GoSafe: Globally Optimal Safe Robot Learning

Dominik Baumann\textsuperscript{1,2}, Alonso Marco\textsuperscript{2}, Matteo Turchetta\textsuperscript{3}, and Sebastian Trimpe\textsuperscript{1,2}

\textbf{Abstract}—When learning policies for robotic systems from data, safety is a major concern, as violation of safety constraints may cause hardware damage. SAFE\textsubscript{OPT} is an efficient Bayesian optimization (BO) algorithm that can learn policies while guaranteeing safety with high probability. However, its search space is limited to an initially given safe region. We extend this method by exploring outside the initial safe area while still guaranteeing safety with high probability. This is achieved by learning a set of initial conditions from which we can recover safely using a learned backup controller in case of a potential failure. We derive conditions for guaranteed convergence to the global optimum and validate GOSAFE in hardware experiments.

\section{I. INTRODUCTION}

Algorithms that enable robotic systems to follow a trajectory or balance typically rely on a mathematical description of their dynamics through a model. Obtaining such a model is getting harder as robotic systems become more complex. To mitigate the need for a dynamics model, model-free machine learning methods aim to directly learn policies from data. However, such approaches require sufficient exploration, which can, due to the unknown dynamics, lead to failures, i.e., violation of safety constraints. In robotic systems, this should be avoided as it may cause hardware damage.

SAFE\textsubscript{OPT} \cite{1} is a model-free algorithm that, starting from a safe, albeit sub-optimal policy, explores new ones to improve the robot’s performance while avoiding failures with high probability. Crucially, it can only explore safe regions of the policy space connected to the initial policy. Thus, if multiple disjoint safe areas exist, SAFE\textsubscript{OPT} cannot detect them and may miss the globally optimal safe policy (see Fig. 1). Disjoint safe areas occur, e.g., in parameterized linear systems \cite{2} and local optima can also be encountered when learning gaits of a bipedal robot \cite{3}.

SAFE\textsubscript{OPT} approaches the policy search problem as a black-box optimization: it suggests a policy and receives information about its reward and safety without considering their generative process. However, when learning policies on dynamical systems like robots, we can monitor the evolution of the system’s state, and intervene in the experiment execution if there is an imminent danger. In this case, we can trigger a safe backup policy to ensure that all safety constraints are met. This allows us to evaluate potentially “unsafe” policies without violating any safety constraint. While SAFE\textsubscript{OPT} forgoes this possibility, GOSAFE exploits it to enable the exploration of disconnected safe regions in policy space, which, in turn, allows us to find globally optimal safe policies. To this end, we extend the definition of the safe set, which in the original SAFE\textsubscript{OPT} work is limited to the policy space, to a new set that joins policy and state space. The projection of this set onto the policy space determines a set of provably safe backup policies, while its projection onto the state space determines the corresponding states from which a backup policy should be triggered. Thus, this set enables us to search globally for the optimal policy while preserving safety.

\textbf{Contributions.} We propose GOSAFE, an extension of SAFE\textsubscript{OPT} that finds globally optimal safe policies for dynamical systems while fulfilling safety constraints at all times. Central to our approach is learning of a set of safe backup policies and states from which these should be triggered, i.e., we essentially learn a safely operable region, which is useful in its own right. We then leverage the obtained knowledge of this safely operable region to evaluate potentially unsafe policies and, therefore, discover better policies than those SAFE\textsubscript{OPT} is limited to. Under appropriate assumptions on the system dynamics, we can guarantee convergence to the global optimum while avoiding failures with high probability. The performance of GOSAFE is demonstrated in experiments on a Furuta pendulum\textsuperscript{1}.

\textbf{Related work.} Learning control policies directly on robotic hardware requires efficient algorithms as dataset collection is time-consuming and causes hardware deterioration. Bayesian optimization (BO) \cite{4} denotes a class of sample-efficient,
black-box optimization algorithms that have, among others, been successfully applied to policy search on hardware platforms [3], [5]–[7]. Based on BO, several algorithms have been proposed, which, apart from learning a control policy, also guarantee safety with high probability [1], [8]–[11]. As discussed above, those methods are restricted to an initially given safe region and may miss the global optimum in case of disjoint safe regions (cf. Fig. 1). Herein, we propose an extension of this line of work to also explore outside the initial safe area in hope of finding better optima, but without incurring in extra failures with high probability. The above methods learn exclusively in parameter space. Methods that also consider the state space, as we do to enable global exploration, have also been proposed. However, they are based on a system model (given or learned from data) [12]–[16]. In contrast, we present a model-free approach.

Other methods for learning optimal policies while avoiding failures have been proposed in [17] leveraging viability theory and in [18]–[20], which are based on reinforcement learning (RL). While [17] cannot guarantee safety during exploration, RL methods are generally difficult to apply to real systems due to their sample inefficiency. Another related line of work is Bayesian optimization under unknown constraints (BOC) [21]–[27], which seeks to optimize an objective function subject to multiple constraints. However, BOC assumes that failures come at no cost, thus incurring many failures during the search. Recently, [28] has proposed a middle-ground solution in which exploration outside the initial safe area is enabled, but only under a pre-established limited number of failures. Contrary to these works, we consider a setting where zero failures are allowed.

II. PROBLEM SETTING

Assume an unknown, Lipschitz-continuous system
\[ \dot{x}(t) = z(x(t), u(t)) \]  

where the control input \( u(t) \in \mathbb{R}^m \) is supposed to drive the system state \( x(t) \in \mathcal{X} \subset \mathbb{R}^d \) to some desired, possibly time-varying, state \( x_{\text{des}}(t) \) from a predefined initial state \( x(0) = x_0 \). The control inputs are computed according to a parameterized policy \( u(t) = \pi(x(t); a) \), where \( a \) are the policy parameters \( a \in \mathcal{A} \subseteq \mathbb{R}^d \). The quality of a policy \( a \) when applied from the initial condition \( x_0 \) is quantified through the unknown reward function \( f : \mathcal{A} \times \mathcal{X} \to \mathbb{R} \).

Our goal is to learn the optimal policy parameters \( a \) for a given initial condition \( x_0 \) while ensuring safety throughout the learning process. We assume that safety is encoded by a set of constraints on the current state of the system, e.g., on the distance to an obstacle. Such constraint functions are expressed through the immediate constraint function \( g_i : \mathcal{X} \to \mathbb{R} \), \( i \in \{1, \ldots, q\} \). The state trajectory is uniquely determined by \( (a, x_0) \) through (1). Thus, we further define \( g_i : \mathcal{A} \times \mathcal{X} \to \mathbb{R} \), \( i \in \{1, \ldots, q\} \) as \( g_i(a, x_0) = \min_{a \geq 0} \tilde{g}_i(x(t)) \), i.e., the minimum value observed throughout a trajectory starting from \( x_0 \) with parameters \( a \). Also \( \tilde{g}_i \) and \( g_i \) are unknown a priori but can be estimated through experiments on the real system.

The constrained optimization problem is
\[ \max_{a \in \mathcal{A}} f(a, x_0) \quad \text{subject to} \quad g_i(a, x_0) \geq 0 \quad \forall i \in \{1, \ldots, q\}, \]
whose constraints must be satisfied at each time step. Generally, we could also optimize over \( \mathcal{X} \) to find initial conditions that yield a higher reward. However, here we assume that the goal is to learn a policy from predefined initial conditions \( x_0 \), as is typically the case when letting a robot learn a certain behavior. To guarantee safety throughout the learning process, we assume that an initial set \( S_0 \) of safe but possibly sub-optimal parameters for the nominal \( x_0 \) are given. In robotics, such initial parameters can often be obtained from a simulation or domain knowledge.

The objective \( f \) and the constraints \( g_i \) are unknown a priori. For estimating them, we assume that both have a bounded norm in a reproducing kernel Hilbert space (RKHS) [29] induced by a kernel \( k \). For this class of functions, [30], [31] have shown that Gaussian processes (GPs) [32] can be used to obtain well-calibrated confidence intervals. Informally, a GP is a Bayesian non-parametric regression tool that places a probabilistic prior over the unknown functions \( f \) and \( g_i \). This allows us to provide confidence intervals and to make statements about their Lipschitz continuity.

III. PRELIMINARIES

In this section, we introduce the necessary background on GPs, BO, and the SAFEOPT framework.

Gaussian processes. A GP is a non-parametric model for a function \( f \), fully defined by a kernel and a mean function [32]. The model is updated with new observations of \( f \), which are assumed to be corrupted by white i.i.d. Gaussian noise, i.e., \( f(\zeta) = f(\zeta) + v, v \sim \mathcal{N}(0, \sigma^2) \). We can then express the posterior distribution conditioned on those observations in closed form, \( \mu_n(\zeta^*) = K_n(\zeta^*)(K_n + I_n \sigma^2)^{-1} f_n \) and \( \sigma_n^2(\zeta^*) = k(\zeta^*, \zeta^*) - k_n(\zeta^*)(K_n + I_n \sigma^2)^{-1} k_n^T(\zeta^*) \), respectively. Here, \( f_n \) is the vector of observed function values, the entry \( i,j \) of the covariance matrix \( K_n \in \mathbb{R}^{n \times n} \) is \( k(\zeta_i, \zeta_j) \), \( k_n(\zeta^*) = [k(\zeta^*, \zeta_1), \ldots, k(\zeta^*, \zeta_n)] \) represents the covariance between the current evaluation \( \zeta^* \) and the observed data points, and \( I_n \) the \( n \times n \) identity matrix.

This allows learning an approximation of the (scalar) reward function \( f \). To include the constraints \( g_i \), we extend the GP framework with a surrogate selector function as in [9],
\[ h(a, x_0, i) = \begin{cases} f(a, x_0) & \text{if } i = 0 \\ g_i(a, x_0) & \text{if } i \in I_g, \end{cases} \]

where \( i \in \mathcal{I} \) with \( \mathcal{I} = \{0, \ldots, q\} \) and \( I_g = \{1, \ldots, q\} \subset \mathcal{I} \) (i.e., the indices of the constraints) denotes whether the reward function or one of the constraints is returned. When indexed for a particular \( i \), the surrogate \( h(\cdot) \) is also a GP that can be used to predict expectations and uncertainties of the reward and constraint functions.

Bayesian optimization. BO adopts GP models to capture the belief about the objective function based on the available data. The GP model’s mean and variance are then used to suggest the next sample location in search of the objective function’s
optimum. One such algorithm is GP-upper confidence bound (GP-UCB) [30], which we will use in later sections, and which trades off exploration and exploitation by suggesting the next sample as 
\[ a_n = \arg\max_{a \in A} (\mu_n - 1(a) + \beta_n^{1/2} \sigma_{n-1}(a)), \]
where the scalar \( \beta_n \) is iteration-dependent and reflects the confidence interval of the GP. By iteratively evaluating the function at locations proposed by GP-UCB and updating the GP, GP-UCB provides theoretical bounds on the cumulative regret, which are ensured by choosing \( \beta_n \) appropriately at each iteration [30].

**SAFEOpt.** SAFEOpt [8] is a variant of GP-UCB that keeps track of a safe set, i.e., a set of parameters that satisfy safety constraints with high probability, given the GP model. After each experiment, the safe set is updated. During learning, SAFEOpt is trading off finding the optimum inside the currently known safe set with expanding the safe set. Existing applications of SAFEOpt to the policy search problem [9] only consider the parameter space \( A \) but ignore the state space \( X \). In parameter space, SAFEOpt can guarantee to find the optimum connected to an initial safe set \( S_0 \) (up to \( \epsilon \)-precision) while guaranteeing safety with high probability.

IV. **Globally Safe Optimization**

While SAFEOpt can provide guarantees on finding the optimum within an initial safe area, it cannot reach better optima that are not connected to this area. Thus, the quality of the solution strongly depends on its initialization. This is due to two main factors: (i) SAFEOpt models safety as a function of the policy parameters only, rather than policy parameters and initial conditions; (ii) SAFEOpt assumes that the violation of a given constraint is instantaneous and there is no possibility to intervene once a controller is chosen. In the following, we present GOSAFE, an extension of SAFEOpt that exploits the additional structure of the policy search problem in robotics to enable exploration outside the initial safe area. In particular, we extend the safe set by also including initial conditions and specifically search for initial conditions \( x(0) = \tilde{x}_0 \) from which we can still guarantee safety using some \( a \in A \). This allows for intervening during experiments and switching to a safe backup policy in case of potential constraint violation. We proceed by giving an intuition for GOSAFE, formally defining the new update rules, and stating its theoretical properties regarding safety and optimality. Finally, we discuss some practical considerations.

**A. Global Optimization Framework**

We start with an intuitive explanation of GOSAFE and then make the required concepts precise. GOSAFE proceeds in three stages (cf. Fig. 2) of increasing complexity. At each iteration, we check the stages’ conditions in the order that follows below and execute the first whose condition is satisfied.

**S1** Given the fixed nominal initial condition \( x_0 \), if we can expand the set of safe parameters or have a candidate for the optimal parameters, we do a SAFEOpt step in parameter space \( A \) as in [9].

**S2** If we can further expand the safe set by considering pairs of initial conditions \( \tilde{x}_0 \) and parameters \( a \) that are guaranteed to be safe, we do a safe exploration in this joint space.

**S3** Else, we further expand the safe set by sampling \( a \) from \( A \), i.e., by doing a global search in parameter space, and only sample \( \tilde{x}_0 \) from the safe set. Here, we may need to intervene, i.e., switch to a backup policy, during the experiment to preserve safety.

**S1** is the simplest as it involves searching over a reduced space (i.e., only the vertical axis in Fig. 2a) and only considers safe parameters. Its purpose is to find safe, optimal parameters for the nominal initial condition of the system. **S2** is more complex as we enlarge the search space to include the initial condition. This stage is needed to determine a set of backup policies and a set of initial conditions from which these backup policies can guarantee safety. As in Fig. 2b, this can reveal safe areas that were disconnected in parameter space and, thus, potentially better optima. **S3** is even more involved as the search space is enlarged further by considering potentially unsafe policy parameters \( a \in A \), which can be safely evaluated due to the backup policies estimated in S2. Note that if we executed **S3** before **S2**, we would need to interrupt many experiments as the safe set would be very conservative. The incentive of **S3** is to find disconnected safe areas (cf. Fig. 2c), which can then be further explored by revisiting **S1** and **S2**. That way, we thoroughly explore all
safe areas and, in the end, find the global optimum in Fig. 2d. In the following, we provide definitions of the required sets and detail the acquisition function for each of the three stages. The whole framework is summarized in Alg. 1.

**Set definitions.** For defining confidence intervals and the required sets, we extend the definitions from [9] to include the initial condition. Following [9], we consider a finite set, \( \mathcal{A} \), bounded \( \mathcal{X} \), and introduce a quantized version of \( \mathcal{X} \), \( \mathcal{X}_\mu \), such that \( \|x(t) - [x(t)]_\mu\|_1 \leq \mu \forall x \in \mathcal{X} \) and \([x(t)]_\mu \) the element in \( \mathcal{X}_\mu \) with the smallest \( L_1 \)-distance to \( x(t) \). Naturally, we then need that if \((a, x_0) \in S_0 \subseteq \mathcal{A} \times \mathcal{X}_\mu \) also all \( x \) for which \( \|x - x_0\|_1 \leq \mu \) are safe, i.e., we assume \( \forall (a, x_0) \in S_0 : g_i(a, x_0) > L_{x_0} \mu \forall i \in \mathcal{I}_g \). Further, we assume that \( f \) and \( g_i \) are Lipschitz-continuous with Lipschitz constants \( L_a \) and \( L_x \). We construct confidence intervals for the surrogate function (2) as \( Q_n(a, \tilde{x}_0, i) = [\mu_{n-1}(a, \tilde{x}_0, i) \pm \beta_n^{1/2} \sigma_{n-1}(a, \tilde{x}_0, i)] \). SAFEOPT requires that the safe set does not shrink. This can be achieved by defining the contained set as \( C_n(a, \tilde{x}_0, i) = C_{n-1}(a, \tilde{x}_0, i) \cap Q_n(a, \tilde{x}_0, i) \), where \( C_0(a, \tilde{x}_0, i) \) is \([L_x \mu, \infty) \cap \forall (a, x_0) \in S_0 \) and \( \mathbb{R} \) otherwise. Then, upper and lower bounds can be defined as \( l_n(a, \tilde{x}_0, i) = \min C_n(a, \tilde{x}_0, i) \) and \( u_n(a, \tilde{x}_0, i) = \max C_n(a, \tilde{x}_0, i) \). Finally, we update the safe set with

\[
S_n = \bigcap_{i \in \mathcal{I}_g (a, \tilde{x}_0)} \{ a' \in \mathcal{A}, \tilde{x}_0' \in \mathcal{X}_\mu \mid l_n(a, \tilde{x}_0, i) - a' || - L_{x} \| \tilde{x}_0 - \tilde{x}_0' \|_1 + \mu \geq 0 \} \cup S_{n-1}.
\]

As in [9], we define two subsets of \( S_n \) to find a trade-off between expanding the safe set and finding the optimal parameters inside the current safe set. These subsets contain parameters that are likely to improve the current estimate of the maximum (potential maximizers, \( M_n \)) or are likely to expand the safe set (potential expanders, \( G_n \)) and are formally defined as \( M_n := \{(a, x_0) \in S_n \mid u_n(a, x_0, 0) \geq \max_{(a', x_0=\tilde{x}_0) \in S_n} l_n(a', x_0, 0)\} \) and \( G_n := \{(a, \tilde{x}_0) \in S_n \mid c_n(a, \tilde{x}_0) > 0\} \), with \( c_n(a, \tilde{x}_0) = |\{(a', \tilde{x}_0') \in \mathcal{A} \times \mathcal{X}_\mu \setminus S_n \mid \exists i \in \mathcal{I}_g : u_n(a', \tilde{x}_0, i) - l_n(a, \tilde{x}_0, i) - L_{x} \| \tilde{x}_0 - \tilde{x}_0' \|_1 + \mu \geq 0\}| \). Here, \( c_n(a, \tilde{x}_0) \) denotes the number of \((a, \tilde{x}_0)\) pairs which are guaranteed to satisfy at least one constraint given an optimistic observation. Note that for \( M_n \), we consider \( x_0 \) instead of \( \tilde{x}_0 \), as we assume that the optimal policy parameters are to be found for the nominal \( x_0 \). For knowing when to intervene in an experiment, we further need to define the border of the safe set, \( \partial S_n := \{ \tilde{x}_0 \in \mathcal{X}_\mu \mid \exists a \in \mathcal{A} : (a, \tilde{x}_0) \in S_n \land (\exists x \in \mathcal{X} : \|x - x_0\|_1 < 2\mu \land \exists a \in \mathcal{A} : (a, [x]_\mu) \in S_n) \} \). Intuitively, it defines the set of discrete states for which at least one neighbour does not belong to the safe set. Lastly, we define a set \( \mathcal{E}_f \) containing those \((a, \tilde{x}_0)\) for which we needed to stop the experiment prematurely during S3.

**S1.** The original SAFEOPT algorithm is obtained by choosing the next sample location

\[
a_n = \arg\max_{(a, x_0=\tilde{x}_0) \in G_n \cup M_n} \max_{i \in \mathcal{I}} w_n(a, x_0, i),
\]

where \( w_n(a, x_0, i) = u_n(a, x_0, i) - l_n(a, x_0, i) \). This corresponds to S1, where we solely explore in parameter space while fixing the initial condition to \( x_0 \).

**S2.** Next, we jointly explore the parameter and the initial condition space, i.e., we consider \( \tilde{x}_0 \). However, we only seek to expand the safe set and not to find a new optimum. Thus, we only consider the set of expanders \( G_n \):

\[
(a_n, \tilde{x}_0) = \arg\max_{(a, x_0) \in G_n \in \mathcal{I}_g} \max_{i \in \mathcal{I}} w_n(a, x_0, i).
\]

**S3.** We then globally sample parameters \( a \in \mathcal{A} \) to find new safe areas, but still choose \( \tilde{x}_0 \) from \( S_n \),

\[
(a_n, \tilde{x}_0) = \arg\max_{(a, \tilde{x}_0) \notin \mathcal{E}_f \cup \mathcal{S}_n : a \in \mathcal{A} \land \exists a': (a', \tilde{x}_0) \in \mathcal{S}_n \in \mathcal{I}_g}
\]

During S3, we exclude pairs \((a, \tilde{x}_0)\) that we already now are safe or for which we needed to interrupt the experiment in previous iterations.

**Experiments.** The routine for carrying out an experiment is described in the “rollout” procedure in Alg. 1. During S3, we need to monitor the state’s evolution to intervene in case of potential constraint violation, i.e., in case \( x(t) \) hits the border of the safe set. If \( \exists t : [x(t)]_\mu \in \partial S_n \), we switch to a safe backup policy. That is, we choose parameters \( a' \) such that \( (a', [x(t)]_\mu) \in S_n \). Further, we add \((a, \tilde{x}_0)\) of the experiment to \( \mathcal{E}_f \). After successful experiments (i.e., experiments during which we did not need to intervene), we update the safe set

\[
S_n = S_{n-1} \cup \{a_n, \tilde{x}_0_n\}.
\]

**Revisiting parameters.** Prematurely stopping an experiment does not imply that the experiment would have failed. Thus, if the initial condition space of the safe set is increased, we check for all prematurely stopped experiments whether they would still be interrupted given the new safe set. If this is not the case for some experiments, we delete those \((a, \tilde{x}_0)\) from \( \mathcal{E}_f \). The acquisitions function (6) may then suggest revisiting these points during subsequent steps.

**B. Theoretical Guarantees**

In this section, we show that GOSAFE enjoys the same safety guarantees as SAFEOPT, while we state conditions under which we can recover the global optimum\(^2\).

**Theorem 1:** Assume that \( h(a, \tilde{x}_0, i) \) with \( i \in \mathcal{I}_g \) has a norm bounded by \( B \) in an RKHS and that the measurement noise is \( \sigma \)-sub Gaussian. Further, assume that \( S_0 \neq \emptyset \). Choose \( \beta_n^{1/2} = B + 4\sigma \sqrt{\gamma(n-1)\mathcal{I}_g + 1 + \ln(1/\delta)} \), where \( \gamma_n \) describes the information capacity associated with a kernel (see [9] for more details). Then, given the proposed algorithm, with probability at least \( 1 - \delta \), where \( \delta \in (0, 1) \), we have \( \forall n \geq 1, \forall \tilde{x}_0 \in \mathcal{I}_g : g_i(x_n(t)) \geq 0 \).

\(^2\)Corrections. Different from the ICRA 2021 paper, we here have a union with \( S_{n-1} \). This is to ensure that points with \( 0 \leq l_n(a_n, \tilde{x}_0, i) \leq L_{x_0} \mu \) are not removed from the safe set. Further, we do not set \( l_n(a_n, \tilde{x}_0, i) = \max \{L_{x_0} \mu, l_n(a_n, \tilde{x}_0, i)\} \) for points added during S3 anymore, correcting a minor error in the ICRA version.

\(^3\)Detailed proofs can be found in the appendix.
We further provide conditions under which we can find the global reachability operator is defined as \( \mathcal{S} \). For \( S \in \mathbb{R}^n \), we switch to a safe backup policy, and mark stop the experiment, switch to a safe backup policy, and mark defining \( R^v(S) \) as the repeated composition of \( R_v(S) \) with itself, we can further obtain its closure as \( \bar{R}_v(S) := \lim_{n \to \infty} R^v(S) \).

**Theorem 2:** Assuming the same as in Theorem 1, \( \text{GoSAFE} \) converges to the optimum within \( \bar{R}_v(S_0) \) with \( \epsilon \)-precision with probability at least \( 1 - \delta \).

**Proof:** During S1 and S2, the convergence of the safe set to the maximum safely reachable region connected to it with \( \epsilon \)-precision follows from [9]. If there are further disconnected safe points \( (a, \bar{x}_0) \in R_v(S_0) \setminus S_n \), those will eventually be explored during S3 since \( \mu_0 \) and \( \mathcal{A} \) are finite. The alternation of these steps guarantees convergence to \( R_v(S_0) \). Then, finding the optimum within \( R_v(S_0) \) with \( \epsilon \)-precision follows from [9].

**Corollary 1:** Let \( a^* = \arg \max_{a \in \mathcal{A}} f(a, x_0) \) subject to \( g(a, x_0) \geq 0 \). If we have \( [x_0 + \int_0^1 z(\tau, \pi(\tau); a^*) d\tau, \bar{a}(\tau)]_\mu \notin \partial R_v(S_n) \) for all \( t \), then \( (a^*, x_0) \in R_v(S_0) \) and \( \text{GoSAFE} \) is guaranteed to find an \( \epsilon \)-optimal solution.

### C. Practical Implementation

The guarantees stated above come at a cost: the safe set needs to be expanded as much as possible, resulting in many experiments. Further, we assume that we can continuously monitor the state. In this section, we discuss how this can be dealt with in practical applications.

**Reducing the search space.** The safe set \( S_n \) contains pairs of policy parameters and initial conditions. While we fix the initial condition to \( x_0 \) when searching for the optimum, we consider both \( a \) and \( \bar{x}_0 \) as free parameters when expanding in the joint initial condition and parameter space. Instead, one can also choose fixed policy parameters when expanding in initial condition space. The chosen parameters should then be likely to yield a large safe area, which could, for instance, be the parameters with the best performance obtained so far.

**Reducing the number of experiments.** Expanding the initial condition space of the safe set may be terminated earlier to reduce the number of experiments. Termination can be triggered by a stopping criterion that determines when the safe set is “big enough” or by executing S2 and S3 for a fixed amount of iterations. Then, one can fix the initial condition to \( x_0 \) and directly search for the global optimum using, e.g.,

\[ a_n = \arg \max_{a \in \mathcal{A}} \max_{x_0} w(n, a, x_0, i) \]  

**Sampling interval.** In practice, we only receive sampled measurements of \( x(t) \). Thus, we introduce a parameter \( \eta \) and interrupt experiments if \( \inf_{y \in \partial S_n} \| x(t) - y \|_1 < \eta \). The threshold must be chosen in accordance with the dynamics of the system and the sampling rate.

### V. EVALUATION

We evaluate GoSAFE on a real Furuta pendulum [33]. The Furuta pendulum is a fast and underactuated, unstable system and, therefore, a challenging platform for GoSAFE.

**Setting.** We consider stabilization of the pendulum using static state feedback \( u(t) = Ku(t) \), where the entries of \( K \) that multiply the pole angle \( (K_0) \) and the angle of the
rotary arm \((K_a)\) are learned in the range \([0, 1]\), while the rest are fixed. We model \(f\) and \(g\) with independent GPs with Matérn 3/2 kernels with length scale 0.01 and variance 5 for the parameter space, and 0.1 and 0.2 for the initial condition space. We choose \(\beta_n \equiv 3\) and constraints are given by \(|\alpha| < \pi/2\) and \(|\beta| < \pi/2\). Accordingly, \(g_1\) is defined as the minimum distance to either of these constraints during an experiment. The reward function in case of a successful experiment, \(R = \frac{1}{T} \sum_{t=0}^{T-1} \left(1 - 0.8|\theta(t)|+0.2|\alpha(t)|\right)\), is strictly positive. If the backup controller needs to kick in, we return a reward and constraint value of 0 and include both in the GP. During experiments, we receive measurements at a frequency of 50 Hz, i.e., considerably slower as the 1 kHz typically used in robotics. We switch to the backup controller whenever \(l_{an}(a, x(t), 1) < 1\). Our implementation builds upon the original SAFEOpt code\(^4\).

**Results.** We initialize the algorithm in the safe area on the right in Fig. 3a and then execute **S1** for 20 iterations. As highlighted in Fig. 3b, **S1** corresponds to standard SAFEOpt. Thus, the algorithm stays inside the initial safe region. While the controller found at the end of **S1** is the best SAFEOpt can find, GOSAFE can now progress to **S2** and **S3** to discover new safe areas, which naturally comes at the cost of more iterations. Thus, we next execute **S2** for 100 iterations. While the state space is of higher dimension, the acquisition function of **S2** is substantially easier than that of **S1**. Therefore, computing the next sample location is even faster in **S2** as compared to **S1**. Then, GOSAFE enters **S3**, where we globally search for new safe areas. We adopt the modifications proposed in Sec. IV-C. In particular, we use (8) to search for new safe areas for the nominal initial conditions only. Most of these experiments are unsafe and terminated early since the system gets too close to \(\partial S_n\). In such cases, the backup policy was triggered to keep the pole in balance. Eventually, a second safe area (left in Fig. 3a) is detected and subsequently expanded by revisiting **S1** (cf. Fig. 3b). That is, we execute SAFEOpt in the newly found safe region. However, the original SAFEOpt algorithm would not have been able to find a second safe area. In this second area, the algorithm finds a controller parameterization with a slightly better reward than the optimum in the initial safe area. Thus, GOSAFE reveals a better optimum than could have been found using SAFEOpt, as can also be confirmed over multiple runs, as shown in Fig. 3c, at the cost of the additional iterations in **S2** and **S3** (cf. Fig. 3b). During multiple runs of the whole algorithm, the pole of the Furuta pendulum never dropped. The results demonstrate that GOSAFE can find additional safe areas and, thus, find optima that the original SAFEOpt algorithm would miss, while still providing safety guarantees\(^5\).

In addition to the above, we also compared GOSAFE against expected improvement with constraints (EIC) \(^{22}\). This is a popular BOC algorithm that suggests data points where the objective is expected to improve the best observed reward so far while discouraging regions of potential constraint violation. We adopt the hyperparameters from GOSAFE, however, choose a length scale of 0.1 for the parameter space GP to enable faster exploration. As can be inferred from Fig. 3c, EIC finds a similar optimum as GOSAFE. While EIC, due to the increased length scale, reaches that optimum significantly faster than GOSAFE, needing only 28 and 18 iterations in two independent runs to find the best guesses shown in Fig. 3c, it incurs in a significant amount of failures (10 and 7, respectively) during exploration. This would cause severe hardware damage on a more fragile robotic system.

The comparison with EIC and SAFEOpt reveals that GOSAFE retains the best of both worlds: it finds the global optimum and guarantees safety during exploration.

**VI. CONCLUSIONS**

We proposed GOSAFE, an extension of the well-known SAFEOpt algorithm. Both algorithms aim at optimizing a policy while guaranteeing safety during exploration. By considering safety not only in parameter but also in state space, GOSAFE can, in contrast to SAFEOpt, explore globally,
switching to a safe backup policy in case of potential constraint violation. Experiments on a Furuta pendulum demonstrate that GoSAFE finds the global optimum also in case it is outside an initially given safe area, while SAFE-OPT remains stuck at the local optimum inside the initial region.

REFERENCES

[1] F. Berkenkamp, A. Schoellig, and A. Krause, “Safe controller optimization for quadrotors with Gaussian processes,” in IEEE International Conference on Robotics and Automation, 2016, pp. 491–496.

[2] E. N. Gryazina and B. T. Polyak, “Stability regions in the parameter space: D-decomposition revisited,” Automatica, vol. 42, no. 1, pp. 13–26, 2006.

[3] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, “Bayesian optimization for learning gaits under uncertainty,” Annals of Mathematics and Artificial Intelligence, vol. 76, no. 1-2, pp. 5–23, 2016.

[4] J. Mockus, V. Tiesis, and A. Zilinskas, “The application of Bayesian methods for seeking the extremum,” Towards Global Optimization, vol. 2, no. 117-129, p. 2, 1978.

[5] R. Antonova, A. Rai, and C. G. Atkeson, “Deep kernels for optimizing locomotion controllers,” arXiv preprint arXiv:1707.09062, 2017.

[6] A. Marco, P. Hennig, J. Bohg, S. Schaal, and S. Trimpe, “Automatic LQR tuning based on Gaussian process global optimization,” in IEEE International Conference on Robotics and Automation, 2016, pp. 270–277.

[7] M. Turchetta, A. Krause, and S. Trimpe, “Robust model-free reinforcement learning with multi-objective Bayesian optimization,” arXiv preprint arXiv:1901.13399, 2019.

[8] Y. Sui, A. Gotovos, J. Burdick, and A. Krause, “Safe exploration for optimization with Gaussian processes,” in International Conference on Machine Learning, 2015, pp. 997–1005.

[9] F. Berkenkamp, A. Krause, and A. Schoellig, “Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics,” arXiv preprint arXiv:1602.04450, 2016.

[10] J. Schreiter, D. Nguyen-Tuong, M. Eberts, B. Bischoff, H. Markert, and M. Toussaint, “Safe exploration for active learning with Gaussian processes,” in Joint European Conference on Machine Learning and Knowledge Discovery in Databases, 2015, pp. 133–149.

[11] M. Schillinger, B. Ortelt, B. Hartmann, J. Schreiter, M. Meister, D. Nguyen-Tuong, and O. Nelles, “Safe active learning of a high pressure fuel supply system,” in EUROSIM Congress on Modelling and Simulation, no. 142, 2018, pp. 286–292.

[12] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, “Safe model-based reinforcement learning with stability guarantees,” in Advances in Neural Information Processing Systems, 2017, pp. 908–918.

[13] F. Berkenkamp, R. Moriconi, A. P. Schoellig, and A. Krause, “Safe learning of regions of attraction for uncertain, nonlinear systems with Gaussian processes,” in IEEE Conference on Decision and Control, 2016, pp. 4661–4666.

[14] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin, “Reachability-based safe learning with Gaussian processes,” in IEEE Conference on Decision and Control, 2014, pp. 1424–1431.

[15] M. Turchetta, F. Berkenkamp, and A. Krause, “Safe exploration in finite Markov decision processes with Gaussian processes,” in Advances in Neural Information Processing Systems, 2016, pp. 4312–4320.

[16] M. Turchetta, F. Berkenkamp, and A. Krause, “Safe exploration for interactive machine learning,” in Advances in Neural Information Processing Systems, 2019, pp. 2891–2901.

[17] S. Heim, A. von Rohr, S. Trimpe, and A. Badri-Spröwitz, “A learnable safety measure,” in Conference on Robot Learning, 2019.

[18] J. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh, “A Lyapunov-based approach to safe reinforcement learning,” in Advances in Neural Information Processing Systems, 2018, pp. 8092–8101.

[19] J. Garcia and F. Fernández, “A comprehensive survey on safe reinforcement learning,” Journal of Machine Learning Research, vol. 16, no. 1, pp. 1437–1480, 2015.

[20] J. M. Hernández-Lobato, M. A. Gelbart, R. P. Adams, M. W. Hoffman, and Z. Ghahramani, “A general framework for constrained Bayesian optimization using information-based search,” The Journal of Machine Learning Research, vol. 17, no. 1, pp. 5549–5601, 2016.

[21] M. A. Gelbart, J. Snoek, and R. P. Adams, “Bayesian optimization with unknown constraints,” in Conference on Uncertainty in Artificial Intelligence, 2014, pp. 250–259.

[22] J. R. Gardner, M. J. Kusner, Z. E. Xu, K. Q. Weinberger, and J. P. Cunningham, “Bayesian optimization with inequality constraints,” in International Conference on Machine Learning, 2014, pp. 937–945.

[23] R. B. Gramacy and H. Lee, “Optimization under unknown constraints,” Bayesian Statistics 9, 2011.

[24] M. Schonlau, W. J. Welch, and D. R. Jones, “Global versus local search in constrained optimization of computer models,” Lecture Notes-Monograph Series, pp. 11–25, 1998.

[25] V. Picheny, “A stepwise uncertainty reduction approach to constrained global optimization,” in International Conference on Artificial Intelligence and Statistics, 2014, pp. 787–795.

[26] A. Marco, D. Baumann, M. Khadiv, P. Hennig, L. Righetti, and S. Trimpe, “Robot learning with crash constraints,” IEEE Robotics and Automation Letters, 2021.

[27] A. Marco, A. von Rohr, D. Baumann, J. M. Hernández-Lobato, and S. Trimpe, “Excursion search for constrained Bayesian optimization under a limited budget of failures,” arXiv preprint arXiv:2005.07443, 2020.
we denote by \( g_i \) the constraint function \( i \), and \( \hat{g}_i \) the immediate constraint function \( i \). By Lemma 14, we have that the true functions are contained in the safe set \( S_n \). Intuitively, we assume that all those points are safe with high probability and that for all pairs we have policy parameters that ensure that also all continuous states \( x \) that are \( \mu \)-close to \( \hat{x}_0 \) are safe.

**Proof:** Let \( S_n \neq \emptyset \). The following properties hold for all \( i \in \mathcal{T}_g \) with probability at least 1 - \( \delta \):

(i) \( \forall (a, \hat{x}_0) \in S_n : g_i(a, \hat{x}_0) \geq 0 \);

(ii) \( \forall (a, \hat{x}_0) \in S_n : \exists a' \in \mathcal{A} : (a', \hat{x}_0) \in S_n \land g_i(a', \hat{x}_0) \geq L_x \mu (a \text{ and } a' \text{ may be identical}); \)

(iii) \( \exists x_0 \in \partial S_n, a \in \mathcal{A} : (a, x_0) \in S_n \land g_i(a, x_0) \geq L_x \mu \).

For the initial \( S_0 \), this assumption is satisfied since we have \( \forall (a, \hat{x}_0) \in S_0 : g_i(a, \hat{x}_0) \geq L_x \mu \) for all \( i \in \mathcal{T}_g \). We first show that under Assumption 2, we can guarantee safety during the global search in \( S_3 \). We then show that the update rules guarantee that Assumption 2 is satisfied for all \( n \) given \( S_0 \).

During the global search, we may evaluate both safe and unsafe policy parameters. To guarantee safety, we thus need to show that (i) before leaving the safe set during an experiment, we trigger a safe backup policy, (ii) if we do not trigger a backup policy, the evaluated pair \( (a, \hat{x}_0) \) is safe with high probability, (iii) triggering a backup policy guarantees constraint satisfaction with high probability. We start by proving that before leaving the safe set, we necessarily visit a state that belongs to the border \( \partial S_n \) of the safe set.

**Lemma 2:** Consider \( \hat{x}_0 \in \mathcal{X}_\mu \) such that \( \exists a' \in \mathcal{A} : (a', \hat{x}_0) \in S_n \). Let \( x(t) \) denote the state of the system at time \( t \) for the trajectory \( \xi_{(t_0, \hat{x}_0, a)} \) for all \( t \geq t_0 \geq 0 \) and with \( a \in \mathcal{A} \). Define \( T_{\text{unsafe}} := \{ t \in [t_0, \infty) : \exists \hat{a} \in \mathcal{A} \text{ such that } (\hat{a}, [x(t)]_\mu) \in S_n \} \). Assume \( T_{\text{unsafe}} \neq \emptyset \) and let \( t_2 := \min(T_{\text{unsafe}}) \).

Then \( \exists t_1 \in [t_0, t_2) \) such that \( [x(t_1)]_\mu \in \partial S_n \) and \( \forall t' \in [t_0, t_1] \) \( \exists a' \in \mathcal{A} \) such that \( (a', [x(t')]_\mu) \in S_n \).

**Proof:** First, consider the case \( x(t_0) = \hat{x}_0 \in \partial S_n \).

Then, we have \( t' = t_0 \) and \( \exists a' \in \mathcal{A} : (a', \hat{x}_0) \in S_n \) by the choice of \( \hat{x}_0 \). Now, assume \( x(t_0) \notin \partial S_n \). Then, we have, by definition of \( \partial S_n \), \( \| x(t_0) - x(t_2) \|_1 > 2\mu \). By definition of \( t_2 \), we further have that

\[ \forall t' < t_2 : \exists a' \text{ such that } (a', [x(t')]_\mu) \in S_n. \]

As \( z \) is Lipschitz-continuous, there exists a unique solution to (1) and continuity of \( x(t) \) follows from the Picard-Lindelöf theorem. Thus, \( \exists t_1 < t_2 \) such that \( \| x(t_2) - x(t_1) \|_1 < \mu \). The discretization further implies that \( \| x(t_1) - [x(t_1)]_\mu \|_1 \leq \mu \).

It follows that

\[ \| x(t_2) - [x(t_1)]_\mu \|_1 \leq \| x(t_2) - x(t_1) \|_1 + \| x(t_1) - [x(t_1)]_\mu \|_1 \leq 2\mu. \]

### Appendix

#### Table I: Summary of main variables and operators.

| Symbol | Explanation |
|--------|-------------|
| \( t \) | time |
| \( z \) | System dynamics |
| \( x \) | System state |
| \( a \) | Policy parameters |
| \( f \) | Reward function |
| \( g_i \) | Constraint function \( i \) |
| \( \hat{g}_i \) | Immediate constraint function \( i \) |
| \( h_i \) | Surrogate selector function for index \( i \) |
| \( \mathcal{T} \) | Set of all indices |
| \( \mathcal{T}_g \) | Set of all indices pertaining to constraints (i.e., \( i > 0 \)) |
| \( \mathcal{A} \) | Parameter space |
| \( \mathcal{X} \) | Continuous state space |
| \( \mu_{\mathcal{X}} \) | Discretization parameter |
| \( \mu_{\mathcal{A}} \) | Discretized state space |

We first show that under Assumption 2, we can guarantee safety during the global search in \( S_3 \). We then show that the update rules guarantee that Assumption 2 is satisfied for all \( n \) given \( S_0 \).

During the global search, we may evaluate both safe and unsafe policy parameters. To guarantee safety, we thus need to show that (i) before leaving the safe set during an experiment, we trigger a safe backup policy, (ii) if we do not trigger a backup policy, the evaluated pair \( (a, \hat{x}_0) \) is safe with high probability, (iii) triggering a backup policy guarantees constraint satisfaction with high probability. We start by proving that before leaving the safe set, we necessarily visit a state that belongs to the border \( \partial S_n \) of the safe set.

**Lemma 2:** Consider \( \hat{x}_0 \in \mathcal{X}_\mu \) such that \( \exists a' \in \mathcal{A} : (a', \hat{x}_0) \in S_n \). Let \( x(t) \) denote the state of the system at time \( t \) for the trajectory \( \xi_{(t_0, \hat{x}_0, a)} \) for all \( t \geq t_0 \geq 0 \) and with \( a \in \mathcal{A} \). Define \( T_{\text{unsafe}} := \{ t \in [t_0, \infty) : \exists \hat{a} \in \mathcal{A} \text{ such that } (\hat{a}, [x(t)]_\mu) \in S_n \} \). Assume \( T_{\text{unsafe}} \neq \emptyset \) and let \( t_2 := \min(T_{\text{unsafe}}) \).

Then \( \exists t_1 \in [t_0, t_2) \) such that \( [x(t_1)]_\mu \in \partial S_n \) and \( \forall t' \in [t_0, t_1] \) \( \exists a' \in \mathcal{A} \) such that \( (a', [x(t')]_\mu) \in S_n \).

**Proof:** First, consider the case \( x(t_0) = \hat{x}_0 \in \partial S_n \).

Then, we have \( t' = t_0 \) and \( \exists a' \in \mathcal{A} : (a', \hat{x}_0) \in S_n \) by the choice of \( \hat{x}_0 \). Now, assume \( x(t_0) \notin \partial S_n \). Then, we have, by definition of \( \partial S_n \), \( \| x(t_0) - x(t_2) \|_1 > 2\mu \). By definition of \( t_2 \), we further have that

\[ \forall t' < t_2 : \exists a' \text{ such that } (a', [x(t')]_\mu) \in S_n. \]

As \( z \) is Lipschitz-continuous, there exists a unique solution to (1) and continuity of \( x(t) \) follows from the Picard-Lindelöf theorem. Thus, \( \exists t_1 < t_2 \) such that \( \| x(t_2) - x(t_1) \|_1 < \mu \). The discretization further implies that \( \| x(t_1) - [x(t_1)]_\mu \|_1 \leq \mu \).

It follows that

\[ \| x(t_2) - [x(t_1)]_\mu \|_1 \leq \| x(t_2) - x(t_1) \|_1 + \| x(t_1) - [x(t_1)]_\mu \|_1 \leq 2\mu. \]
Since \( t_1 < t_2 \), (10) implies that \( \exists a \in A \) such that \( (a^i, [x(t)]_\mu) \in S_n \). Taking this and (11) yields \( [x(t)]_\mu \in \partial S_n \). This together with (10) completes the proof.  

Thus, before leaving the safe set, we necessarily visit a state that belongs to \( \partial S_n \).

Next, we show that for experiments for whose entire duration we stay inside the safe set, the immediate constraint is satisfied with high probability. For this, we first establish that satisfying the constraint function implies satisfaction of the immediate constraint function.

**Lemma 3:** For any \( x \in X \), if there exists \( a \in A \) such that \( g_i(a, x) \geq 0 \forall i \in I_n \), then \( \bar{g}_i(x) \geq 0 \forall i \in I_n \).

**Proof:** By definition of the immediate constraint function, it holds that

\[
\bar{g}_i(x) \geq \min_{x' \in \xi(a, x)} \bar{g}_i(x') = g_i(a, x) \geq 0
\]

for all \( i \in I_n \).

This lets us conclude that during experiments for which we stay inside the safe set, the immediate constraint function is satisfied with high probability.

**Lemma 4:** For all \( x \in X \) such that \( \exists a \in A : (a, [x]_\mu) \in S_n \), we have with probability at least \( 1 - \delta \), \( \bar{g}_i(x) \geq 0 \) for all \( i \in I_n \).

**Proof:** By Assumption 2(ii), for all \( (a, [x]_\mu) \in S_n \), we have some \( a' \in A \) such that \( g_i(a', [x]_\mu) \geq \xi \lambda \mu \) for all \( i \in I_n \) with probability at least \( 1 - \delta \). Thus, we have for all \( i \in I_n \) that \( g_i(a', [x]_\mu) \geq 0 \) by Lipschitz-continuity and \( \bar{g}_i(x) \geq 0 \) by Lemma 3.

Lastly, we need to prove that switching to a safe backup policy when evaluating unsafe policy parameters guarantees safety.

**Lemma 5:** With probability at least \( 1 - \delta \), we have \( \bar{g}_i(x(t)) \geq 0 \forall i \in I_n \), \forall t > 0 during experiments in S3.

**Proof:** In S3, we start from \( \bar{x}_0 \) such that \( \exists a \in A : (a, [x(t)]_\mu) \in S_n \). For all \( t \) such that \( \exists a' : (a', [x(t)]_\mu) \in S_n \), \( \bar{g}_i(x(t)) \geq 0 \forall i \in I_n \) with probability at least \( 1 - \delta \) follows from Lemma 4. If \( \exists t' \) such that \( [x(t')]_\mu \in \partial S_n \), we apply \( a' \) such that \( (a', [x(t')]_\mu) \in S_n \), where \( g_i(a', [x(t')]_\mu) \geq \xi \lambda \mu \) by Assumption 2(iii). Then, we have for all \( i \in I_n \)

\[
0 \leq g_i(a', [x(t')]_\mu) \geq \xi \lambda \mu
\]

by Lipschitz continuity. Markov property of (1).  

This proves that the second part of the trajectory after applying \( a' \) is safe. Safety of the first part is guaranteed by Lemmas 2 and 4.

Combining the results presented so far then proves safety of S3.

**Lemma 6:** Assume a successful experiment of S3. For the evaluated \( (a, \bar{x}_0) \), we have that \( g_i(a, \bar{x}_0) \geq 0 \forall i \in I_n \) with probability at least \( 1 - \delta \).

**Proof:** For a successful experiment S3 we never have \( [x(t)]_\mu \in \partial S_n \). Therefore, we have by Lemma 2 and Assumption 2 that for all \( t, \exists \alpha' : (a', [x(t)]_\mu) \in S_n \land g_i(a', [x(t)]_\mu) \geq \xi \lambda \mu \) with probability at least \( 1 - \delta \). Thus, for each of these pairs \( (a', [x(t)]_\mu) \), we have for all \( i \in I_n \) with probability at least \( 1 - \delta \), \( g_i(a', x(t)) \geq 0 \) by Lipschitz-continuity.  

Thus, we have shown that during global exploration, we will not violate any safety constraints with high probability assuming a safe set \( S_n \) as above. We now show that updating the safe set preserves safety guarantees.

**Lemma 7:** The safe set \( S_n \) satisfies Assumption 2 for any \( n \geq 0 \).

**Proof:** We prove the lemma by induction. For the initial safe set at \( n = 0 \), we assumed that for all \( (a, \bar{x}_0) \in S_n \) also states that are \( \mu \)-close are safe, i.e., \( g_i(a, \bar{x}_0) \geq \xi \lambda \mu \forall i \in I_n \). Thus, for \( n = 0 \) the claim holds by assumption.

For the induction step, we distinguish between local and global search. For the local search, i.e., S1 and S2, the proof is a straightforward extension of [9, Lem. 11]. In particular, assume some \( n \geq 1 \) such that Assumption 2 holds. Then, if we do an update step S1 or S2, we have for all \( (a, \bar{x}_0) \in S_n \setminus S_{n-1} \), \( i \in I_n, \exists (a', \bar{x}'_0) \in S_{n-1} \) such that

\[
0 \leq t_n(a', \bar{x}'_0) - t_n(a - a') \leq L_\lambda (\|\bar{x}_0 - \bar{x}'_0\|_1 + \mu) \\
\leq g_i(a', \bar{x}'_0) - L_\lambda (\|\bar{x}_0 - \bar{x}'_0\|_1 + \mu)
\]

By Cor. 2\( \leq g_i(a, \bar{x}_0) - L_\lambda \mu \).  

By Lipschitz-continuity Then, all three parts of Assumption 2 are satisfied for \( (a, \bar{x}_0) \) pairs that are added in S1 or S2.

Now, for S3, again assume some \( n \geq 1 \) such that Assumption 2 holds for \( S_{n-1} \). Doing an update step S3, we evaluate a pair

\( (a, \bar{x}_0) \notin E_n \cup S_{n-1} : a \in A \land \exists a' : (a', \bar{x}_0) \in S_{n-1} \). (12)

For every successful experiment after which we update the safe set, we further have

\[
\forall x \in \xi_{(t, n, \bar{x}_0, a)} : x \notin \partial S_{n-1}.
\]

The safe set is, in case of a successful experiment, updated with \( S_n = S_{n-1} \cup (a, \bar{x}_0) \). For \( (a, \bar{x}_0) \), Lemma 6 guarantees \( g_i(a, \bar{x}_0) \geq 0 \forall i \in I_n \) with probability at least \( 1 - \delta \), i.e., Assumption 2(ii) is satisfied. Further, due to (12) Assumption 2(ii) was already satisfied for the considered \( \bar{x}_0 \) in \( S_{n-1} \). Finally, (13) guarantees \( \bar{x}_0 \notin \partial S_{n-1} \). Therefore, the update will not violate Assumption 2(iii).

To conclude the proof, we still need to prove safety during S1 and S2.

**Lemma 8:** Let \( S_n \subseteq A \times X_\mu \). Then, with probability at least \( 1 - \delta \), \( g_i(x) \geq 0 \) for all \( i \in I_n, x \in \xi_{(t, n, \bar{x}_0, a)} \), with \( (a, \bar{x}_0) \in S_n \), for all \( t_0 \geq 0 \).

**Proof:** Assume to the contrary that \( \exists x \in \xi_{(t, n, \bar{x}_0, a)} \), with \( (a, \bar{x}_0) \in S_n \), such that \( \bar{g}_i(x) < 0 \). Then

\[
0 \geq \bar{g}_i(x) \geq \min_{x' \in \xi_{(t, n, \bar{x}_0, a)}} \bar{g}_i(x')
\]

Since \( x \in \xi_{(t, n, \bar{x}_0, a)} \)

\[
= \min_{x' \in \xi_{(t, \bar{x}_0, a)}} \bar{g}_i(x')
\]

Markov property of (1)

\[
= g_i(a, \bar{x}_0).
\]

By def.
This is a contradiction.

Taking Lemma 5 and Lemma 7 guarantees safety during S3 while Lemma 8 guarantees safety during S1 and S2. Combining these results then proves the theorem.

**B. Extended Proof of Theorem 2 and Corollary 1**

Having shown that GoSAFE provides safety guarantees, we now discuss optimality. In particular, we will show that GoSAFE converges, with ε-precision and probability at least $1 - \delta$, to the optimum within the maximum safety reachable set $\bar{R}_c$. To prove ε-convergence to the optimum within $\bar{R}_c$, we first need to establish that we sufficiently explore $R_c$. We start by establishing that the safe set does not shrink.

**Lemma 9:** For any $n \geq 1$ it holds that $S_{n+1} \supseteq S_n \supseteq S_0$.

**Proof:** For the sake of contradiction, assume the opposite is true. That is, assume $\exists x \in \partial R$ such that (i) $\exists a \in A : (x, a) \in S \subseteq R$ and (ii) $x \notin \partial S$. However, from (i) and (ii) it follows by definition that $\forall x' \in X$ such that $\|x - x'\|_1 < 2\mu \exists a' \in A : ([x']_{a'}, a') \in S \subseteq R$. Thus, $x \notin \partial R$, which is a contradiction.

This lets us conclude that safe experiments during $S$, i.e., experiments during which we did not hit $\partial S$, will also be successful under $R$.

**Lemma 10:** If $S \subseteq R$, then $\partial R \subseteq \partial S \cup \{x \in \mathcal{X}_\mu \text{ such that } \exists a \in A : (x, a) \in S\}$.

**Proof:** For the sake of contradiction, assume the opposite is true. That is, assume $\exists x \in \partial R$ such that $\exists a \in A : (x, a) \in S \subseteq R \cap \partial S$, and $x \notin \partial S$. However, from (i) and (ii) it follows by definition that $\forall x' \in X$ such that $\|x - x'\|_1 < 2\mu \exists a' \in A : ([x']_{a'}, a') \in S \subseteq R$. Thus, $x \notin \partial R$, which is a contradiction.

This lets us conclude that safe experiments during $S$, i.e., experiments during which we did not hit $\partial S$, will also be successful under $R$.

**Lemma 11:** If $S \subseteq R$ and $\exists x \in \mathcal{X}_{0, x_0, a}$ such that $\exists a \in A : (x_0, a') \in S$, then $\exists a \in A : (x_0, a) \in S$.

**Proof:** For the sake of contradiction, assume the opposite is true. That is, assume $\exists x \in \mathcal{X}_{0, x_0, a}$ such that $\exists a \in A : (x_0, a') \in S$, and $\exists a \in A : (x_0, a) \in S$. However, from (ii) and (iii) it follows by definition that $\forall x' \in X$ such that $\|x - x'\|_1 < 2\mu \exists a' \in A : ([x']_{a'}, a') \in S \subseteq R$. Thus, $x \notin \partial R$, which is a contradiction.

Thus, we can finally prove the desired statement.

**Lemma 12:** It holds that $S \subseteq R \implies R(S) \subseteq R(R)$ for both $R = R_c$ and $R = R_c^{\prime}$, where the closure $R_c^{\prime}$ is defined in the same way as $R_c$.

**Proof:** Consider first $R_c^{\prime}$, the adapted reachability operator from [30]. Let $(a, x_0) \in R_c^{\prime}(S)$. Then, the proof is a straightforward extension of [9, Lem. 3(iv)]. In particular, by definition, we have for all $i \in \mathcal{I}_x$, $\exists (a', x_0') \in S, g_i(a', x_0') - \epsilon - L_a(a - a')_1 - L_x(x_0 - x_0')_1 + \mu \geq 0$. But, since $S \subseteq R$, we have that $(a', x_0') \in R$, which then also implies that $(a, x_0) \in R_c^{\prime}(R)$. By repeated application, we get the statement for the closure.

Consider now the global reachability operator $R_c$. Let $(a, x_0) \in R_c(S) \setminus R_c^{\prime}(S)$. Then, by definition, we have $\exists x \in \mathcal{X}_{(a, x_0), a} : (x, a) \in R_c^{\prime}(S)$, which by Lemma 11 implies $\exists x \in \mathcal{X}_{(a, x_0), a} : (x, a) \in R_c^{\prime}(R)$ since $S \subseteq R$. Thus, we have $R_c(S) \subseteq R_c(R)$. Also here we get the statement about the closure by repeated application.

This lets us conclude that if more points can be safely explored, the reachability operator $R_c$ will be non-empty.

**Lemma 13:** For any $n \geq 1$, if $R_c(S_n) \setminus S_n \neq \emptyset$, then $R_c(S_{n+1}) \setminus S_n \neq \emptyset$. Similarly, $R_c^{\prime}(S_n) \setminus S_n \neq \emptyset$ implies $R_c^{\prime}(S_{n+1}) \setminus S_n \neq \emptyset$.

**Proof:** Lemmas 9 and 12 satisfy the conditions for [9, Lem. 6] and the proof is the same as shown therein.

In Alg. 1, we state that we switch to a new stage if $w_n(x_0) < \epsilon$. Thus, to prevent our algorithm from getting stuck, we need to show that this happens eventually. The statement is essentially the same as in [9, Cor. 2]. As shown therein, the time step after which the uncertainty is below $\epsilon$ can be computed as the smallest integer $N_\mu$ satisfying $\frac{\beta N_\mu}{\gamma + N_\mu} \geq \frac{\epsilon}{2\mu}$ and depends on the noise in the system through $C_1 = \frac{1}{\log(1 + \sqrt{\gamma - \epsilon})}$, the information capacity $\gamma$, $\beta$, which we defined in Lemma 1, and on the tolerance $\epsilon$. Generally, we would need to define two individual $N_\mu$ for $S_1$ and $S_2$, since $S_1$ acts in a reduced search space. Consequently, the information capacity of $S_2$ should be larger. Therefore, also its $N_\mu$ should be bigger. For ease of presentation, we consider only one single $N_\mu$, which is the larger of both quantities, i.e., the one we need in $S_2$.

**Lemma 14:** After a finite $N_\mu$ for which $S_n = S_{n+N_\mu}$, we have $w_{n+N_\mu}(a, x_0, i) < \epsilon \forall i \in \mathcal{I}$ for all $(a, x_0) \in G_{n+N_\mu}$. Further, we have $w_{n+N_\mu}(a, x_0, i) < \epsilon \forall i \in \mathcal{I}$ for all $(a, x_0) \in G_{n+N_\mu} \cup M_{n+N_\mu}$.

**Proof:** In $S_1$, we do standard $\text{SafeOpt}$ with $\tilde{x}_0 = x_0$. Thus, the proof for $(a, x_0) \in G_{n+N_\mu} \cup M_{n+N_\mu}$ follows from [9, Cor. 2].

For $S_2$, due to $S_n = S_{n+N_\mu}$, we have $G_{n+N_\mu} \subseteq G_{n}$ since for any $(a, x_0) \in S_n$, $w_n(a, x_0, i) < \epsilon$ for all $i \in \mathcal{I}_x$, since $w_n(a, x_0, i) < \epsilon$ for all $i \in \mathcal{I}_x$. By definition, we have $(a_n, x_n) = \arg \min_{(a, x) \in G_{n}} w_n(a, x, t)$. Thus, we have the same setting as [9, Cor. 2] in only $G$ instead of $G \cup M$. Thus, the proof follows as shown therein.

Taking these results, we can show that after a finite number of evaluations, the safe set must increase unless it is impossible to do so. We first show that if there are pairs $(a, x_0)$ that can be explored during $S_1$ and $S_2$, these will eventually be explored with high probability.

**Lemma 15:** For any $n \geq 1$, if $R_c^{\prime}(S_n) \setminus S_n \neq \emptyset$, then, with probability at least $1 - \delta$, $S_{n+1} \supseteq S_n$.

**Proof:** The proof is a straightforward extension of [9, Lem. 7]. In particular, by Lemma 13, we get that $R_c^{\prime}(S_n) \setminus S_n \neq \emptyset$. Consider $(a, x_0) \in R_c^{\prime}(S_n) \setminus S_n$. Then, by definition,
for all $i \in \mathcal{I}_g$

\[ \exists (a', \tilde{x}_0) \in S_n : \]  
\[ g_i(a', \tilde{x}_0) - \epsilon - L_n||a - a'||_1 - L_\alpha(||\tilde{x}_0 - \tilde{x}_0'||_1 + \mu) \geq 0. \]  

Now, assume that after exploring $N_n$ points, the safe set has not grown, i.e., $S_{n+N_n} = S_n$. Then, we have $(a, \tilde{x}_0) \in \mathcal{A} \times \mathcal{X}_\mu \setminus S_{n+N_n}$ while $(a', \tilde{x}_0') \in S_{n+N_n}$ and (14) holds. Therefore, since by definition $u_n(\cdot) \geq g_i(\cdot) - \epsilon$ (cf., Corollary 2), we also have that $\epsilon_{n+N_n}(a', \tilde{x}_0') > 0$, which implies $(a', \tilde{x}_0') \in G_{n+N_n}$.

Finally, as we have $S_{n+N_n} = S_n$ and $(a', \tilde{x}_0') \in G_{n+N_n}$, Lemma 14 implies

\[ w_{n+N_n}(a', \tilde{x}_0', i) \leq \epsilon \forall i \in \mathcal{I}_g. \]  

To keep notation uncluttered, define $\kappa := L_n||a - a'||_1 - L_\alpha(||\tilde{x}_0 - \tilde{x}_0'||_1 + \mu)$. Then, we have for all $i \in \mathcal{I}_g$

\[ l_{n+N_n}(a', \tilde{x}_0, i) - \kappa \geq g_i(a', \tilde{x}_0) - w_{n+N_n}(a', \tilde{x}_0, i) - \kappa \]  

Cor. 2

\[ \geq g_i(a', \tilde{x}_0) - \epsilon - \kappa \]  

By (15)

\[ \geq 0. \]  

Thus, we get $(a, \tilde{x}_0) \in S_{n+N_n}$, which is a contradiction. ■

Now, we show the same for S3.

Lemma 16: For any $n \geq 1$, if $\mathcal{R}_e \setminus S_n \neq \emptyset$ and $\mathcal{R}^e_n \setminus S_n = \emptyset$, then, with probability at least $1 - \delta$, $S_{n+N_n} \supseteq \mathcal{A} \times \mathcal{X}_\mu \setminus S_n$.

Proof: Lemma 14 implies that the conditions for S1 and S2 are not satisfied and, thus, we are in S3. In the general case, we have $\mathcal{E}_t = \emptyset$ and (6) recommends the most uncertain point in

\[ \{(a, \tilde{x}_0) \in (\mathcal{A} \times \mathcal{X}_\mu) : (a', \tilde{x}_0) \in S_n, a' \in \mathcal{A}\}. \]

This set contains at most $|\mathcal{A} \times \mathcal{X}_\mu \setminus S_n|$ points. Denote with $(\tilde{a}, \tilde{x}_0)$ the point recommended by (6). After completing the rollout procedure in Alg. 1, we have two possible outcomes:

\[ \hat{x} \in \xi_{(0, \tilde{x}_0, a)} : x \in \partial S_n \]  

(16a)

\[ \exists x \in \xi_{(0, \tilde{x}_0, a)} : x \in \partial S_n. \]  

(16b)

In case (16a), we update the safe set with (7) and $S_{n+N_{n+1}} \supseteq S_n$. In case (16b), we claim that $(\tilde{a}, \tilde{x}_0) \notin \mathcal{R}_e(S_n)$. In particular, by assumption, we have $S_n = \mathcal{R}^e(S_n)$. However, $(\tilde{a}, \tilde{x}_0) \notin \mathcal{A} \times \mathcal{X}_\mu \setminus (S_n \times \mathcal{E}_t)$, and, hence, $(\tilde{a}, \tilde{x}_0) \notin \mathcal{R}^e(S_n)$. Moreover, (16b) implies that $\exists x \in \xi_{(0, \tilde{x}_0, a)} : x \in \partial \mathcal{R}^e(S_n)$ as $\mathcal{R}^e(S_n) = S_n$. As $\mathcal{R}_e$ is the union of 2 sets and the reasoning above forbids $(\tilde{a}, \tilde{x}_0)$ to be part of either, we have $(\tilde{a}, \tilde{x}_0) \notin \mathcal{R}_e(S_n)$. Thus, $(\tilde{a}, \tilde{x}_0)$ is added to $\mathcal{E}_t$ and not suggested in further iterations.

S3 can be repeated at most $|\mathcal{A} \times \mathcal{X}_\mu \setminus S_n|$ times. If we are in case (16b) for the first $|\mathcal{A} \times \mathcal{X}_\mu \setminus S_n| - 1$ iterations, we end up with $|\mathcal{A} \times \mathcal{X}_\mu \setminus (S_n \times \mathcal{E}_t)| = 1$. Since, by assumption, $\mathcal{R}_e \setminus S_n \neq \emptyset$, we have $\mathcal{R}_e \neq \emptyset$ by Lemma 13. Thus, at iteration $|\mathcal{A} \times \mathcal{X}_\mu \setminus S_n|$, we must have $(\tilde{a}, \tilde{x}_0) \in \mathcal{R}_e(S_n)$. By definition of $\mathcal{R}_e(S_n)$, we know that the trajectory induced by $(\tilde{a}, \tilde{x}_0)$ will not hit $\partial \mathcal{R}^e(S_n)$. Further, as we have $\mathcal{R}^e(S_n) \setminus S_n = \emptyset$ by assumption, we have $\mathcal{R}^e(S_n) = S_n$ and, therefore, \[ \partial \mathcal{R}^e(S_n) = \partial S_n. \] That is, $(\tilde{a}, \tilde{x}_0)$ will induce a successful experiment. Then, we can update the safe set with (7), which completes the proof.

Taking both results lets us finally state that we will eventually enlarge the safe set unless it is impossible to do so:

**Corollary 3:** If $\mathcal{R}_e \setminus S_n \neq \emptyset$, $S_{n+N_n+1} \supseteq \mathcal{A} \times \mathcal{X}_\mu \setminus S_n$.

Proof: Due to Lemma 13, $\mathcal{R}_e \setminus S_n \neq \emptyset$ implies $\mathcal{R}_e \setminus S_n \neq \emptyset$. Thus, the conditions for either S1 or S2 or S3 are satisfied. In case S1 or S2, $S_{n+N_n} \supseteq S_n$ follows from Lemma 15. In case S3, $S_{n+N_n+1} \supseteq \mathcal{A} \times \mathcal{X}_\mu \setminus S_n$ follows from Lemma 16.

**Remark 1:** In Alg. 1 and the theoretical results, we assumed that we first expand locally (using S1 and S2) as much as possible and only then do a global search step S3. However, the definition of the global reachability operator $\mathcal{R}_e$ and its closure also allow alternating the stages. Following such an approach would result in many more S3 experiments that would need to be stopped prematurely and repeated as the safe set increases, but, ultimately, would converge to the same result.

We can then show that we get $\epsilon$-close to the optimum value within the safely reachable set, $\mathcal{R}_e(S_n)$:

**Lemma 17:** Define $\hat{a} = \arg \max_{(a, x_0) \in \mathcal{R}_e(S_n)} l_n(a, x_0, 0)$, where setting the last argument of $l_n(\cdot)$ to zero indicates that we are considering the lower confidence bound of the reward function. Further, define $N_{\max, n} = |\mathcal{A} \times \mathcal{X}_\mu \setminus S_n| + N_n$. If for any $n \geq 1$, $S_{n+N_{\max, n}} = S_n$, then the following holds with probability at least $1 - \delta$:

\[ f(\hat{a}, x_0) \geq \max_{(a, x_0) \in \mathcal{R}_e(S_n)} f(a, x_0) - \epsilon. \]  

(17)

Proof: Let

\[ (a^*, x_0) := \arg \max_{(a, x_0) \in S_{n+N_{\max, n}}} f(a, x_0), \]

which is by definition contained in $M_{n+N_{\max, n}}$. This implies $w_{n+N_{\max, n}}(a^*, x_0, i) < \epsilon \forall i \in \mathcal{I}$ by Lemma 14. For the sake of contradiction, assume that

\[ f(\hat{a}, x_0) < f(a^*, x_0) - \epsilon. \]  

(18)

Then, we have

\[ l_{n+N_{\max, n}}(a^*, x_0, 0) \leq l_{n+N_{\max, n}}(\hat{a}, x_0, 0) \quad \text{By Def. of } \hat{a} \]

\[ \leq f(\hat{a}, x_0) \quad \text{Lem. 1} \]

\[ < f(a^*, x_0) - \epsilon \quad \text{By (18)} \]

\[ \leq u_{n+N_{\max, n}}(a^*, x_0, 0) - \epsilon \quad \text{Lem. 1} \]

\[ \leq l_{n+N_{\max, n}}(a^*, x_0, 0), \quad \text{as } a^* \in M_{n+N_{\max, n}} \]

which is a contradiction.

Finally, by Corollary 3, $S_{n+N_{\max, n}} = S_n$ implies that $\mathcal{R}_e(S_n) \subseteq S_n = S_{n+N_{\max, n}}$. Therefore,

\[ \max_{(a, x_0) \in \mathcal{R}_e(S_n)} f(a, x_0) - \epsilon \leq \max_{(a, x_0) \in S_{n+N_{\max, n}}} f(a, x_0) - \epsilon \]

\[ = f(a^*, x_0) - \epsilon \quad \text{(19)} \]

\[ \leq f(\hat{a}, x_0). \]  

■
For Lemma 17, we need that $S_{n+N_{\max,n}} = S_n$ is satisfied for some $n$. The fact that this happens follows from [9, Lem. 10]. Thus, GoSAFE will converge to the optimum of the safely reachable set with $\epsilon$-precision with probability at least $1 - \delta$, which proves Theorem 2. If the optimum of the safely reachable set is at the same time the global optimum, GoSAFE converges with $\epsilon$-precision to the global optimum, which proves Corollary 1.