Reducing Safety Interventions in Provably Safe Reinforcement Learning

Jakob Thumm[1], Guillaume Pelat[1], and Matthias Althoff[1]

***Abstract*— Deep Reinforcement Learning (RL) has shown promise in addressing complex robotic challenges. In real-world applications, RL is often accompanied by failsafe controllers as a last resort to avoid catastrophic events. While necessary for safety, these interventions can result in undesirable behaviors, such as abrupt braking or aggressive steering. This paper proposes two safety intervention reduction methods: proactive replacement and proactive projection, which change the action of the agent if it leads to a potential failsafe intervention. These approaches are compared to state-of-the-art constrained RL on the OpenAI safety gym benchmark and a human-robot collaboration task. Our study demonstrates that the combination of our method with provably safe RL leads to high-performing policies with zero safety violations and a low number of failsafe interventions. Our versatile method can be applied to a wide range of real-world robotic tasks, while effectively improving safety without sacrificing task performance.**

I. INTRODUCTION

A. *Motivation*

Reinforcement learning (RL) has emerged as a promising approach for solving complex tasks in a variety of robotic applications, such as manipulation [1]–[3], mobile robots [4]–[6], and autonomous driving [7]–[9]. Recent works have shown that safety can be guaranteed for RL agents using additional system knowledge [10]–[18]. However, these safety mechanisms can result in undesirable behavior, such as sudden braking or high torques, which can be uncomfortable for humans or demanding for robot hardware. In addition, bringing the system to an invariably safe state (ISS) [19], e.g., a full stop through a failsafe maneuver, can lead to significant recovery times. Consequently, keeping the number of safety interventions as low as possible is desirable.

This work addresses the problem of safe reach-avoid robotic tasks in environments with static and dynamic obstacles. Specifically, the robot aims to reach a target position while avoiding collisions with obstacles. In this work, we guarantee the safety of robots using a generalized version of our proposed shield in [18]. We seek to minimize the number of safety interventions while guaranteeing safety to promote natural and interference-free robot behavior.

Our two proposed approaches for achieving the desired reduction of safety interventions are *proactive replacement* and *proactive projection*. The integration of these methods in the safe RL learning cycle is illustrated in Fig. 1. We first check if the RL action potentially triggers a failsafe intervention. If that is the case, we can either proactively replace the action with a verified randomized action, or proactively project the action to a verified action that lies nearby in Euclidean space. Our safety shield and the proposed proactive replacement and projection methods utilize set-based reachability analysis, making them real-time capable for most high-dimensional systems. We incorporate all possible obstacle trajectories with limited prior knowledge, system and measurement uncertainties, and time delays in our reachability analysis. To our best knowledge, we are the first to propose a technique that can significantly reduce the number of safety interventions in provably safe RL for any robotic environment. We compare our two approaches to constrained RL, which tries to solve the original RL problem while keeping the number of safety interventions below a predetermined threshold.

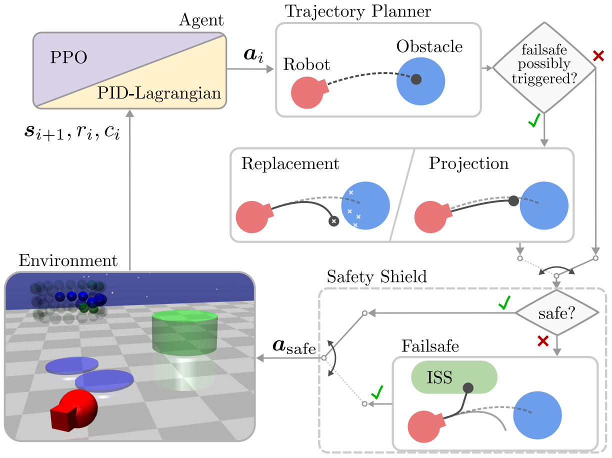

Fig. 1. Proposed intervention reduction technique: First, the agent generates an action a_i, which is then transformed into an intended trajectory via a trajectory planner. In case the intended trajectory would potentially trigger a failsafe controller, we replace it using proactive replacement or proactive projection. We verify the chosen trajectory during execution and fall back on a failsafe trajectory that ends in an invariably safe state (ISS) when needed.

B. *Related work*

The safety of RL agents has been a significant concern since the introduction of RL and deep RL [20], [21]. To broadly categorize safe RL, Brunke et al. [22] classify safe RL methods according to their types of constraints, including soft, probabilistic, and hard constraints.

This work does not rely on probabilistic constraint methods, so readers interested in this topic are referred to [22].

[1]The authors are with the School of Informatics, Technical University of Munich, 85748 Garching, Germany. {jakob.thumm, guillaume.pelat, althoff}@tum.de

The most common soft constraint method is constrained RL, which aims to learn the policy with the highest reward while limiting the number of constraint violations [20]. The Lagrangian method [20] is a popular way to address constrained RL problems by converting them to a dual problem, following constrained optimization theory [23, Chapter 5] and optimizing the Lagrangian multiplier in conjunction with the RL policy. More recent works, such as constrained policy optimization [24], constrained RL with a proportional–integral–derivative-controlled Lagrange multiplier (PID-Lagrangian) [25], and worst-case soft actor-critic [26] build on the Lagrangian method and make it applicable to deep RL. A drawback of constrained RL is that it cannot guarantee safety in a provable way, as the learned behavior is not formalized or proven. Additionally, most constrained RL implementations have a non-zero constraint violation threshold, making it impossible to ensure safety with complete certainty.

This work falls in the group of provably safe RL, which fulfills hard constraints by incorporating additional system knowledge in the form of abstract models into the RL framework. Krasowski et al. [27] categorizes provably safe RL into three classes based on the methods used to alter the actions of the RL agent: action replacement [11], [16], [28], [29], action projection [13], [14], [30], [31], and action masking [12], [15], [32]. However, both action projection and replacement can cause noticeable interventions with the adverse effects mentioned earlier. To address this issue, some works have used a negative reward to penalize safety interventions [11], [13], [16]. Nonetheless, this approach requires careful hand-tuning and often results in either no reduction of interventions or significant performance loss, as we will discuss in more detail later. To evaluate the effectiveness of safe RL methods, the OpenAI safety gym [33] has become a widely used benchmark, offering a range of tasks that require balancing performance and safety considerations.

C. Contributions

Our contributions are fourfold. First, we introduce the first RL agent to the OpenAI safety gym [33] that guarantees zero constraint violations. Second, we propose proactive replacement and proactive projection, two particular forms of action replacement and projection, that significantly reduce the number of failsafe interventions in safe RL and compare it to constrained RL. Third, we evaluate our proposed approaches in the OpenAI safety gym and a human-robot collaboration task[1]. Finally, we demonstrate our proposed method's real-world impact using a six degree-of-freedom (DoF) manipulator in a human environment.

D. Article structure

Sec. II introduces the necessary notation for the RL approaches and set-based reachability analysis, and formalizes a generalization of our previously introduced safety shield to general robotic environments. We then present how the frequency of failsafe interventions can be reduced in Sec. III. Sec. IV discusses the main results of our experiments. Finally, we conclude this article in Sec. V.

II. PRELIMINARIES

This section introduces the required foundations of RL, set-based reachability analysis, and our safety shield.

A. Reinforcement learning

RL aims to find an optimal policy for Markov decision processes (MDPs), defined by the tuple $(\mathcal{S}, \mathcal{A}, R, p, \gamma)$ [34]. This work focuses on continuous state and action spaces $\mathcal{S}$ and $\mathcal{A}$, respectively, as is common in robotic applications. We define the state-transition probability density function $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, \infty)$ to describe the probability of reaching the next state $\boldsymbol{s}_{i+1}$ when choosing action $\boldsymbol{a}_i$ in state $\boldsymbol{s}_i$. The environment provides a reward $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ for each transition, which is discounted by the discount factor $\gamma \in [0, 1]$. The agent learns a stochastic policy $\pi(\boldsymbol{a}_i|\boldsymbol{s}_i)$ from which action $\boldsymbol{a}_i$ is sampled in state $\boldsymbol{s}_i$.

For constrained RL, Altman [20] extends the MDP by a cost function $C : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, a cost limit $d : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and a cost discount $\gamma_C \in [0, 1]$. In practice, we limit ourselves to a fixed cost limit $d \in \mathbb{R}$.

B. Reachablility analysis

In this work, we adopt a set-based reachability analysis approach for ensuring safety and finding low-interfering actions. We consider systems with the dynamics $\dot{\boldsymbol{x}}(t) = f(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{w}(t))$, with bounded control inputs $\boldsymbol{u}(t) \in \mathcal{U}$, disturbances $\boldsymbol{w}(t) \in \mathcal{W}$, and possible states $\boldsymbol{x}(t) \in \mathcal{X}$. We denote the control input trajectory in the time interval $[t_0, t]$ as $\boldsymbol{u}([t_0, t])$ and adopt this notation for all signals. Given such a control input trajectory, an initial state $\boldsymbol{x}_0$, and a disturbance trajectory $\boldsymbol{w}([t_0, t])$, the system follows the trajectory $\chi\left(t, \boldsymbol{x}_0, \boldsymbol{u}([t_0, t]), \boldsymbol{w}([t_0, t])\right)$. The forward reachable set $\mathcal{R}(t)$ of a system starting in the set of initial states $\mathcal{X}_0$ with unknown control inputs comprises all states reachable at time t:

$$\begin{aligned}\mathcal{R}(t) = \{&\chi\left(t, \boldsymbol{x}_0, \boldsymbol{u}([t_0, t]), \boldsymbol{w}([t_0, t])\right) \mid \\ &\boldsymbol{x}_0 \in \mathcal{X}_0, \forall t : \boldsymbol{u}(t) \in \mathcal{U}, \boldsymbol{w}(t) \in \mathcal{W}\}\ .\end{aligned} \tag{1}$$

A general state-feedback control law $\boldsymbol{u}_\text{Z}(t) = \Phi_\text{Z}(x(t)) \in \mathcal{U}$ can also generate an input signal, resulting in the trajectory $\chi_\text{Z}\left(t, \boldsymbol{x}_0, \boldsymbol{u}_\text{Z}([t_0, t]), \boldsymbol{w}([t_0, t])\right)$. We denote the reachable set under a feedback controller by $\mathcal{R}_Z(t)$:

$$\begin{aligned}\mathcal{R}_Z(t) = \{&\chi_\text{Z}\left(t, \boldsymbol{x}_0, \boldsymbol{u}_\text{Z}([t_0, t]), \boldsymbol{w}([t_0, t])\right) \mid \\ &\boldsymbol{x}_0 \in \mathcal{X}_0, \forall t : \boldsymbol{w}(t) \in \mathcal{W}\}\ .\end{aligned} \tag{2}$$

All states reachable during the time interval $[t_0, t_1]$ are given by $\mathcal{R}\left([t_0, t_1]\right) = \bigcup_{t \in [t_0, t_1]} \mathcal{R}(t)$. We denote the set of points that a system can occupy in Euclidean space at time t and a time interval by $\mathcal{O}(\mathcal{R}(t))$ and $\mathcal{O}(\mathcal{R}([t_0, t_1]))$, respectively, and further refer to it as the reachable occupancy (RO). We further introduce a point in Euclidean space as $\boldsymbol{p}$, a ball as $\mathcal{B}(\mathbf{c}, r) = \{\boldsymbol{p} \mid \|\boldsymbol{p} - \mathbf{c}\|_2 \leq r\}$ with center $\mathbf{c}$ and radius r, and a function ball that overapproximates a RO with a ball $\mathcal{O} \subseteq \text{ball}\left(\mathcal{O}\right) = \mathcal{B}(\mathbf{c}, r)$.

[1]Our code and experimental evaluation are available at https://github.com/JakobThumm/safety-intervention-reduction.

C. Safety shield

To ensure the safety of RL agents, we utilize the safety shield for manipulators proposed in [18] and generalize it to arbitrary robotic environments. The safety shield relies on the existence of an ISS that is reachable from any state. An example of an ISS is an entirely stopped robot for manipulators or mobile robots, as per ISO 10218-1 2021 [35]. As we demonstrated in [18], a safety shield for RL is less restrictive when it operates on a higher frequency than the output frequency of the RL agent. We, therefore, execute each RL action $\boldsymbol{a}_i$ for L time steps and perform a safety shield update at every time step.

At each time step k, we calculate an intended and a failsafe trajectory. Without loss of generality, we reset the clock to t_0 at each time step. The intended trajectory χ_I results from the desired agent action output and is executed with the control law $\Phi_\mathrm{I}(\boldsymbol{x}(t), \boldsymbol{a})$ for L time steps. The failsafe trajectory χ_F leads the robot to an ISS using a failsafe controller $\Phi_\mathrm{F}(\boldsymbol{x}(t))$ in F time steps, where F depends on the state of the robot. We append an entire failsafe trajectory to a single step of the intended trajectory to form a so-called shielded trajectory

$$\chi_\mathrm{S} = \begin{cases} \chi_\mathrm{I}\left(t, \boldsymbol{x}_0, \boldsymbol{u}_\mathrm{I}([t_0, t]), \boldsymbol{w}([t_0, t])\right), & t \in [t_0, t_D] \\ \chi_\mathrm{F}\left(t, \boldsymbol{x}_1, \boldsymbol{u}_\mathrm{F}([t_D, t]), \boldsymbol{w}([t_D, t])\right), & t \in [t_D, t_S], \end{cases} \tag{3}$$

with $D = 1$, t_S as the time at the end of the shielded trajectory, and $S = D + F$.

We verify the shielded trajectory by calculating the ROs of the robot $\mathcal{O}(\mathcal{R}_S([t_0, t_S]))$ and the J obstacles $\mathcal{O}_\mathrm{obs}([t_0, t_S]) = \bigcup_{j \in J} \mathcal{O}(\mathcal{R}^j([t_0, t_S]))$ for each partial time interval in the overall time interval $[t_0, t_S]$; see Fig. 3. We then check if the ROs are intersection-free for all partial time intervals using our open-source toolbox SaRA [36]:

$$\mathcal{O}(\mathcal{R}_S([t_{k-1}, t_k])) \cap \mathcal{O}_\mathrm{obs}([t_{k-1}, t_k]) = \varnothing, \forall k \in \{1, \dots, S\}\,. \tag{4}$$

We ensure safety indefinitely through induction by assuming that the robot starts in an ISS and executing the last verified failsafe trajectory if the verification in (4) fails. The set-based representation of the system enables us to guarantee safety in both simulation and real-world applications, thus bridging the simulation-to-reality gap. For a detailed implementation of our safety shield for the OpenAI safety gym, we refer the reader to the Appendix.

III. METHODOLOGY

This work investigates four methods for reducing the frequency of failsafe interventions. The state-of-the-art method is to assign a fixed negative reward for each failsafe intervention [11], [13], [16]. However, the evaluation of reward tuning for the `Point-Button` environment (see Sec. IV), in Fig. 2 shows that this approach often results in either no improvement in the frequency of failsafe interventions or a drastic reduction in performance. To improve this, we introduce two safety reduction techniques, proactive replacement and projection, and compare them with constrained RL.

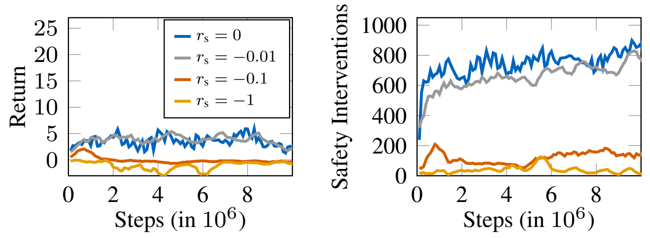

Fig. 2. Reward and shield activation for different values of reward punishment r_s for shield usage with the shielded PPO agent. If the reward punishment is too weak ($r_\mathrm{s} = -0.01$), the shield usage is not affected, and if it is slightly too high ($r_\mathrm{s} = -0.1$), the agent learns to never use the shield at all cost.

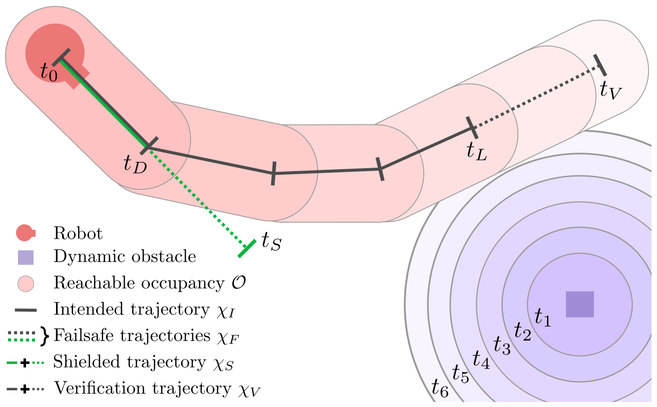
t_0
t_D
t_S
t_L
t_V
t_1
t_2
t_3
t_4
t_5
t_6
Robot
Dynamic obstacle
Reachable occupancy $\mathcal{O}$
Intended trajectory χ_I
Failsafe trajectories χ_F
Shielded trajectory χ_S
Verification trajectory χ_V

Fig. 3. Four types of trajectories can be distinguished in proactive replacement: The intended trajectory results from the action output of the RL agent. To build the shielded trajectory, a failsafe trajectory is appended to a single step of the intended trajectory. The verification trajectory comprises the intended trajectory and a subsequent failsafe trajectory, and is used to validate the RL action. In this particular example, the shielded trajectory is safe, but the RL action is replaced because the verification trajectory collides with the reachable set of the dynamic obstacle.

A. Proactive replacement

Our first approach replaces actions that result in an intersection between the ROs of the robot and obstacles. The new action as any other action does not impair safety due to the safety shield as described in Sec. II-C. To prevent interventions during the RL step, we construct a verification trajectory χ_V as depicted in Fig. 3, composed of the intended trajectory of the RL step followed by a failsafe trajectory. We define the verification trajectory analogous to (3) except that D is replaced by L and S is replaced by $V = L + F$. After constructing the verification trajectory, we verify it for potential failsafe interventions by checking for intersections with the robot's ROs in the time interval $[t_0, t_V]$ using (4), where S is replaced by V.

If the initial verification fails, we sample up to M replacement actions uniformly from the action space and repeat the verification until an action is successfully verified. The agent executes the first verified replacement action found during the RL step. If we cannot find a verifiable replacement action, we execute an environment-specific neutral action, e.g., the zero vector $\boldsymbol{a} = \boldsymbol{0}$. As a result, the proactive replacement

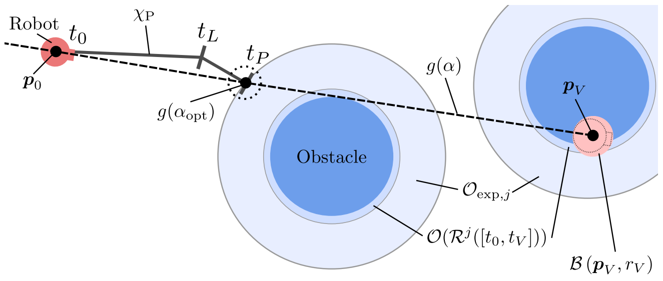

Fig. 4. Our proactive projection method finds the longest intersection-free line segment from the current robot position to the end of the verification trajectory.

might fail to prevent safety interventions, which we evaluate in our experiments in Sec. IV.

B. Proactive projection

Our second intervention reduction approach is a type of action projection. The goal is to find a reachable point in Euclidean space near the position of the robot at the end of the intended trajectory without colliding with the environment. For this, we model the robot as a multi-body system and describe its RO as the union of the ROs of its N bodies $\mathcal{O}(\mathcal{R}) = \cup_{n \in N} \mathcal{O}(\mathcal{R}^n)$. We overapproximate the RO of a robot body with a capsule $\mathcal{C}(\boldsymbol{p}, \boldsymbol{d}, r, l) = \left\{ \boldsymbol{p} + \beta\, l \, \frac{\boldsymbol{d}}{\|\boldsymbol{d}\|_2} \mid \beta \in [-1, 1] \right\} \oplus \mathcal{B}(\boldsymbol{0}, r)$, where $\oplus$ is the Minkowski sum[2]. The ROs of the obstacles in the time interval $[t_0, t_V]$ are expanded by the radius of the robot body capsules $r_\text{exp} = r_V + \epsilon$ using

$$\begin{aligned} \mathcal{O}_\text{exp} &= \bigcup_{j \in J} \mathcal{O}_{\text{exp},j} \\ &= \bigcup_{j \in J} \mathcal{O}\left(\mathcal{R}^j([t_0, t_V])\right) \oplus \mathcal{B}(\boldsymbol{0}, r_\text{exp}) \,. \end{aligned} \tag{5}$$

Depending on the current position $\boldsymbol{p}_0$, we select different projection strategies. Suppose $\boldsymbol{p}_0$ is already in $\mathcal{O}_\text{exp}$. In that case, we use the projection of $\boldsymbol{p}_0$ to the nearest point outside of $\mathcal{O}_\text{exp}$ as the new target position

$$\begin{aligned} &\min && \tilde{\boldsymbol{p}}^\top \tilde{\boldsymbol{p}} \\ &\text{subject to} && (\boldsymbol{p}_0 + \tilde{\boldsymbol{p}}) \notin \mathcal{O}_\text{exp} \,, \end{aligned} \tag{6}$$

where $\tilde{\boldsymbol{p}}$ is the displacement between target and current position. To solve the non-convex optimization problem in (6), we use the l_1 penalty method for sequential convex optimization using trust regions presented in [37, Algorithm 1] and refer the reader to the Appendix for implementation details.

If at the beginning of the RL step $\boldsymbol{p}_0 \notin \mathcal{O}_\text{exp}$, we construct a straight line for each robot body part from its current Carthesian position $\boldsymbol{p}_0^n$ to its predicted Carthesian position at the end of the verification trajectory $\boldsymbol{p}_V^n$ as

$$g^n(\alpha) = \boldsymbol{p}_0^n + \alpha\left(\boldsymbol{p}_V^n - \boldsymbol{p}_0^n\right), \quad \alpha \in \mathbb{R}\,, \tag{7}$$

[2] $\mathcal{A} \oplus \mathcal{B} = \{\boldsymbol{a} + \boldsymbol{b} \mid \boldsymbol{a} \in \mathcal{A}, \boldsymbol{b} \in \mathcal{B}\}$

and $h^n(\alpha) = \boldsymbol{d}_0^n + \alpha\left(\boldsymbol{d}_V^n - \boldsymbol{d}_0^n\right)$. We then look for the longest intersection-free line segment for each robot body part with $\mathcal{G}^n([\alpha_1, \alpha_2]) = \{g^n(\alpha) + \beta\, l^n\, h^n(\alpha) \mid \alpha \in [\alpha_1, \alpha_2], \beta \in [-1, 1]\}$ from the current robot body part position to $\boldsymbol{p}_V^n$ using

$$\begin{aligned} &\max && \alpha_\text{opt} \\ &\text{subject to} && 0 < \alpha_\text{opt} < 1\,, \\ &&& \mathcal{G}^n([0, \alpha_\text{opt}]) \cap \mathcal{O}_\text{exp} = \varnothing, \quad n \in 1, \ldots, N\,. \end{aligned} \tag{8}$$

The line segment approach in (8) is illustrated exemplary for $h^n = \boldsymbol{0}$ and $N = 1$ in Fig. 4. Finally, we use a trajectory planner to plan a trajectory χ_P from $\boldsymbol{p}_0$ to an ISS in an ϵ-bound around $\boldsymbol{p}_V$. The trajectory planner has to ensure that the first L steps of the trajectory have a constant control input. As the projection only checks against intersection in the end-configuration of the verification trajectory, it is not guaranteed that the resulting projected trajectory is intersection-free in (4) with S being replaced by V. Therefore, if χ_P would result in an intersection, we can repetitively reduce α for M times to find a more conservative projection point. If we cannot project the action to a verifiable trajectory χ_P, we execute the neutral action and continue to ensure safety with our safety shield.

C. Constrained RL

We compare the two presented methods with a constrained RL approach that aims to minimize the number of failsafe interventions in an episode. We assign a constant cost C_F to each failsafe intervention in an RL step and train a PID-Lagrangian agent to perform the environment task while staying below a threshold of safety interventions. This approach minimizes safety interventions without any additional implementation effort.

IV. EXPERIMENTS

This section discusses the effects of our proposed intervention reduction methods in two different applications, the OpenAI safety gym benchmark and a human-robot collaboration setting.

A. OpenAI safety gym

Our OpenAI safety gym experiments evaluate the proposed safety reduction methods on two continuous control tasks: `Point-Goal` and `Point-Button`. For both tasks, we consider a cylindrical robot modeled as a point mass that is controlled by its yaw rate and acceleration in the direction of travel. The agent perceives its velocity, acceleration, and distance to the target and the obstacles through a velocity, an acceleration, and a LiDAR sensor, respectively. Each episode starts with randomized positions of the robot, obstacles, and goals.

The `Point-Goal` task requires the agent to navigate to a target area while avoiding hazards. On the other hand, in the `Point-Button` task, the agent needs to move to the correct button from multiple options while avoiding dynamic obstacles called gremlins. For both tasks, the agent incurs a cost of 1 for being in a hazard or an incorrect button area

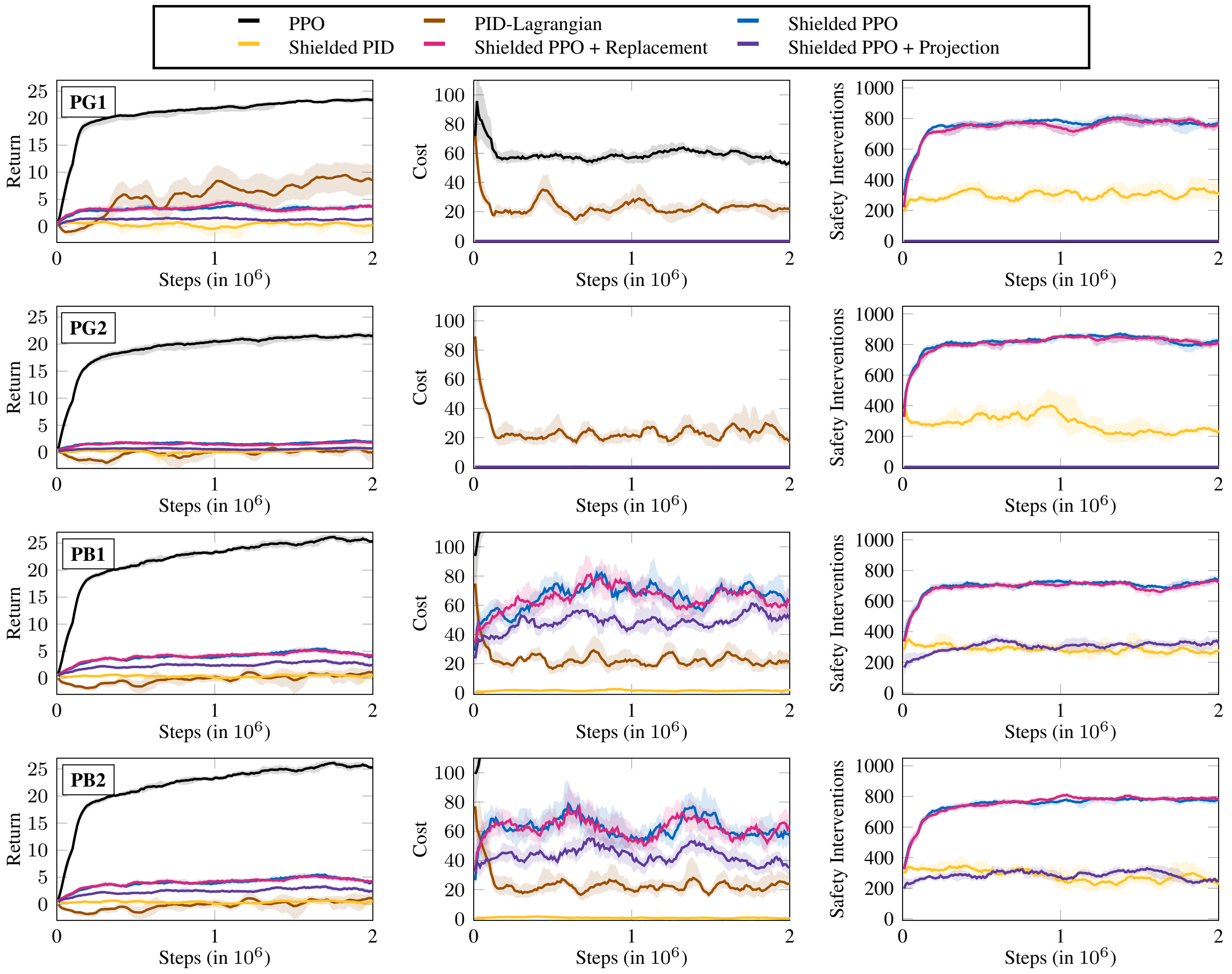

Fig. 5. Mean and its 95% confidence interval over ten random seeds for the OpenAI safety gym environments. From top to bottom: `Point-Goal1`, `Point-Goal2`, `Point-Button1`, and `Point-Button2`. From left to right: the return, cost, and number of safety interventions per episode with length 1000. Our proactive projection method reduces the number of failsafe interventions drastically with only slight performance loss. The costs for the shielded agents in the Button environments are non-zero because we did not account for incorrectly pushed buttons. There are no other collisions with the static environment or the gremlins.

and for colliding with a vase or a gremlin. The shielded PID-Lagrangian agent additionally receives a cost of $C_F = 1$ for failsafe interventions. Both tasks have two difficulty levels, with level 2 featuring more obstacles. Each episode lasts 1000 RL steps, with a new goal randomly selected upon completion. All training results in Fig. 5 show the mean metrics and their 95% confidence intervals[3] over 2 million RL steps on ten random seeds. We kept the training hyperparameters from [25] for all training runs on every environment.

In the `Point-Button` environments, we exclude button constraints from our safety shield as buttons trigger instantaneously from being eligible to invalid upon contact. This breaks the assumption that a stopped robot is in an ISS. Therefore, we only test the capabilities of our shielded PID-Lagrangian agent to reduce shield interventions and incorrect button violations simultaneously.

Our results in Fig. 5 demonstrate the effectiveness of our safety shield in mitigating all environmental costs in the `Point-Goal` tasks and all costs except those due to button constraints in the `Point-Button` task, serving as the first provably safe benchmark in the OpenAI safety gym. Our proactive projection method significantly reduces failsafe interventions of the shielded PPO agent across all environments. In contrast, the proactive replacement method is less effective in lowering safety interventions due to its failure to find a verified replacement action when using our randomized selection. The proactive projection method receives slightly less return than our shielded PPO agent, but still performs better than the shielded PID-Lagrangian agent. In the `Point-Button` tasks, the PID-Lagrangian agent reduces the incorrect button presses significantly. This

[3]We use the default values of the `scipy` bootstrap function.

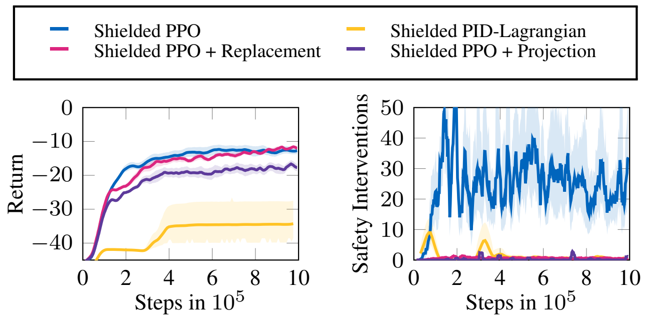

Fig. 6. Mean and its 95% confidence interval over five random seeds of the cumulative reward and failsafe interventions per episode for the human-robot collaboration environment with 1 million RL steps and 400 RL steps per episode.

indicates that constrained RL is a suitable technique for reducing non-safety-critical constraints, which would be hard to shield against otherwise.

B. Human-robot collaboration

To test the transferability of our findings to more complex robotics tasks, we trained a reaching task in a human-robot simulation. We deployed the trained agents on a Schunk LWA 4P manipulator in a real-world setting. Our six DoF manipulator has to move from a given initial end effector position $\boldsymbol{p}_{\text{init}}$ to a target end effector position $\boldsymbol{p}_{\text{goal}}$, with the episode terminating upon goal achievement. The RL agent receives two types of observations: the difference between the target and current end effector position in all three dimensions and the Euclidean distance between the end effector and the human head, i.e., $\boldsymbol{s} = [(\boldsymbol{p}_{\text{goal}} - \boldsymbol{p}_0)^\top, \|\boldsymbol{p}_0 - \boldsymbol{p}_{\text{head}}\|_2]^\top$. The RL actions are the desired change in end effector positions from the current position $\boldsymbol{p}_0$ to the next position $\boldsymbol{p}_L$, i.e., $\boldsymbol{p}_L = \boldsymbol{p}_0 + \boldsymbol{a}$. Our proactive projection technique is only applied to the RO of the end effector, i.e., $N = 1$, to reduce calculation time. The human in the environment follows pre-recorded movements with randomized start positions. As safety is paramount in human environments, we only compare the shielded agents and refer to [18] for a detailed comparison with unsafe agents.

Fig. 6 shows that the shielded PPO agent performs well in the reaching task but triggers a failsafe intervention in 2.5% of the RL steps, with up to 7.5% at the beginning of the training. Our proposed failsafe prevention techniques reduce the number of safety interventions by a factor of 10 to around 0.25%, with no decrease in performance for proactive replacement and only a slight decrease for the projection method. Contrary to the OpenAI safety gym environment, the PID-Lagrangian agent is unable to learn a suitable policy that is both high-performing and reduces failsafe interventions, even after exhaustive hyperparameter optimization. We attribute the limited success of PID-Lagrangian to the increased complexity of the human-robot environment.

The runtime analysis[4] in Fig. 7 illustrates that both proactive replacement and proactive projection do not add a significant overhead to the calculation time in each RL step, despite the highly complex environment. Proactive replacement shows significant outliers that occur when no replacement action can be found. These outliers can be reduced by lowering the number of resamples, or selecting a different replacement strategy.

[4]Run on an Intel® Core™ i7-10750H CPU @ 2.60GHz × 12 and 32GB DDR4 RAM @ 3.2 GHz.

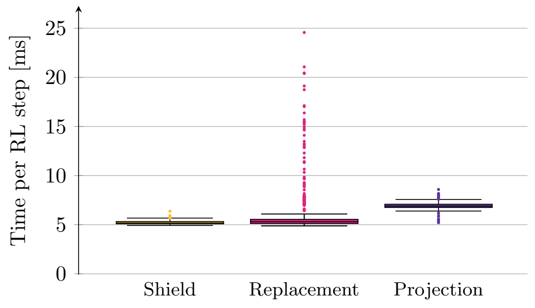

Fig. 7. Comparison of calculation times per RL step for the shielded agent, the shielded agent plus proactive replacement, and the shielded agent plus proactive projection in the human-robot environment. We report the median (black line), 25%- and 75%-quantile (box), the smallest value that is larger than the 25%-quantile − 1.5 · interquartile range (lower whisker), the largest value that is smaller than the 75%-quantile + 1.5 · interquartile range (upper whisker), and outliers (dots).

We also present the performance of our agents on a real-world robot in our supplementary video. The shielded PPO agent consistently fulfills the task, while the action replacement and projection techniques allow the agent to move away from the human RO, resulting in smoother trajectories and fewer safety interventions.

V. CONCLUSIONS

Our results clearly show that our proposed proactive projection method significantly reduces the frequency of failsafe interventions while maintaining competitive performance. Contrary to existing intervention reduction methods, our approach does not require careful parameter tuning to reduce failsafe interventions, making it easily applicable in complex robotic environments. The results of evaluations on the real manipulator further strengthen our findings and make us confident that our approach is transferable to other tasks. To further demonstrate the effectiveness of combining a safety shield with proactive projection, we intend to test it in more challenging real-world human-robot collaboration environments, such as construction sites or manufacturing facilities. These environments pose additional safety challenges due to heavy machinery and high-risk tasks and thus provide an opportunity to evaluate the robustness of our approach.

APPENDIX

A. Adaptions to safety shield for point robot

In the OpenAI safety gym, we consider a robot modelled as a point mass, whose state can be described by its position $\boldsymbol{p}^\top = [p_{\text{x}}, p_{\text{y}}]$, velocity $\boldsymbol{v}^\top = [v_{\text{x}}, v_{\text{y}}]$, and orientation φ, i.e., $\boldsymbol{x} = [p_{\text{x}}, p_{\text{y}}, v_{\text{x}}, v_{\text{y}}, \varphi]^\top$. The robot has two inputs $\boldsymbol{u} =$

$[u_1, u_2]$. The nonlinear system dynamics are given by [33]

$$\dot{\boldsymbol{p}} = \boldsymbol{v}, \tag{9a}$$

$$\dot{\boldsymbol{v}} = \mathbf{R}_{\mathrm{z}}(\varphi)\,[u_1, 0]^\top - \boldsymbol{v}\frac{k_d}{m} \tag{9b}$$

$$\dot{\varphi} = u_2\,, \tag{9c}$$

where $\mathbf{R}_{\mathrm{z}}(\varphi)$ describes the rotation matrix around the z-axis with angle φ, $k_d = 0.01\,\mathrm{kg\,s^{-1}}$, and the mass of the robot is $m = 5.19\,\mathrm{g}$. The input is limited by $|u_1| \leq u_{1,\max} = 9.63\,\mathrm{m\,s^{-2}}$ and $|u_2| \leq u_{2,\max} = 1\,\mathrm{s^{-1}}$. It is important to note that the robot dynamics are only approximated by these equations and do not necessarily reflect the physical behavior of a real-world robot.

We directly obtain the intended trajectory from (9). The aim of the failsafe trajectory is to brake the robot as fast as possible, so the objective function is

$$\max_{u_1,u_2} \left(\|\boldsymbol{v}\|_2^2 - \|\boldsymbol{v} + \Delta t \dot{\boldsymbol{v}}\|_2^2\right)\,, \tag{10}$$

which has a global optimum at $u_{1,\text{opt}} = (\cos(\varphi)v_{\mathrm{x}} + \sin(\varphi)v_{\mathrm{y}})(\frac{k_d}{m} - \frac{1}{\Delta t})$ and $u_{2,\text{opt}} = (\operatorname{atan2}(v_{\mathrm{y}}, v_{\mathrm{x}}) - \varphi)/\Delta t$. Therefore, the failsafe controller uses $\boldsymbol{u} = [\operatorname{clip}(u_{1,\text{opt}}, -u_{1,\max}, u_{1,\max})\,, \operatorname{clip}(u_{2,\text{opt}}, -u_{2,\max}, u_{2,\max})]$, where $\operatorname{clip}(a, b, c)$ clips a to the interval $[b, c]$.

For the reachability analysis, we consider the OpenAI safety gym as a two-dimensional environment. The reachable occupancies of the obstacles are calculated using the velocity-constrained model proposed by Liu et al. [38]. To calculate the reachable occupancy of the point robot, only the position is relevant, so we define its trajectory with abuse of notation as the solution of (9a) $\chi_{\boldsymbol{p}}(t, \boldsymbol{x}_0, \boldsymbol{u}([t_0, t]), \boldsymbol{w}([t_0, t]))$. To achieve fast calculation times, we linearly interpolate the trajectory of the robot between two shield steps t_0 and t_1 as

$$\tilde{\chi}_{\boldsymbol{p}}(t, \boldsymbol{x}_0, \boldsymbol{u}([t_0, t]), \boldsymbol{w}([t_0, t])) = \xi \boldsymbol{p}(t_0) + (1 - \xi)\boldsymbol{p}(t_1)\,, \tag{11}$$

with $\xi \in [0, 1]$. The resulting linearization error in the position can be over-approximated by

$$\zeta = \left\|\tilde{\chi}_{\boldsymbol{p}}\left(\frac{t_1 + t_0}{2}\right) - \chi_{\boldsymbol{p}}\left(\frac{t_1 + t_0}{2}\right)\right\|_2 \leq \frac{d^2 p(t)}{dt^2}\frac{(\Delta t)^2}{8}\,, \tag{12}$$

as shown by Beckert et al. [39] for general point movements with known acceleration limits. For the point robot, this results in $\zeta = \frac{a_{0,\max}(\Delta t)^2}{8m}$. The reachable set of the position of the robot is therefore

$$\mathcal{R}_{\boldsymbol{p}}(t) = (\xi(t)\boldsymbol{p}(t_0) + (1 - \xi(t))\boldsymbol{p}(t_1)) \oplus \mathcal{B}(\mathbf{0}, \zeta)\,, \tag{13}$$

where $\xi(t) = \frac{t_1 - t}{t_1 - t_0}$. To get the reachable occupancy of the robot, we simply add its radius r to the reachable set

$$\mathcal{O}(\mathcal{R}_{\boldsymbol{p}}(t)) = \mathcal{R}_{\boldsymbol{p}}(t) \oplus \mathcal{B}(\mathbf{0}, r)\,. \tag{14}$$

The linearization in (11) simplifies the reachable occupancy of the robot between two shield steps to a capsule, which allows for fast intersection checking with the environment.

The OpenAI safety gym only provides lidar measurements of the obstacles, which can result in occlusions of obstacles by other obstacles. Such occlusions can be handled using set-based predictions as shown exemplary for an autonomous driving task in [40]. However, this solution increases the size of the reachable sets, making the safety shield more restrictive. To mitigate this issue, we decided to augment the safety shield with precise information about the exact position of each obstacle. The shielded RL agent, however, does not have access to the exact positions, ensuring a fair comparison with unshielded agents.

B. Implementation details for proactive projection

We specify the constraint function of (6) as

$$-\operatorname{sd}(\boldsymbol{p}_0 + \tilde{\boldsymbol{p}}, \mathcal{O}_{\text{exp},j}) \leq 0, \forall j \in J\,. \tag{15}$$

Here, $\operatorname{sd}(\boldsymbol{p}, \mathcal{O})$ is the signed distance between a point $\boldsymbol{p}$ and a set $\mathcal{O}$ with

$$\operatorname{sd}(\boldsymbol{p}, \mathcal{O}) = \operatorname{sign}\left((\boldsymbol{p}_E - \boldsymbol{p})^\top(\boldsymbol{p}_C - \boldsymbol{p})\right)\|\boldsymbol{p}_E - \boldsymbol{p}\|_2\,, \tag{16}$$

where $\boldsymbol{p}_E$ is the closest point on the edge of $\mathcal{O}$ to $\boldsymbol{p}$, and $\boldsymbol{p}_C$ is the center of $\mathcal{O}$. As the projection at the beginning of an RL step is time-critical, we use a simplified linearization of the inequality constraints for the *convexify* step of [37, Algorithm 1]:

$$h^j(\tilde{\boldsymbol{p}}) = -\operatorname{sd}(\boldsymbol{p}_0 + \tilde{\boldsymbol{p}}, \mathcal{O}_{\text{exp},j}) \tag{17}$$

$$\approx \operatorname{sd}(\boldsymbol{p}_0, \mathcal{O}_{\text{exp},j})\frac{\boldsymbol{p}_E^j - \boldsymbol{p}_0}{\|\boldsymbol{p}_E^j - \boldsymbol{p}_0\|_2^2}\tilde{\boldsymbol{p}} - \operatorname{sd}(\boldsymbol{p}_0, \mathcal{O}_{\text{exp},j})\,. \tag{18}$$

ACKNOWLEDGMENT

The authors gratefully acknowledge financial support by the Horizon 2020 EU Framework Project CONCERT under grant 101016007.

REFERENCES

[1] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” in *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2016.

[2] O. M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, “Learning dexterous in-hand manipulation,” *International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[3] R. Liu, F. Nageotte, P. Zanne, M. de Mathelin, and B. Dresp-Langley, “Deep reinforcement learning for the control of robotic manipulation: A focussed mini-review,” *Robotics*, vol. 10, no. 1, pp. 1–13, 2021.

[4] L. Tai, G. Paolo, and M. Liu, “Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation,” in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2017, pp. 31–36.

[5] K. Zhu and T. Zhang, “Deep reinforcement learning based mobile robot navigation: A review,” *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674–691, 2021.

[6] L. Chang, L. Shan, C. Jiang, and Y. Dai, “Reinforcement based mobile robot path planning with improved dynamic window approach in unknown environment,” *Autonomous Robots*, vol. 45, no. 1, pp. 51–76, 2021.

[7] A. El Sallab, M. Abdou, E. Perot, and S. Yogamani, “Deep reinforcement learning framework for autonomous driving,” in *Proc. of the IS&T Int. Symp. on Electronic Imaging (EI): Autonomous Vehicles and Machines*, 2017, pp. 70–76.

[8] X. Wang, H. Krasowski, and M. Althoff, "Commonroad-rl: A configurable reinforcement learning environment for motion planning of autonomous vehicles," in *Proc. of the IEEE Int. Intelligent Transportation Systems Conf. (ITSC)*, 2021, pp. 466–472.

[9] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2022.

[10] F. Berkenkamp, M. Turchetta, A. P. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," in *Proc. of the Int. Conf. on Neural Information Processing Systems (NeurIPS)*, 2017, pp. 908–919.

[11] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, 2018, pp. 2669–2678.

[12] N. Fulton and A. Platzer, "Safe reinforcement learning via formal methods: Toward safe control through proof and learning," in *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, 2018, pp. 6485–6492.

[13] T.-H. Pham, G. De Magistris, and R. Tachibana, "Optlayer - practical constrained optimization for deep reinforcement learning in the real world," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2018, pp. 6236–6243.

[14] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick, "End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks," in *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, 2019, pp. 3387–3395.

[15] H. Krasowski, X. Wang, and M. Althoff, "Safe reinforcement learning for autonomous lane changing using set-based prediction," in *Proc. of the IEEE Int. Intelligent Transportation Systems Conf. (ITSC)*, 2020, pp. 1–7.

[16] Y. S. Shao, C. Chen, S. Kousik, and R. Vasudevan, "Reachability-based trajectory safeguard (RTS): a safe and fast reinforcement learning safety layer for continuous control," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3663–3670, 2021.

[17] N. Hunt, N. Fulton, S. Magliacane, T. N. Hoang, S. Das, and A. Solar-Lezama, "Verifiably safe exploration for end-to-end reinforcement learning," in *Proc. of the Int. Conf. on Hybrid Systems: Computation and Control (HSCC)*, 2021.

[18] J. Thumm and M. Althoff, "Provably safe deep reinforcement learning for robotic manipulation in human environments," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2022, pp. 6344–6350.

[19] E. I. Liu, C. Pek, and M. Althoff, "Provably-safe cooperative driving via invariably safe sets," in *Proc. of the IEEE Intelligent Vehicles Symp. (IV)*, 2020, pp. 516–523.

[20] E. Altman, "Constrained markov decision processes with total cost criteria: Lagrangian approach and dual linear program," *Mathematical Methods of Operations Research*, vol. 48, no. 3, pp. 387–417, 1998.

[21] 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.

[22] 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, no. 1, pp. 411–444, 2022.

[23] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[24] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2017, pp. 22–31.

[25] A. Stooke, J. Achiam, and P. Abbeel, "Responsive safety in reinforcement learning by pid lagrangian methods," in *Proc. of the Int. Conf. on Machine Learning (ICML)*, 2020, pp. 9133–9143.

[26] Q. Yang, T. D. Simão, S. H. Tindemans, and M. T. J. Spaan, "Wcsac: Worst-case soft actor critic for safety-constrained reinforcement learning," in *Proc. of the AAAI Conf. on Artificial Intelligence (AAAI)*, 2021, pp. 10 639–10 646.

[27] H. Krasowski, J. Thumm, M. Müller, X. Wang, and M. Althoff, "Provably safe reinforcement learning: A theoretical and experimental comparison," in *https://arxiv.org/abs/2205.06750*, 2022.

[28] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin, "A general safety framework for learning-based control in uncertain robotic systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 7, pp. 2737–2752, 2019.

[29] O. Bastani, "Safe reinforcement learning with nonlinear dynamics via model predictive shielding," in *Proc. of the American Control Conf. (ACC)*, 2021, pp. 3488–3494.

[30] S. Gros, M. Zanon, and A. Bemporad, "Safe reinforcement learning via projection on a safe set: How to achieve optimality?" in *Proc. of the IFAC World Congress*, 2020, pp. 8076–8081.

[31] K. P. Wabersich and M. N. Zeilinger, "A predictive safety filter for learning-based control of constrained nonlinear dynamical systems," *Automatica*, vol. 129, 2021.

[32] G. Mason, R. Calinescu, D. Kudenko, and A. Banks, "Assured reinforcement learning with formally verified abstract policies," in *Proc. of the Int. Conf. on Agents and Artificial Intelligence (ICAART)*, 2017, pp. 105–117.

[33] A. Ray, J. Achiam, and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," *arXiv preprint arXiv:1910.01708*, 2019.

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

[35] ISO, "Robotics - safety requirements - part 1: Industrial robots," International Organization for Standardization, Tech. Rep. DIN EN ISO 10218-1:2021-09 DC, 2021.

[36] S. Schepp, J. Thumm, S. B. Liu, and M. Althoff, "SaRA: A tool for safe human–robot coexistence and collaboration through reachability analysis," in *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2022, pp. 4312–4317.

[37] J. Schulman, Y. Duan, J. Ho, A. Lee, I. Awwal, H. Bradlow, J. Pan, S. Patil, K. Goldberg, and P. Abbeel, "Motion planning with sequential convex optimization and convex collision checking," *International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[38] S. B. Liu, H. Roehm, C. Heinzemann, I. Lütkebohle, J. Oehlerking, and M. Althoff, "Provably safe motion of mobile robots in human environments," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2017, pp. 1351–1357.

[39] D. Beckert, A. Pereira, and M. Althoff, "Online verification of multiple safety criteria for a robot trajectory," in *Proc. of the IEEE Conf. on Decision and Control (CDC)*, 2017, pp. 6454–6461.

[40] P. F. Orzechowski, A. Meyer, and M. Lauer, "Tackling occlusions limited sensor range with set-based safety verification," in *Proc. of the IEEE Int. Intelligent Transportation Systems Conf. (ITSC)*, 2018, pp. 1729–1736.