Safe Reinforcement Learning with Chance-constrained Model Predictive Control

Samuel Pfrommer SAM.PFROMMER@BERKELEY.EDU
Tanmay Gautam TGAUTAM23@BERKELEY.EDU
Alec Zhou AULOEHCZ@BERKELEY.EDU
Somayeh Sojoudi SOJOUDI@BERKELEY.EDU
Department of Electrical Engineering and Computer Sciences at UC Berkeley

Editors: R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

Real-world reinforcement learning (RL) problems often demand that agents behave safely by obeying a set of designed constraints. We address the challenge of safe RL by coupling a *safety guide* based on model predictive control (MPC) with a modified policy gradient framework in a linear setting with continuous actions. The guide enforces safe operation of the system by embedding safety requirements as chance constraints in the MPC formulation. The policy gradient training step then includes a safety penalty which trains the base policy to behave safely. We show theoretically that this penalty allows for a provably safe optimal base policy and illustrate our method with a simulated linearized quadrotor experiment.

Keywords: Safe reinforcement learning, model predictive control, chance programming, policy gradient algorithms

1. Introduction

Reinforcement learning has been extensively studied in the context of closed environments, where it has gained popularity for its success in mastering games such as Atari and Go (Sutton and Barto, 1998; Mnih et al., 2015; Silver et al., 2017). A pressing need to deploy autonomous agents in the physical world has introduced a new challenge: agents must be able to interact with their environments in a safe and comprehensible manner. This is especially critical in industrial settings (Dalal et al., 2018).

For safety-critical tasks, the trial-and-error nature of exploration in RL often prevents agent deployment in the real world during training, motivating the use of simulators. However, when dealing with complex environments, simulators may fail to sufficiently model the complexity of the environment (Achiam and Amodei, 2019). Furthermore, reward functions may be unknown a priori, making learning in simulation impossible. This is where methods that guarantee safe exploration during training offer a substantial advantage.

Our work employs policy gradients and model predictive control (MPC) as its primary building blocks to address the safe RL problem. Policy gradient methods learn a parameterized policy to maximize long-term expected rewards using gradient ascent and play a central role in reinforcement learning due to their ability to handle stochasticity, superior convergence properties and training stability, and efficacy in high-dimensional action spaces (Sutton and Barto, 1998). This family of algorithms is also *model-free*, relying solely on reward signals from the environment without

modeling any dynamics. Policy gradient variations have since proliferated under the deep learning paradigm, notably including "natural" policy gradients and actor-critic methods in addition to techniques such as experience replay and importance sampling for better sample efficiency (Peters and Schaal, 2008; Wang et al., 2017).

Model predictive control is a flexible optimal control framework that has seen successes across a wide variety of settings, including process control in chemical plants and oil refineries, power electronics and power system balancing, autonomous vehicles and drones, and building control (Qin and Badgwell, 2003; Rawlings and Mayne, 2009). It is *model-based*, requiring the system dynamics to be identified either a priori or through learning (Koller et al., 2018). Its interpretability lends itself to robust extensions, where system uncertainties and disturbances can be incorporated to probabilistically guarantee agent safety (Koller et al., 2018).

1.1. Related Work

Safety filters are the closest line of work to our proposed algorithm (Wabersich and Zeilinger, 2019, 2021). This is a decoupled method that takes sampled actions from any base policy and uses an MPC controller as the "safety filter" to correct unsafe behaviors. However, these two components function independently, which may lead to conflicting and potentially oscillatory behaviour between the MPC and RL objectives. The computationally taxing safety filter must also be used at both training and test times, making the technique ill-suited for real-world deployment on constrained hardware. Cheng et al. (2019) proposes a related framework that combines model-free RL algorithms with control barrier functions to guarantee safety during training. While this approach accommodates model uncertainty and learns the dynamics online, it is decoupled in a manner similar to safety filters and retains the same drawbacks. Wagener et al. (2021) describes SAILR - an alternative intervention-based approach that utilizes advantage functions to learn a safe policy during training. While empirically the authors demonstrate that this algorithm outperforms other safe RL methods, it is still shown to occasionally violate safety constraints during training.

Constrained reinforcement learning (CRL) aims to formalize the reliability and safety requirements of an agent by encoding these explicitly as constraints within the RL optimization problem. Achiam et al. (2017) proposes a trust-region based policy search algorithm for CRL with guarantees, under some policy regularity assumptions, that the policy stays within the constraints in expectation. This approach cannot be used in applications where safety must be ensured at all visited states. Dalal et al. (2018) addresses the CRL problem by adding a safety layer to the policy that analytically solves an action correction formulation for each state. While this approach guarantees constraint satisfaction, it does not yield a safe policy at the end of training. In Tessler et al. (2018), the constraints are embedded as a penalty signal into the reward function, guiding the policy towards a constraint satisfying solution. Similar to Achiam et al. (2017), safety is not ensured at each state.

Model-based RL methods generally offer higher sample efficiency than their model-free counterparts and can be applied in safety-critical settings with more interpretable safety constraints. This area of work includes learning-based robust MPC (Koller et al., 2018). Berkenkamp et al. (2017) proposes an algorithm that considers safety in terms of Lyapunov stability guarantees. More specifically, the approach demonstrates how, starting from an initial safe policy, the safe region of attraction can be expanded by collecting data within the safe region and adapting the policy.

Imitation learning attempts to learn a policy by direct supervision from expert demonstration. This approach is frequently plagued by distribution mismatch and compounding errors. Dataset

Aggregation (DAgger) is an iterative method used to mitigate these drawbacks by reducing the distribution mismatch (Ross et al., 2011). In (Menda et al., 2019), the authors extend DAgger to EnsembleDAgger, which addresses the challenge of safe exploration by quantifying the confidence of the learned policy. It does this by using an ensemble of neural networks to estimate the variance of the action proposed by the learned policy at a particular state. While showing solid empirical performance, EnsembleDAgger lacks formal safety guarantees.

1.2. Paper Contributions

Our approach wraps a policy gradient *base policy* with an MPC-based *safety guide* that corrects any potentially unsafe actions. The base policy learns to optimize the agent's long-term behaviour, while the MPC component accounts for state-space safety constraints. By optimizing over an action distribution in the safety guide, we show that adding a safety penalty to the policy gradient loss allows for a provably safe optimal base policy. This resolves tension between the base policy and the safety guide and permits the removal of the computationally expensive safety guide after training.

2. Background

2.1. Notation

Throughout this work, we let $s_t \in \mathcal{X}$, $a_t \in \mathcal{A}$, and $r(s_t, a_t) \in \mathbb{R}$ refer to the state, action, and reward at time t. A sequence of states and actions is termed a trajectory and denoted by τ, and the sum of rewards over a trajectory is denoted $r(\tau)$. We focus on the setting where $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{A} \subseteq \mathbb{R}^m$. Since our action space is continuous, we represent a stochastic policy as $\pi : \mathcal{X} \to \mathcal{N}(\mathcal{A})$, where $\mathcal{N}(\mathcal{A})$ is a Gaussian distribution over actions. More specifically, we can write $\pi(\,\cdot \mid s) = \mathcal{N}(\mu(s), \Sigma(s))$ for some Gaussian mean $\mu(s)$ and covariance $\Sigma(s)$. The space of such policies is denoted as Π. When such a policy is parameterized by a vector θ, we use the notation π_θ. With some abuse of notation, we write $\tau \sim \pi$ to denote sampling a trajectory from the policy π; similarly, $(s, a) \sim \pi$ denotes sampling a state s from the stationary distribution induced by π and then sampling a from $\pi(\,\cdot \mid s)$. Furthermore, $\|\cdot\|_p$ denotes the ℓ_p-norm within $\mathbb{R}^n$. The symbol $\mathbf{1}_n$ defines an n-dimensional column vector of ones, and $\mathrm{Tr}(A)$ denotes the trace of the matrix A. $\mathbb{E}_{p(x)}[\cdot]$ is the expectation operator with respect to the probability distribution $p(x)$.

2.2. Policy Gradient

Policy gradient methods attempt to find the optimal parameters θ^* for the objective

$$\max_\theta J(\pi_\theta), \qquad J(\pi_\theta) = \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{M} \gamma^t r(s_t, a_t)\right]. \tag{1}$$

The vanilla policy gradient approach performs gradient ascent to maximize this objective (Williams, 1992). The gradient can be approximated with the Monte-Carlo estimator

$$\nabla_\theta J(\pi_\theta) \approx \frac{1}{N}\sum_{j=1}^{N}\sum_{t=0}^{M} \nabla_\theta \log \pi_\theta(a_t^j \mid s_t^j) \sum_{t'=t}^{M} \gamma^{t'-t} r(s_{t'}^j, a_{t'}^j), \tag{2}$$

with $0 \leq \gamma < 1$ a discount factor. While many variance-reduction techniques can be used to improve (2), for simplicity of exposition we employ this basic formulation.

2.3. Model Predictive Control

Model predictive control is a purely optimization-based planning framework. Given a dynamics model and a set of state and action constraints (safety requirements, physical limitations, etc.), the finite-horizon MPC problem computes the near-optimal open-loop action sequence that minimizes a specified cumulative cost function. The first of these actions is executed, and the entire optimization repeats on the next time step. While the MPC framework offers concreteness in its constraints, it requires a pre-specified reward function and is incapable of forming reward-maximizing plans beyond its horizon.

2.4. Problem Setting

We represent the environment dynamics as a known linear time-invariant system

$$s_{t+1} = As_t + Ba_t, \tag{3}$$

with initial state s_0, dynamics matrix $A \in \mathbb{R}^{n\times n}$, and input matrix $B \in \mathbb{R}^{n\times m}$. The safety requirements are captured by a polyhedral state safe set $\mathcal{S} \subset \mathcal{X}$. The goal is to learn a policy which maximizes the cumulative reward signal r while ensuring that the exploration during training is safe at all times, i.e. $s_t \in \mathcal{S}$ for all t.

3. Method

A high-level overview of our method combining policy gradient learning and model predictive control is displayed in Figure 1. We first outline the construction of the *safety guide*, which solves a chance-constrained MPC optimization to enforce the safety of actions proposed by the underlying base policy. This allows for guaranteed safety during training time with arbitrarily high probability. Section 3.2 discusses how the safety guide is incorporated into the overarching policy optimization.

3.1. Safety Guide Design

The safety guide solves a convex MPC problem for each time step during training to ensure system safety. This safety guide is not needed later at test time, which is justified theoretically in Section 4. We begin by making the following assumption.

Assumption 1 *There exists a polyhedral terminal safe set $\mathcal{S}_T \subset \mathcal{S} \subset \mathcal{X}$ that is* invariant*, meaning that for any state $s \in \mathcal{S}_T$, there exists a sequence of control inputs that keep the system in $\mathcal{S}_T$ for all subsequent time steps.*

The construction of invariant sets has a well-established theory due to its applications in systems and control. For linear systems, several recursive algorithms have been proposed to construct polyhedral invariant sets (Gilbert and Tan, 1991; Pluymers et al., 2005), with nonlinear systems considered in Bravo et al. (2003); Korda et al. (2013).

Algorithm 1 specifies the safety guide optimization problem. Intuitively, the safety guide attempts to find an action distribution that is as close as possible to that outputted by the base policy, subject to safety constraints. Taking inspiration from techniques in the obstacle avoidance literature (Blackmore et al., 2011), we formulate this in a chance-constrained model predictive fashion.

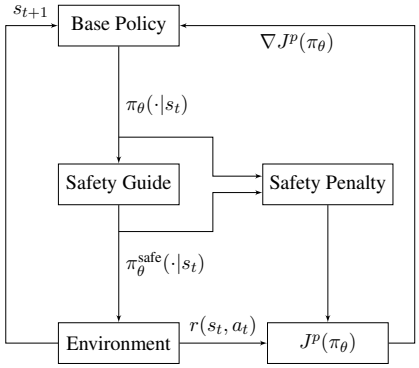


Figure 1: The training scheme. The base policy π_θ suggests a distribution over actions given s_t. The safety guide potentially shifts this distribution to ensure safety and outputs the distribution $\pi_\theta^{\text{safe}}(\,\cdot \mid s_t)$, from which the next action is sampled. The environment reward and a safety penalty on the distance between these two distributions are combined in the objective, whose gradient is approximated using Monte Carlo rollouts.

Algorithm 1 Safety guide

Input: starting state s', base policy mean $\mu_\theta^a(s')$ and covariance $\Sigma_\theta^a(s')$

Parameters: Planning horizon H, safety tolerance ϵ, system matrices A and B, state safe set $\mathcal{S}$, safe terminal set $\mathcal{S}_T$, feasible action set $\mathcal{A}$

Solve the convex optimization problem

$$
\begin{aligned}
\underset{\substack{\overline{\Sigma}_0^a \\ \mu_0^a,\dots,\mu_H^a \\ \mu_0^s,\dots,\mu_H^s}}{\arg\min} \quad & \mathrm{KL}\left(\mathcal{N}\big(\mu_0^a, \overline{\Sigma}_0^a \overline{\Sigma}_0^{a\intercal}\big) \parallel \mathcal{N}\big(\mu_\theta^a(s'), \Sigma_\theta^a(s')\big)\right) \\
& \mu_{t+1}^s = A\mu_t^s + B\mu_t^a, \;\; 0 \le t < H \\
& \overline{\Sigma}_t^s \coloneqq A^t B \overline{\Sigma}_0^a, \;\; 0 \le t < H \\
& \Pr\left[\mathcal{N}(\mu_t^s, \overline{\Sigma}_t^s \overline{\Sigma}_t^{s\intercal}) \notin \mathcal{S}\right] < \epsilon, \;\; 0 \le t < H \\
& \Pr\left[\mathcal{N}(\mu_H^s, \overline{\Sigma}_H^s \overline{\Sigma}_H^{s\intercal}) \notin \mathcal{S}_T\right] < \epsilon, \;\; t = H \\
& \mu_t^a \in \mathcal{A}, \;\; 0 < t \le H \\
& \mu_0^s = s'
\end{aligned}
$$

If infeasible **then** relax constraints and resolve

Return $\pi_\theta^{\text{safe}}(\,\cdot \mid s') = \mathcal{N}\big(\mu_0^{a*}, \overline{\Sigma}_0^{a*} \overline{\Sigma}_0^{a*\intercal}\big)$

VARIABLES. The optimization variables consist of a sequence of state means μ_t^s and open-loop control actions μ_t^a over a planning horizon of length H, with the first action containing some uncertainty represented by $\overline{\Sigma}_0^a$. The bar over any Σ denotes that this matrix is related to the relevant covariance matrix via $\Sigma = \overline{\Sigma}\,\overline{\Sigma}^\intercal$. This decomposition allows for subsequent chance constraints to be expressed as closed-form convex constraints. Since we are interested in allowing the base policy to have as much freedom as possible, we avoid the additional conservatism that would result from incorporating uncertainty over future actions and allow these to be chosen deterministically.

OBJECTIVE. The safety guide objective minimizes the divergence between the base policy action distribution and the distribution of the MPC's first action. If the base policy distribution allows for subsequent actions that maintain safety, the objective vanishes and the returned safe distribution is the original distribution specified by the base policy. KL divergence is not symmetric; we choose this argument order to make the objective convex in the variables μ_0^a and $\overline{\Sigma}_0^a$. To see this, consider the following form for the KL divergence, dropping references to s' for notational convenience:

$$
\begin{aligned}
& \mathrm{KL}\left(\mathcal{N}\big(\mu_0^a, \overline{\Sigma}_0^a \overline{\Sigma}_0^{a\intercal}\big) \parallel \mathcal{N}\big(\mu_\theta^a, \Sigma_\theta^a\big)\right) \\
= \;& \log\det \Sigma_\theta^a - \log\det \overline{\Sigma}_0^a \overline{\Sigma}_0^{a\intercal} - n + \mathrm{Tr}\big((\Sigma_\theta^a)^{-1} \overline{\Sigma}_0^a \overline{\Sigma}_0^{a\intercal}\big) + (\mu_\theta^a - \mu_0^a)^\intercal \big(\Sigma_\theta^a\big)^{-1} (\mu_\theta^a - \mu_0^a).
\end{aligned}
$$

Recall that symbols subscripted by θ are constants in the optimization, while symbols subscripted by 0 are optimization variables. Therefore we disregard the constant terms $\log\det \Sigma_\theta^a$ and $-n$. Convexity of $-\log\det \overline{\Sigma}_0^a \overline{\Sigma}_0^{a\intercal}$ follows from multiplicative properties of the determinant and concavity of the $\log\det$ operator. For the remaining terms, we assume that Σ_θ^a is positive definite, a practically satisfied assumption. The fourth term $\mathrm{Tr}\big((\Sigma_\theta^a)^{-1} \overline{\Sigma}_0^a \overline{\Sigma}_0^{a\intercal}\big)$ can then be rewritten as $\mathrm{Tr}(XX^T)$ with

$X = \sqrt{(\Sigma_\theta^a)^{-1}}\,\overline{\Sigma}_0^a$, which is a convex function composed with a linear function and is therefore convex. Finally, the last term is a positive definite quadratic form and is therefore convex in μ_0^a.

DYNAMICS. The state propagation equations follow from known properties of linear transformations of Gaussian random variables (Liu, 2019). Since actions after index 0 are entirely deterministic, we can express state uncertainty at future time steps $\overline{\Sigma}_t^s$ directly as linear functions of the initial action uncertainty $\overline{\Sigma}_0^a$. This parallels results in the chance-constrained path planning literature (Blackmore et al., 2011).

SAFETY CONSTRAINTS. The safety constraints for $\mathcal{S}$ and $\mathcal{S}_T$ can be handled similarly. Consider the chance constraint $\Pr\left[s \notin \mathcal{S}\right] < \epsilon$, with $s \sim \mathcal{N}(\mu_t^s, \overline{\Sigma}_t^s \overline{\Sigma}_t^{s\intercal})$. Evaluating such a constraint using sampling would require prohibitively many samples for small ϵ and result in a nonconvex optimization problem. We instead leverage techniques from chance constrained optimization to represent this constraint deterministically. Let the polyhedral safe set be defined by r linear inequalities as

$$\mathcal{S} = \bigcap_{i=1}^{r} \{s \mid u_i^\intercal s \leq v_i\}.$$

Deriving a tight closed-form expression for a joint constraint over multiple linear inequalities is a nontrivial problem that is typically handled by an approximation scheme (Cheng and Lisser, 2012). We conservatively bound the probability of violating each inequality by ϵ/r, noting that this implies

$$\Pr\ [s \notin S] \leq \sum_{i=1}^{r} \Pr\ [u_i^\intercal s > v_i] \leq \sum_{i=1}^{r} \frac{\epsilon}{r} = \epsilon.$$

We now aim to derive a closed-form counterpart for r constraints of the form

$$\Pr[u_i^T s > v_i] \leq \frac{\epsilon}{r}, \qquad s \sim \mathcal{N}(\mu_t^s, \overline{\Sigma}_t^s \overline{\Sigma}_t^{s\intercal}).$$

Since s is normally distributed, this constraint is equivalent to the deterministic constraint

$$v_i - \mu_t^{s\intercal} u_i \geq \Phi^{-1}\left(1 - \frac{\epsilon}{r}\right) \|\overline{\Sigma}_t^s u_i\|_2,$$

where Φ is the standard Gaussian CDF (Duchi, 2021). Each of our r constraints now becomes a second-order cone constraint and can be handled by conventional convex optimization solvers. If the original problem is infeasible, we relax these constraints with slack variables which we linearly penalize in the objective.

3.2. Policy Gradient with Safety Penalty

We now modify the standard policy gradient formulation (1) to include a term penalizing corrections by the safety guide, effectively training the base policy to behave safely. Our objective becomes

$$\max_\theta J^p(\pi_\theta), \qquad J^p(\pi_\theta) = \mathop{\mathbb{E}}_{(s,a)\sim\pi_\theta^{\text{safe}}}\Big[r(s,a) - \beta\, d\big(\pi_\theta^{\text{safe}}(\,\cdot \mid s), \pi_\theta(\,\cdot \mid s)\big)\Big], \tag{4}$$

where d is a positive definite statistical distance which is continuous in s for $\pi_\theta, \pi_\theta^{\text{safe}} \in \Pi$ and $\beta > 0$ is a regularization parameter. For notational convenience, the expectation draws from the stationary

state distribution induced by π_θ^{safe} and the associated action distribution. We show in Section 4.2 that any positive definite, continuous d results in a safe base policy after training. We choose the squared l_2 parameter distance for its numerical properties:

$$d\big(\pi_\theta^{\text{safe}}(\,\cdot \mid s), \pi_\theta(\,\cdot \mid s)\big) := \|\mu^a_{\text{safe}} - \mu^a_\theta\|_2^2 + \|\Sigma^a_{\text{safe}} - \Sigma^a_\theta\|_2^2.$$

We can now obtain our optimal parameters θ^* using gradient ascent on a Monte Carlo estimator similar to (2) with an added term for the safety penalty.

4. Theoretical Analysis

We show that our policy leads to safe exploration at training time with arbitrarily high probability. We then prove that coupling reward maximization with a safety penalty in (4) leads to a safe optimal base policy. This is highly desirable as it eliminates conflict between the base policy and the safety guide, mitigates distributional shift, and reduces the computational burden on the agent at test time.

4.1. Training Time Safety

Consider a standard episodic training setting where an episode terminates after a set number of time steps or upon violation of the state safety constraints.

Proposition 1 *Consider an arbitrary natural number T and safety tolerance $\epsilon > 0$ from Algorithm 1. Then over T training steps, the expected number of states s_t such that $s_t \notin \mathcal{S}$ is at most ϵT.*

This follows directly from the constraints on the optimization problem in Algorithm 1. Specifically, there is an ϵ chance of sampling an action from the safe distribution that leads to an unsafe state, in which case the episode ends in at most H time steps. Assumption 1 guarantees that with probability $1 - \epsilon$ the action sampled will be safe and subsequent optimizations will remain feasible.

Since ϵ is a design parameter, this expectation can be driven to be arbitrarily small, at the cost of imposing additional conservatism in the exploration process. In practice, this quantity can be effectively set to zero by a small concession on the size of the safe sets $\mathcal{S}$ and $\mathcal{S}_T$. Shrinking these by some factor $1 - \delta$ gives the safe policy a buffer to the true unsafe region, allowing it to recover from unsafe actions by softening the chance inequality constraints in Algorithm 1. Our experiments in Section 5 use this technique to maintain *perfect safety* over the course of a million training steps.

4.2. Base Policy Safety

In order to derive theoretical guarantees for the optimal policy of (4), we introduce two assumptions.

Assumption 2 *The parameterized base policy class π_θ is a universal approximator. Namely, for every policy $\pi^* \in \Pi$ and desired ϵ, there exists a parameterized $\pi_\theta \in \Pi$ such that*

$$\sup_{s,a} |\pi^*(a \mid s) - \pi_\theta(a \mid s)| < \epsilon.$$

Assumption 3 *The reward $r(\tau)$ is bounded over all trajectories τ.*

Assumption 2 parallels a standard assumption in the deep learning literature that a richly parameterized network is arbitrarily expressive. Assumption 3 is similarly benign, and is immediately satisfied in a typical setting where rewards are bounded and trajectories are finite.

Lemma 2 *For every $\pi^* \in \Pi$ and $\epsilon_J > 0$, there exists a learned parameterization π_θ such that*

$$J(\pi^*) - J(\pi_\theta) < \epsilon_J,$$

where $J(\pi) = \mathbb{E}_{\tau\sim\pi}\, r(\tau)$ is the standard reinforcement learning objective.

Proof Let $p(s)$ be the initial state distribution and $p(s_{t+1} \mid s_t, a_t)$ represent the environment transition dynamics. For notational simplicity, we define $\Delta J := J(\pi^*) - J(\pi_\theta)$. Then we can write

$$\Delta J = \int r(\tau) p(s_0) \delta\pi(\tau) \prod_{t=0}^{M} p(s_{t+1} \mid s_t, a_t) d\tau, \tag{5}$$

where

$$\delta\pi(\tau) = \prod_{t=0}^{M} \pi^*(a_t \mid s_t) - \prod_{t=0}^{M} \pi_\theta(a_t \mid s_t).$$

Assumption 2 implies that there exists a parameter vector θ such that $\delta\pi(\tau)$ can be bounded for all τ by an arbitrarily small quantity. Since $r(\tau)$ in (5) is bounded by Assumption 3 and probability distributions integrate to 1, ΔJ can be driven arbitrarily close to zero. ■

Lemma 2 relates the universal approximation properties from Assumption 2 to the reward incurred by the policy. We now proceed with the main theoretical result.

Theorem 3 *An optimal parameter vector θ^* which maximizes (4) is such that the base policy $\pi_{\theta*}$ is safe; i.e., the equality $\pi_{\theta*}(\,\cdot \mid s_t) = \pi^{\text{safe}}_{\theta^*}(\,\cdot \mid s_t)$ holds except on a set of measure zero with respect to the stationary state density function induced by $\pi_{\theta*}$ in a Radon-Nikodym sense.*

Proof To prove by contradiction, assume that $\pi_{\theta*}$ is not safe. Define the set of states where the policy diverges from its safe representation as

$$U = \{s : d(\pi^{\text{safe}}_{\theta^*}(\,\cdot \mid s), \pi_{\theta*}(\,\cdot \mid s)) > 0\} = \{s : d(s) > 0\}.$$

with some abuse of notation. Now, consider the measure μ^* induced by the stationary state density function of $\pi_{\theta*}$. Since probability distributions integrate to one, μ^* is finite on compact sets; Euclidian space is also locally compact Hausdorff and second countable, and hence we have μ^* regular (Theorem 7.8 in Folland (1999)).

Since d is continuous in s by assumption, U is the inverse image of an open set under a continuous function and is therefore open. Regularity of μ^* and $\mu^*(U) > 0$ (by assumption) implies that there exists a compact set $\bar{U} \subset U$ such that $\mu^*(\bar{U}) > 0$. Since continuous functions attain their minimum over compact sets, we have that $d(s) > \delta$ for all $s \in \bar{U}$ for some $\delta > 0$.

We now show that the difference in objectives between the safe and base policies is given by

$$J^p(\pi^{\text{safe}}_{\theta^*}) - J^p(\pi_{\theta*}) \geq \beta\delta\mu^*(\bar{U}) > 0.$$

Observe that the state action marginal in the expectation (4) is always taken with respect to $\pi^{\text{safe}}_{\theta^*}$; therefore, the reward terms vanish and the safety penalty is the only remaining term. By the previous discussion, this is at least $\delta\mu^*(\bar{U})$, providing the desired expression.

Finally, we invoke Lemma 2 to construct a policy $\pi_{\theta'} \in \Pi$ such that $J^p(\pi^{\text{safe}}_{\theta^*}) - J^p(\pi_{\theta'}) < \beta\delta\mu^*(\bar{U})/2$, noting that the safety penalty in (4) can be driven arbitrarily close to zero by Assumption 2 and continuity of d. This implies $J^p(\pi_{\theta'}) > J^p(\pi_{\theta*})$, which is a contradiction. ■

Theorem 3 shows that the optimal parameters θ^* for our objective (4) produce a *safe base policy* $\pi_{\theta*}$. Provided that gradient ascent effectively maximizes (4), we can be confident that the policy has learned to behave safely and no longer requires the safety guide. This has three key advantages.

1. *Harmony between the base policy and safety guide.* Without a safety penalty, there is limited incentive for the base policy to learn to correct its own unsafe actions; the executed actions and ensuing rewards are always drawn from the action distribution of the safety guide. As noted in Koller et al. (2018), this decoupling can lead to a perpetual conflict between the base policy and the safety guide, with the base policy constantly approaching the boundaries of the safe set and the guide constantly correcting. Theorem 3 shows that our method resolves this issue.

2. *Mitigation of distributional shift.* One potential concern with this method involves distributional shift; our policy gradient step updates the base policy, while rewards are sampled using the safe policy. Theorem 3 implies that as training progresses, the distributional shift between these two policies decays to zero.

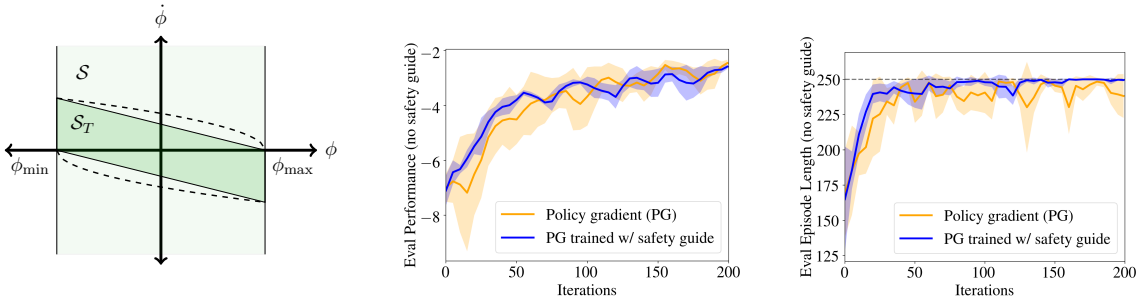
3. *Reduction of computational burden.* Solving the safety guide optimization problem requires significant computational effort. Theorem 3 shows that the safety guide can be removed at test time without compromising safety. This can free up agent resources for other tasks.

We note that in the setting where $J^p(\pi_\theta)$ is not completely maximized, the safety penalty $d(s)$ can still be concretely evaluated in any region of the state space. This provides the designer of the system with a quantitative measure of the level of safety of the base policy as well as insights into which regions of the state space are most dangerous.

5. Numerical Experiments

Consider a two-dimensional quadrotor with state $s_t = [x_t, \dot{x}_t, y_t, \dot{y}_t, \phi_t, \dot{\phi}_t]$, where (x, y) is the quadrotor position and ϕ is the counter-clockwise angle to the vertical. The episode terminates if the quadrotor hits the ground or tilts more than 0.5 radians. For early termination, the reward penalizes impact speed for hitting the ground ($r(s_t) = -1 - 2|\dot{y}_t|$) or rotational speed for excessive tilt ($r(s_t) = -1 - 5|\dot{\phi}_t|$). Otherwise, the quadrotor is incentivized to hover close to the ground while remaining centered horizontally ($r(s_t) = -0.01y_t - 0.01|x_t|$). The control inputs are $a_t = [f_t, \tau_t]$, with $f_t \in [-2, 2]$ the vertical thrust and $\tau_t \in [-2, 2]$ the torque. Using the time step $\Delta t = 0.02$, we simulate the system using the following linearized dynamics about the hovering equilibrium

$$\begin{bmatrix} x_{t+1} \\ \dot{x}_{t+1} \\ y_{t+1} \\ \dot{y}_{t+1} \\ \phi_{t+1} \\ \dot{\phi}_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & -g\Delta t & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ \dot{x}_t \\ y_t \\ \dot{y}_t \\ \phi_t \\ \dot{\phi}_t \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \Delta t/m & 0 \\ 0 & \Delta t/I \end{bmatrix} \begin{bmatrix} f_t \\ \tau_t \end{bmatrix}$$



(*a*) (*b*) (*c*)

Figure 2: Experimental setup and results for the quadrotor setting. (a) The ϕ-$\dot{\phi}$ plane of the quadrotor system safety sets. The dashed lines represent the true bounds of the terminal safety set $\mathcal{S}_T$; we inner approximate this by a polytope. In practice, we also slightly shrink $\phi_{\min}$ and $\phi_{\max}$ by some factor $1-\delta$. (b) Test-time performances of a policy gradient agent trained with and without the safety guide on the double integrator task. The thick line indicates mean performance over five runs, with the shaded area representing the standard deviation. (c) Test-time average episode length. The policy trained with the safety guide achieves the maximum episode length of 250 even when the safety guide is removed, indicating that the base policy has learned to behave safely.

for mass $m = 1$, inertia $I = 1$, and gravity $g = 1$. We design our safety set $\mathcal{S}$ to have the constraints $y \geq 0.05$ and $-0.45 \leq \phi \leq 0.45$. Our terminal safe set $\mathcal{S}_T$ consists of the same position bounds as well as a position-dependent velocity bound that captures the maximum velocity that can be brought to zero by the end of the corresponding safe set interval. Since this curve scales with the square root of distance, we inner approximate this by a polytope (Figure 2a). The safety tolerance, planning horizon, and safety penalty are set as $\epsilon = 0.01$, $H = 15$, and $\beta = 1.5$. Our network consists of two hidden layers of size 64 with $\tanh$ nonlinearities. We collected 5000 steps per batch with an episode length of 250 steps. The learning rate is 0.002 and discount factor is $\gamma = 0.95$. Our reported results include the top 5 of 10 seeds by average eval performance—a common approach for mitigating policy initialization variance (Wu et al., 2017). The safety guide optimization problem is solved using MOSEK (ApS, 2021).

Our training approach achieved perfect safety over a training corpus of a million steps without compromising performance (Figure 2b). Furthermore, Figure 2c shows that safety guide-trained policy rapidly achieves the optimal average episode length of 250 steps even when the safety guide is removed. This suggests that the safety penalty effectively induces the base policy to behave safely without having to try unsafe actions.

6. Conclusion

This work addresses the challenge of safe RL using a novel approach that combines a policy gradient agent with a chance-constrained MPC safety guide. The safety guide receives as input the proposed action distribution from the base policy and imposes additional safety requirements. By design, the safety guide intervenes minimally and modifies the base policy's proposed action distribution only if it inevitably leads towards an unsafe region of the state space. An additional safety penalty on these corrections in the overall objective allows us to provide theoretical guarantees that our base policy learns to behave safely without having to explore unsafe actions. We empirically justify our proposed method through numerical experiments on a linearized quadrotor control task.

References

Joshua Achiam and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. 2019.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. *CoRR*, abs/1705.10528, 2017.

MOSEK ApS. *The MOSEK optimization toolbox for Python manual. Version 9.0.*, 2021.

Felix Berkenkamp, Matteo Turchetta, Angela P. Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees, 2017.

Lars Blackmore, Masahiro Ono, and Brian C Williams. Chance-constrained optimal path planning with obstacles. *IEEE Transactions on Robotics*, 27(6):1080–1094, 2011.

J. M. Bravo, D. Limon, T. Alamo, and E. F. Camacho. On the computation of invariant sets for constrained nonlinear systems: An interval arithmetic approach. In *2003 European Control Conference (ECC)*, pages 288–293, 2003. doi: 10.23919/ECC.2003.7084969.

Jianqiang Cheng and Abdel Lisser. A second-order cone programming approach for linear programs with joint probabilistic constraints. *Operations Research Letters*, 40(5):325–328, 2012.

Richard Cheng, Gábor Orosz, Richard Murray, and Joel Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:3387–3395, 07 2019. doi: 10.1609/aaai.v33i01.33013387.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces, 2018.

John Duchi. Lecture notes in ee364a: convex optimization 1, 2021.

Gerald B Folland. *Real analysis: modern techniques and their applications*, volume 40. John Wiley & Sons, 1999.

E.G. Gilbert and K.T. Tan. Linear systems with state and control constraints: the theory and application of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36(9): 1008–1020, 1991. doi: 10.1109/9.83532.

Torsten Koller, Felix Berkenkamp, Matteo Turchetta, and Andreas Krause. Learning-based model predictive control for safe exploration, 2018.

Milan Korda, Didier Henrion, and Colin N. Jones. Convex computation of the maximum controlled invariant set for polynomial control systems, 2013.

Li-Ping Liu. Linear transformation of multivariate normal distribution: marginal, joint and posterior. 2019.

Kunal Menda, K. Driggs-Campbell, and Mykel J. Kochenderfer. Ensembledagger: A bayesian approach to safe imitation learning. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5041–5048, 2019.

V. Mnih, K. Kavukcuoglu, D. Silver, Andrei A. Rusu, J. Veness, Marc G. Bellemare, A. Graves, Martin A. Riedmiller, A. Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, A. Sadik, Ioannis Antonoglou, Helen King, D. Kumaran, Daan Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2008.02.003. Robotics and Neuroscience.

B. Pluymers, J.A. Rossiter, J.A.K. Suykens, and B. De Moor. The efficient computation of polyhedral invariant sets for linear systems with polytopic uncertainty. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 804–809 vol. 2, 2005. doi: 10.1109/ACC.2005.1470058.

S Joe Qin and Thomas A Badgwell. A survey of industrial model predictive control technology. *Control engineering practice*, 11(7):733–764, 2003.

J.B. Rawlings and D.Q. Mayne. *Model Predictive Control: Theory and Design*. Nob Hill Pub., 2009. ISBN 9780975937709.

Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík, editors, *AISTATS*, volume 15 of *JMLR Proceedings*, pages 627–635. JMLR.org, 2011.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm, 2017.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 1998. ISBN 0-262-19398-1.

Chen Tessler, Daniel J. Mankowitz, and Shie Mannor. Reward constrained policy optimization, 2018.

Kim P. Wabersich and Melanie N. Zeilinger. Linear model predictive safety certification for learning-based control, 2019.

Kim P. Wabersich and Melanie N. Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems, 2021.

Nolan Wagener, Byron Boots, and Ching-An Cheng. Safe reinforcement learning using advantage-based intervention. In *ICML*, 2021.

Ziyu Wang, Victor Bapst, Nicolas Heess, Volodymyr Mnih, Remi Munos, Koray Kavukcuoglu, and Nando de Freitas. Sample efficient actor-critic with experience replay, 2017.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *Advances in neural information processing systems*, 30, 2017.