the results in Fig. 2(d) and 2(f) illustrate SORL's consistently higher returns while also maintaining near-dominant safety performance. Thus, we observe that SORL can strike a great balance between safety and optimality offering a great novel solution for safe performance among Safe RL algorithms.

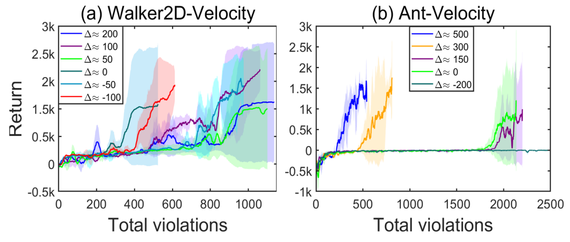

Fig. 3: Undiscounted return of the policy versus the total number of violations during the training phase.

D. Ablation Analysis

In this section, we study the effect of Δ on SORL's performance in two environments. We report the undiscounted return of the policy the algorithm achieves whenever a safety violation has occurred which can show sample efficiency of the algorithm in terms of safety. To gain a better understanding of the effect of Δ, we also chose negative values to study its performance under too aggressive specifications. As depicted in Fig. 3, while being aggressive in Walker2D-Velocity helps SORL attain higher returns in lower number of constraint violations, better performance in Ant-Velocity requires more conservativeness. This may be due to the difference in the dynamics of the robots and their constraints since Ant-Circle also requires a more conservative Δ value. The dynamics differences is more evident in the comparison between $(-100, -50)$ (where there is no safety guarantee) with $\Delta \approx (500, 300)$ in their respective tasks which indicates while being a little more aggressive in one helps in the reduction of the number of constraint violations, the opposite holds true for the other task.

VI. Conclusions

This paper focuses on the problem of safe exploration and decision-making for RL agents. A novel Safe RL approach based on multi-objective policy optimization framework was proposed which optimized the policy toward optimality and safety, simultaneously. Through theoretical analysis, the safety of SORL's converged policy was guaranteed through a condition which allowed the introduction of the concept of aggressiveness. The concept provided an intuitive way to tune SORL's safety-related hyperparameter. Finally, three main safety topics (viz., system-level safety, collision avoidance, and safe manipulation) were studied through seven different tasks in total. We evaluated reward and safety performance of the proposed algorithm against six other state-of-the-art model-free Safe RL approaches. The results showed SORL's great capability in attaining better safety performance while achieving better or comparable returns.

References

[1] J. García and D. Shafie, "Teaching a humanoid robot to walk faster through safe reinforcement learning," *Engineering Applications of Artificial Intelligence*, vol. 88, p. 103360, 2020.

[2] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 2811–2817.

[3] D. Isele, A. Nakhaei, and K. Fujimura, "Safe reinforcement learning on autonomous vehicles," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–6.

[4] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[5] 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.

[6] S. Gu, L. Yang, Y. Du, G. Chen, F. Walter, J. Wang, Y. Yang, and A. Knoll, "A review of safe reinforcement learning: Methods, theory and applications," *arXiv preprint arXiv:2205.10330*, 2022.

[7] E. Altman, *Constrained Markov decision processes: stochastic modeling*. Routledge, 1999.

[8] D. Bertesekas, "Nonlinear programming. athena scientific," *Belmont, Massachusetts*, 1999.

[9] Y. Shen, M. J. Tobia, T. Sommer, and K. Obermayer, "Risk-sensitive reinforcement learning," *Neural computation*, vol. 26, no. 7, pp. 1298–1328, 2014.

[10] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," *arXiv preprint arXiv:1805.11074*, 2018.

[11] Y. Zhang, Q. Vuong, and K. Ross, "First order constrained optimization in policy space," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 338–15 349, 2020.

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

[13] K. Srinivasan, B. Eysenbach, S. Ha, J. Tan, and C. Finn, "Learning to be safe: Deep rl with a safety critic," *arXiv preprint arXiv:2010.14603*, 2020.

[14] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, "Safe exploration in continuous action spaces," *arXiv preprint arXiv:1801.08757*, 2018.

[15] H. Yu, W. Xu, and H. Zhang, "Towards safe reinforcement learning with a safety editor policy," *Advances in Neural Information Processing Systems*, vol. 35, pp. 2608–2621, 2022.

[16] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, "Learning-based model predictive control for safe exploration," in *2018 IEEE conference on decision and control (CDC)*. IEEE, 2018, pp. 6059–6066.

[17] H.-L. Hsu, Q. Huang, and S. Ha, "Improving safety in deep reinforcement learning using unsupervised action planning," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5567–5573.

[18] G. Thomas, Y. Luo, and T. Ma, "Safe reinforcement learning by imagining the near future," *Advances in Neural Information Processing Systems*, vol. 34, pp. 13 859–13 869, 2021.

[19] 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.

[20] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[21] A. Hans, D. Schneegaß, A. M. Schäfer, and S. Udluft, "Safe exploration for reinforcement learning." in *ESANN*. Citeseer, 2008, pp. 143–148.

[22] A. Abdolmaleki, S. Huang, L. Hasenclever, M. Neunert, F. Song, M. Zambelli, M. Martins, N. Heess, R. Hadsell, and M. Riedmiller, "A distributional view on multi-objective policy optimization," in *International Conference on Machine Learning*. PMLR, 2020, pp. 11–22.

[23] A. Abdolmaleki, S. H. Huang, G. Vezzani, B. Shahriari, J. T. Springenberg, S. Mishra, D. TB, A. Byravan, K. Bousmalis, A. Gyorgy *et al.*, "On multi-objective policy optimization as a tool for reinforcement learning," *arXiv preprint arXiv:2106.08199*, 2021.

[24] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[25] J. Ji, B. Zhang, J. Zhou, X. Pan, W. Huang, R. Sun, Y. Geng, Y. Zhong, J. Dai, and Y. Yang, "Safety gymnasium: A unified safe reinforcement learning benchmark," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

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

[27] S. Gronauer, "Bullet-safety-gym: A framework for constrained reinforcement learning," mediaTUM, Tech. Rep., 2022.

[28] R. de Lazcano, K. Andreas, J. J. Tai, S. R. Lee, and J. Terry, "Gymnasium robotics," 2023. [Online]. Available: http://github.com/Farama-Foundation/Gymnasium-Robotics

[29] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.