Safe Pontryagin Differentiable Programming

Anonymous Author(s)
Affiliation
Address
email

Abstract

We propose a Safe Pontryagin Differentiable Programming (Safe PDP) methodology, which establishes a theoretical and algorithmic safe differentiable framework to solve a broad class of safety-critical learning and control tasks—problems that require the guarantee of both immediate and long-term constraint satisfaction at any stage of the learning and control progress. In the spirit of interior-point methods, Safe PDP handles different types of state and input constraints by incorporating them into the cost and loss through barrier functions. We prove the following fundamental features of Safe PDP: first, both the constrained solution and its gradient in backward pass can be approximated by solving a more efficient unconstrained counterpart; second, the approximation for both the solution and its gradient can be controlled for arbitrary accuracy using a barrier parameter; and third, importantly, any intermediate results throughout the approximation and optimization are strictly respecting all constraints, thus guaranteeing safety throughout the entire learning and control process. We demonstrate the capabilities of Safe PDP in solving various safe learning and control tasks, including safe policy optimization, safe motion planning, and learning MPCs from demonstrations, on different challenging control systems such as 6-DoF maneuvering quadrotor and 6-DoF rocket powered landing.

1 Introduction

Safety is usually a priority in the deployment of a learning and control algorithm to real-world systems. For a physical system (agent), safety is normally given in various states and inputs constraints, which must not be violated by the algorithm at *any stage* of the learning and control process, otherwise will cause irrevocable or unacceptable failure/damage/loss. Those systems are referred to as *safety-critical*. The constraints in a safety-critical system can include the *immediate* ones, which are directly imposed on the system state and input at certain/all time instances, and the *long-term* ones, which are defined on the trajectory of system states and inputs over a long time period.

Compared to the abundant results that focus on system optimality [1–3], systematic and principled treatments for safety-critical learning and control problems seem largely insufficient, particularly in the following gaps (detailed in Section 1.1). First, existing safety strategies are either too conservative, which may restrict the task performance, or violation-tolerable, which only pursues the near-constraint guarantee and thus are not strictly constraint-respecting. Second, a generic safety paradigm capable of handling different types of constraints, including system state and input (or mixed), immediate, or/and long-term constraints, is still lacked. Third, some existing safety strategies suffer from huge computational- and data- complexity, difficult to be integrated into any differentiable programming frameworks to solve large-scale learning and continuous control tasks.

To address the above research gaps, this paper aims to develop a new theoretical and algorithmic framework for safe learning and control with the following key abilities. First, the framework provides a systematic treatment for *different types of constraints* in a safety-critical problem; second, it attains *provable safety- and optimality- guarantees* throughout the learning and control process; third, it is flexible to perform *safe learning of any unknown aspects of a safety-critical system*, including policy, dynamics, state and input constraints, and control cost function; finally, it can be integrated into any *differentiable programming framework* to efficiently solve large-scale safe learning and control tasks.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

1.1 Related Work

In both learning and control fields, safety has been defined in different criteria, such as worst case [4, 5], risk-sensitive [6], ergodicity [7], (adversarial) robust [8, 9], etc., most of which are formulated by directly altering an objective function [10]. In this paper, we focus only on constrained learning and control problems, where constraints are *explicitly formulated and must be satisfied*. We categorize existing constrained learning and control techniques into the *at-convergence safety* methods, which only concern the solution satisfying constraints at convergence, or the *in-progress safety* methods, which attempt to guarantee constraint satisfaction during the entire learning and control process.

At-convergence safety methods. In reinforcement learning (RL), a constrained agent is typically formulated as a Constrained Markov Decision Process (CMDP) [11], seeking a policy that not only optimizes a reward but also satisfies the upper bounds for certain cumulative costs. A commonly used strategy [12–17] to solve CMDPs is to use the primal-dual method, by establishing the unconstrained Lagrangian and performing saddle-point optimization. In deep learning, recent work [18] also uses the primal-dual method to train deep neural networks with constraints. In control, the primal-dual method has been used to solve constrained optimal control problems [19, 20] and constrained trajectory optimization [21]. While the solutions to all the above problems can be proved to satisfy constraints at convergence [22, 23], the primal-dual type methods cannot guarantee constraint satisfaction during optimization, as experimentally shown in [13, 24], thus are not suitable for safety-critical tasks.

In-progress safety methods. To achieve safety during training, [25] and recent [26] solve CMDPs by introducing additional constraints into the Trust Region Policy Optimization (TRPO) [27]. Despite their promising results, some potential drawbacks remain unresolved: first, these methods only give the 'near constraint' guarantee and have not fully eliminated constraint violation; and second, as their implementation to continuous problems require function approximations, it is not clear if safety is still guaranteed in presence of approximation errors. Another line of constrained RL [24, 28–30] leverages Lyapunov theory [31] to bound the state of an agent within a sub-level set of the Lyapunov function. However, choosing a valid Lyapunov function for general systems is still an open problem to date [32], particularly for the constrained RL here, since it requires the Lyapunov function to be both consistent with the constraints and non-conservative to permit the true optimal policy (unknown, however) [28]. Besides, all aforementioned constrained RL based on CMDP formulation only considers cumulative constraints and cannot handle the immediate ones— constraints directly on agent state and input at all time. In practice, the immediate constraints are more common; e.g., for a robot arm, we more care about if it is in a safe workspace at any time and if the input motor torques are in the allowable range.

In control, in-progress safety can be achieved in two model-based frameworks: reachability theory [33] and control barrier functions [34, 35]. Safe control based on the reachability theory [33, 36–38] explicitly considers the adversarial factors in a system, such as model uncertainty and disturbances, and seeks a strategy that maintains the system within specified constraints in spite of the adversarial factors. However, the safety is obtained at the cost of solving the Hamilton-Jacobi-Isaacs equations [39], which become computationally intractable for high-dimensional systems [33]. Control barrier functions [34, 35] enable to constrain a system only on safety boundaries, making it a less-conservative safety strategy and appealing for safety-critical tasks [40–42]. Most of the methods based on control barrier functions consider affine dynamics (in inputs) and directly use the given constraint function as a control barrier function. Such a choice could be problematic for the system which is uncontrollable at the boundary of the sublevel set (i.e., whatever allowable control, the sublevel set of the constraint function is never invariant). Thus, how to find a valid control barrier function is still an ongoing research topic [43–45]. The above two safe control frameworks favorably consider the pure state constraints and cannot be readily extended to other constraints, such as constraints defined on both state and input, long-term cumulative constraints defined on the system trajectory level, etc.

Interior-point methods and control. Interior-point methods (IPMs) [46–48] solve constrained optimization by sequentially finding solutions to unconstrained problems with the objective combining the original objective and a barrier that prevents from leaving feasible regions. IPMs have been used to solve *constrained* optimal control problems in [49–55], but they are limited to constrained linear quadratic regulars (LQRs). While IPMs for nonlinear constrained optimal control are studied in [56–60], they mostly focus on developing efficient algorithms to solve the unconstrained approximation (from the perturbed KKT conditions) and lack of a complete proof for asymptotic convergence of true control solution. More notably, all existing work has not studied/established the *results regarding the differentiablity of general constrained optimal control systems*, which are crucial in order to integrate a control system into any differentiable programming framework for large-scale computational tasks.

1.2 Paper Contributions

This paper proposes a safe differentiable programming methodology named as *Safe Pontryagin Differentiable Programming* (or Safe PDP). Safe PDP provides a systematic treatment of different types of system constraints, including state and inputs (or mixed), immediate, and long-term constraints, with provable safety- and performance-guarantee. Safe PDP also establishes a unified differentiable programming framework to efficiently solve a board class of safety-critical learning and control tasks, such as safe policy optimization, safe motion planning, and safe learning from demonstrations, etc.

In the spirit of interior-point methods, Safe PDP incorporates different types of constraints into the control cost and loss through barrier functions, approximating a constrained problem using an unconstrained counterpart. The contributions of Safe PDP are both theoretical and algorithmic. Theoretically, we prove that (I) not only the constrained solution but also its gradient in backward pass can be safely approximated by solving a more efficient unconstrained counterpart (see Theorem 2); (II) any intermediate results throughout the approximation and optimization are strictly safety-guaranteed, i.e., never violating constraints, (see Theorem 2 and Theorem 3); and (III) the approximations for both solution and gradient can be controlled for arbitrarily accuracy by a simple barrier parameter (see Theorem 2 and Theorem 3), which could mean that the 'conservativeness (i.e., over-approximations)' for safety can be well controlled. Arithmetically, (IV) Safe PDP also provides an efficient algorithm to differentiate the solution of a constrained optimal control system (see Theorem 1); and (V) built on the previous PDP [61], Safe PDP is capable and versatile to efficiently solve a board class of safety-critical learning and control problems, including safe policy optimization, safe motion planning, learning MPCs from demonstrations (see Section 7). All the above theoretical and algorithmic features of Safe PDP has not been (fully) achieved in any previous work to authors' best knowledge.

2 Safe PDP Problem Formulation

Consider a class of constrained optimal control systems (models) $\boldsymbol{\Sigma}(\boldsymbol{\theta})$, which are parameterized by a tunable parameter $\boldsymbol{\theta} \in \mathbb{R}^r$ in its control cost function, dynamics, initial condition, and constraints:

$$\boldsymbol{\Sigma}(\boldsymbol{\theta}): \quad \begin{aligned} &\textit{control cost:} && J(\boldsymbol{\theta}) = \sum_{t=0}^{T-1} c_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) + c_T(\boldsymbol{x}_T, \boldsymbol{\theta}) \\ &\text{subject to} && \\ &\textit{dynamics:} && \boldsymbol{x}_{t+1} = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) \quad \text{with} \quad \boldsymbol{x}_0 = \boldsymbol{x}_0(\boldsymbol{\theta}) \\ &\textit{terminal constraints:} && \boldsymbol{g}_T(\boldsymbol{x}_T, \boldsymbol{\theta}) \leq \boldsymbol{0}, \quad \boldsymbol{h}_T(\boldsymbol{x}_T, \boldsymbol{\theta}) = \boldsymbol{0} \\ &\textit{path constraints:} && \boldsymbol{g}_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) \leq \boldsymbol{0}, \quad \boldsymbol{h}_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) = \boldsymbol{0}. \end{aligned} \tag{1}$$

Here, $\boldsymbol{x}_t \in \mathbb{R}^n$ is the system state; $\boldsymbol{u}_t \in \mathbb{R}^m$ is the control input; $c_t : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^r \to \mathbb{R}$ and $c_T : \mathbb{R}^n \times \mathbb{R}^r \to \mathbb{R}$ are the stage and final costs, respectively; $\boldsymbol{f} : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^r \to \mathbb{R}^n$ is the dynamics with initial state $\boldsymbol{x}_0 = \boldsymbol{x}_0(\boldsymbol{\theta}) \in \mathbb{R}^n$; $t = 0, 1, ..., T$ is the time step with T the time horizon; $\boldsymbol{g}_T : \mathbb{R}^n \times \mathbb{R}^r \to \mathbb{R}^{q_T}$ and $\boldsymbol{h}_T : \mathbb{R}^n \times \mathbb{R}^r \to \mathbb{R}^{s_T}$ are the final inequality and equality constraints, respectively; $\boldsymbol{g}_t : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^r \to \mathbb{R}^{q_t}$ and $\boldsymbol{h}_t : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^r \to \mathbb{R}^{s_t}$ are the immediate inequality and equality constraints, respectively. All inequalities (here and below) are entry-wise. We consider that all functions in $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ are three-times continuously differentiable (i.e., C^3) with respect to its arguments. Although we here have parameterized all aspects of $\boldsymbol{\Sigma}(\boldsymbol{\theta})$, for a specific application (see Section 7), one only needs to parameterize the unknown aspects in $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ and keep others given. Any unknown aspect in $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ can be implemented using differentiable neural networks. For a given $\boldsymbol{\theta}$, the above $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ produces a trajectory $\boldsymbol{\xi}_{\boldsymbol{\theta}} = \{\boldsymbol{x}^{\boldsymbol{\theta}}_{0:T}, \boldsymbol{u}^{\boldsymbol{\theta}}_{0:T-1}\}$ by solving the following Problem B($\boldsymbol{\theta}$):

$$\begin{aligned} \boldsymbol{\xi}_{\boldsymbol{\theta}} = \{\boldsymbol{x}^{\boldsymbol{\theta}}_{0:T}, \boldsymbol{u}^{\boldsymbol{\theta}}_{0:T-1}\} \in \arg\min_{\{\boldsymbol{x}_{0:T}, \boldsymbol{u}_{0:T-1}\}} \quad & J(\boldsymbol{\theta}) \\ \text{subject to} \quad & \boldsymbol{x}_{t+1} = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) \quad \text{with} \quad \boldsymbol{x}_0 = \boldsymbol{x}_0(\boldsymbol{\theta}), \\ & \boldsymbol{g}_T(\boldsymbol{x}_T, \boldsymbol{\theta}) \leq \boldsymbol{0} \quad \text{and} \quad \boldsymbol{h}_T(\boldsymbol{x}_T, \boldsymbol{\theta}) = \boldsymbol{0}, \\ & \boldsymbol{g}_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) \leq \boldsymbol{0} \quad \text{and} \quad \boldsymbol{h}_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) = \boldsymbol{0}. \end{aligned} \tag{B($\boldsymbol{\theta}$)}$$

Here, we use $\in$ since $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ to Problem B($\boldsymbol{\theta}$) may not be unique in general, thus constituting a solution set $\{\boldsymbol{\xi}_{\boldsymbol{\theta}}\}$. We will discuss the existence and uniqueness of $\{\boldsymbol{\xi}_{\boldsymbol{\theta}}\}$ in Section 4.

For a specific task, we aim to find a specific model $\boldsymbol{\Sigma}(\boldsymbol{\theta}^*)$, i.e, searching for a specific $\boldsymbol{\theta}^*$, such that its trajectory $\boldsymbol{\xi}_{\boldsymbol{\theta}^*}$ from B($\boldsymbol{\theta}^*$) meets the following two *given* requirements. First, $\boldsymbol{\xi}_{\boldsymbol{\theta}^*}$ minimizes a *given task loss* $\ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$; and second, $\boldsymbol{\xi}_{\boldsymbol{\theta}^*}$ satisfies the *given task constraints* $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) \leq 0$, $i = 1, 2, ..., l$.

Note that, we need to distinguish between the two types of objective functions: task loss $\ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ and control cost $J(\boldsymbol{\theta})$, and also the two types of constraints: task constraints $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ and model constraints $\boldsymbol{g}_t(\boldsymbol{\theta})$. In fact, $J(\boldsymbol{\theta})$ and $\boldsymbol{g}_t(\boldsymbol{\theta})$ in $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ are *unknown and parameterized by* $\boldsymbol{\theta}$ and can represent the unknown inherent aspects of a physical agent, while $\ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ and $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ are *given and known* according to the specific task (note that they also explicitly include $\boldsymbol{\theta}$ since $\boldsymbol{\theta}$ needs to be regularized in some learning cases). The problem of searching for $\boldsymbol{\theta}^*$ can be formally written as:

$$\begin{aligned} \boldsymbol{\theta}^* \;=\; & \arg\min_{\boldsymbol{\theta}} \quad \ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) \\ & \text{subject to} \quad R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) \leq 0, \quad i = 1, 2, ..., l, \\ & \qquad\quad \boldsymbol{\xi}_{\boldsymbol{\theta}} \text{ solves Problem B}(\boldsymbol{\theta}) \ . \end{aligned} \tag{P}$$

Assume $\ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ and $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ are both twice-continuously differentiable. Under Problem P, for a specific learning and control task, one only needs to specify the parameterization details of $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ and give a task loss $\ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ and constraints $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) \leq 0$. Section 7 will give representative examples.

3 Gradient-based Methods to Solve Problem P and its Challenges

In essence, Problem P belongs to bi-level optimization [62]—each time the parameter $\boldsymbol{\theta}$ is updated in Problem P, the corresponding trajectory $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ needs to be solved from the lower-level Problem B($\boldsymbol{\theta}$). Intuitively, one could approach Problem P using gradient-based methods by ignoring the process of solving Problem B($\boldsymbol{\theta}$) and just viewing $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ as an explicit differentiable function of $\boldsymbol{\theta}$. Thus, similar to interior-point methods [46], one can introduce a logarithmic barrier function for each task constraint, $-\ln\big(-R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})\big)$, and a barrier parameter $\epsilon > 0$. This leads to solving the following unconstrained Problem SP(ϵ) sequentially using any off-the-shelf gradient-descent solver:

$$\boldsymbol{\theta}^*(\epsilon) = \arg\min_{\boldsymbol{\theta}} \ \ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) - \epsilon \sum\nolimits_{i=1}^{l} \ln\big(-R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})\big) \tag{SP(ϵ)}$$

for a fixed ϵ. By controlling $\epsilon \to 0$, $\boldsymbol{\theta}^*(\epsilon)$ is expected to converge to the solution $\boldsymbol{\theta}^*$ to Problem P. Although plausible, the above process has the following technical challenges to be addressed:

1) Since $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ is a solution to the constrained optimal control Problem B($\boldsymbol{\theta}$), is $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ differentiable? how to efficiently obtain $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ with complexity only $\mathcal{O}(T)$?
2) Since we want to obtain $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ at as low cost as possible, instead of handling the *constrained* Problem B($\boldsymbol{\theta}$), can we use *unconstrained* one to approximate both $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$? More importantly, can the accuracy of the approximations for $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ be arbitrarily and safely controlled?
3) If we approximate $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$, can we guarantee that such approximations are safe in a sense that any approximation for $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ always respect the system constraints $\boldsymbol{g}_t \leq 0$ and $\boldsymbol{g}_T \leq 0$?
4) Finally, if we want to use the unconstrained Problem SP(ϵ) to approximate Problem P, under what conditions the above method works? and can we guarantee that any indeterminate $\boldsymbol{\theta}^*(\epsilon)$ will never violate the task constraints, i.e., $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}^*(\epsilon)}, \boldsymbol{\theta}^*(\epsilon)) \leq 0$, $i = 1, 2, ..., l$?

We will address the above challenges in the following sections. For better reference, we here provide an overview of all results. Challenge 1) will be addressed in Section 4 and the result is in Theorem 1, where $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ will be efficiently solved using the auxiliary control system $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$ in (3). Challenges 2) and 3) will be addressed in Section 5 and the results are in Theorem 2, where both $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ will be approximated using the unconstrained system $\boldsymbol{\Sigma}(\boldsymbol{\theta}, \gamma)$ in (5). Challenge 4) will be addressed in Section 6 and the result is in Theorem 3. Section 7 will give some representative applications.

4 Differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and Auxiliary Control System for $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$

4.1 Differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$

For the constrained optimal control system $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ in (1), we define the following Hamiltonian L_t for $t = 0, 1, ..., T-1$ and L_T, respectively,

$$L_t = c_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) + \boldsymbol{\lambda}_{t+1}' \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) + \boldsymbol{v}_t' \boldsymbol{g}_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) + \boldsymbol{w}_t' \boldsymbol{h}_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}), \tag{2a}$$

$$L_T = c_T(\boldsymbol{x}_T, \boldsymbol{\theta}) + \boldsymbol{v}_T' \boldsymbol{g}_T(\boldsymbol{x}_T, \boldsymbol{\theta}) + \boldsymbol{w}_T' \boldsymbol{h}_T(\boldsymbol{x}_T, \boldsymbol{\theta}), \tag{2b}$$

where $\boldsymbol{\lambda}_t \in \mathbb{R}^n$ is the costate, $\boldsymbol{v}_t \in \mathbb{R}^{q_t}$ and $\boldsymbol{w}_t \in \mathbb{R}^{s_t}$ are multipliers for the inequality and equality constraints, respectively. The well-known second-order condition for $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ to be *a local isolated (locally unique) minimizing trajectory* to $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ in Problem B($\boldsymbol{\theta}$) has been well-established since [63]. For completeness, we present it in Lemma A.2 in Appendix A. Lemma A.2 also states that there exist costate sequence $\boldsymbol{\lambda}^{\boldsymbol{\theta}}_{1:T}$, and multiplier sequences $\boldsymbol{v}^{\boldsymbol{\theta}}_{0:T}$ and $\boldsymbol{w}^{\boldsymbol{\theta}}_{0:T}$, such that ($\boldsymbol{\xi}_{\boldsymbol{\theta}}$, $\boldsymbol{\lambda}^{\boldsymbol{\theta}}_{1:T}$, $\boldsymbol{v}^{\boldsymbol{\theta}}_{0:T}$, $\boldsymbol{w}^{\boldsymbol{\theta}}_{0:T}$) satisfies the well-known Constrained Pontryagin Minimum Principle (C-PMP) given in (S.5) in Lemma A.2. Based on the above, one can have the following result for the differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$.

Lemma 1 (Differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$)**.** *Given a fixed $\bar{\boldsymbol{\theta}}$, consider the following conditions hold for $\boldsymbol{\Sigma}(\bar{\boldsymbol{\theta}})$:*

(i) the second-order conditions (Lemma A.2) are satisfied for $\boldsymbol{\Sigma}(\bar{\boldsymbol{\theta}})$;

(ii) the gradients of all binding constraints at $\boldsymbol{\xi}_{\bar{\boldsymbol{\theta}}}$ are linearly independent (binding constraints includes all equality and active inequality constraints $g_{t,i}$=0, t=0, 1, ..., T, i=1, 2, ..., q_t);

(iii) strict complementarity holds at $\boldsymbol{\xi}_{\bar{\boldsymbol{\theta}}}$, i.e., active inequality constraint has positive multiplier.

Then, for all $\boldsymbol{\theta}$ in a neighborhood of $\bar{\boldsymbol{\theta}}$, there exists a unique once-continuously differentiable function, ($\boldsymbol{\xi}_{\boldsymbol{\theta}}$, $\boldsymbol{\lambda}^{\boldsymbol{\theta}}_{1:T}$, $\boldsymbol{v}^{\boldsymbol{\theta}}_{0:T}$, $\boldsymbol{w}^{\boldsymbol{\theta}}_{0:T}$), satisfying the second-order conditions (Lemma A.2) for the constrained optimal control system $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ with $\left(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\lambda}^{\boldsymbol{\theta}}_{1:T}, \boldsymbol{v}^{\boldsymbol{\theta}}_{0:T}, \boldsymbol{w}^{\boldsymbol{\theta}}_{0:T}\right)=\left(\boldsymbol{\xi}_{\bar{\boldsymbol{\theta}}}, \boldsymbol{\lambda}^{\bar{\boldsymbol{\theta}}}_{1:T}, \boldsymbol{v}^{\bar{\boldsymbol{\theta}}}_{0:T}, \boldsymbol{w}^{\bar{\boldsymbol{\theta}}}_{0:T}\right)$ at $\boldsymbol{\theta}=\bar{\boldsymbol{\theta}}$, hence, $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ is a local isolated minimizing trajectory to $\boldsymbol{\Sigma}(\boldsymbol{\theta})$; further, for all $\boldsymbol{\theta}$ near $\bar{\boldsymbol{\theta}}$, the strict complementarity is preserved, and the linear independence of the gradients of all binding constraints at $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ hold.

The proof of Lemma 1 can directly follow the well-known first-order sensitivity result in Theorem 2.1 in [64]. Here, Conditions (i)-(iii) are the sufficient conditions to guarantee the applicability of the well-known implicit function theorem [65] to the C-PMP. Condition (ii) is well-known and serves as a sufficient condition for the constraint qualification to establish the C-PMP (see Corollary 3, pp. 22, [46]). Condition (iii) is necessary to ensure that the Jacobian matrix in the implicit function theorem is invertible, and it also leads to the persistence of strict complementarity, saying that the inactive inequalities remain inactive and active ones remain active and there is no 'switch' between them near $\bar{\boldsymbol{\theta}}$. Both our empiricism and previous algorithms [61, 64, 66–68] show that the conditions (i)-(iii) are mild and the differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ can be achieved almost everywhere in the space of $\boldsymbol{\theta}$.

4.2 Auxiliary Control System to Solve $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$

If the conditions (i)-(iii) in Lemma 1 for the differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ hold, we next show that $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ can also be efficiently solved by introducing an auxiliary control system, similar to the PDP developed in [61]. First, we define the new state and input (matrix) variables $X_t = \frac{\partial \boldsymbol{x}_t}{\partial \boldsymbol{\theta}} \in \mathbb{R}^{n\times r}$ and $U_t = \frac{\partial \boldsymbol{u}_t}{\partial \boldsymbol{\theta}} \in \mathbb{R}^{m\times r}$, respectively. Then, we introduce the following *auxiliary control system*,

$$\boxed{\begin{aligned}
\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}}): \quad & \textit{control cost:} \quad \bar{J} = \mathrm{Tr}\sum_{t=0}^{T-1}\left(\frac{1}{2}\begin{bmatrix}X_t\\U_t\end{bmatrix}'\begin{bmatrix}L^{xx}_t & L^{xu}_t\\ L^{ux}_t & L^{uu}_t\end{bmatrix}\begin{bmatrix}X_t\\U_t\end{bmatrix} + \begin{bmatrix}L^{x\theta}_t\\L^{u\theta}_t\end{bmatrix}'\begin{bmatrix}X_t\\U_t\end{bmatrix}\right)\\
& \qquad\qquad + \mathrm{Tr}\left(\frac{1}{2}X_T' L^{xx}_T X_T + (L^{x\theta}_T)' X_T\right)\\
& \text{subject to}\\
& \textit{dynamics:} \quad X_{t+1} = F^x_t X_t + F^u_t U_t + F^\theta_t \quad \text{with} \quad X_0 = X^{\boldsymbol{\theta}}_0\\
& \textit{terminal constraint:} \quad \bar{G}^x_T X_T + \bar{G}^\theta_T = \mathbf{0}, \qquad H^x_T X_T + H^\theta_T = \mathbf{0},\\
& \textit{path constraint:} \quad \bar{G}^x_t X_t + \bar{G}^u_t U_t + \bar{G}^\theta_t = \mathbf{0}, \qquad H^x_t X_t + H^u_t U_t + H^\theta_t = \mathbf{0}.
\end{aligned}} \tag{3}$$

Here, L^x_t and L^{xx}_t denote the first- and second- order derivatives, respectively, of the Hamiltonian L_t in (2) with respect to $\boldsymbol{x}$; F^x_t, H^x_t, $\bar{G}_t$ denote the first-order derivatives of $\boldsymbol{f}_t$, $\boldsymbol{h}_t$, $\bar{\boldsymbol{g}}_t$, respectively, with respect to $\boldsymbol{x}$, where $\bar{\boldsymbol{g}}_t$ is the vector of stacking all active inequality constraints in $\boldsymbol{g}_t$; and the similar convention applies to other and below notations. All derivative matrices defining $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$ are evaluated at $\left(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\lambda}^{\boldsymbol{\theta}}_{1:T}, \boldsymbol{v}^{\boldsymbol{\theta}}_{0:T}, \boldsymbol{w}^{\boldsymbol{\theta}}_{0:T}\right)$, where $\boldsymbol{\lambda}^{\boldsymbol{\theta}}_{1:T}$, $\boldsymbol{v}^{\boldsymbol{\theta}}_{0:T}$, and $\boldsymbol{w}^{\boldsymbol{\theta}}_{0:T}$ are usually the byproducts of a constrained optimal control solver [69] or can be easily solved from the C-PMP given $\boldsymbol{\xi}_{\boldsymbol{\theta}}$, as done in [61]. We note that $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$ is a *Equality-constrained Linear Quadratic Regulator (LQR)* system, as its control cost function is quadratic and dynamics and constraints are linear. For the above $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$, we have the following important result *without additional assumptions*.

Theorem 1 ($\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ is a global unique minmizing trajectory to $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$)**.** *Let the conditions (i)-(iii) in Lemma 1 for differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ to the constrained optimal control system $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ hold. Then, the auxiliary control system $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$ in (3) has a unique global minimizing trajectory, denoted as $\left\{X^{\boldsymbol{\theta}}_{0:T}, U^{\boldsymbol{\theta}}_{0:T-1}\right\}$, which is exactly $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$, i.e.,*

$$\left\{X^{\boldsymbol{\theta}}_{0:T}, U^{\boldsymbol{\theta}}_{0:T-1}\right\} = \frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \quad \textit{with} \quad X^{\boldsymbol{\theta}}_t = \frac{\partial \boldsymbol{x}^{\boldsymbol{\theta}}_t}{\partial \boldsymbol{\theta}} \quad \textit{and} \quad U^{\boldsymbol{\theta}}_t = \frac{\partial \boldsymbol{u}^{\boldsymbol{\theta}}_t}{\partial \boldsymbol{\theta}}. \tag{4}$$

The proof of the above theorem is in Appendix B. Theorem 1 states that as long as the conditions (i)-(iii) in Lemma 1 for differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ are satisfied, without additional assumptions, the auxiliary control system $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$ in (3) always has the unique global minimizing trajectory, which is exactly $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$. Thus, obtaining $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ is equivalent to solving $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$, which be efficiently done thanks to the recent development of the equality-constrained LQR algorithms [70–72], all having a complexity of $\mathcal{O}(T)$. The algorithm that implements Theorem 1 is given in Algorithm 1 in Appendix E.1.

5 Safe Unconstrained Approximation of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$

From Section 4, we know that one can solve the constrained optimal control $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ to obtain $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and solve its auxiliary control system $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$ to obtain $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$. Although theoretically appealing, there are several implementation challenges. First, solving a constrained optimal control Problem B($\boldsymbol{\theta}$) is difficult since the popular trajectory optimization methods, such as iLQR [73] and DDP [74], are designed only for unconstrained systems. Second, establishing $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$ requires to additionally know the values of multipliers $\boldsymbol{v}^{\boldsymbol{\theta}}_{0:T}$ and $\boldsymbol{w}^{\boldsymbol{\theta}}_{0:T}$. Third, to construct $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$, one also needs to identify all those active inequality constraints, which can be numerically difficult due to numerical error (as we will show in later experiments). All of those difficulties motive us to develop a more efficient paradigm to obtain both $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ from $\boldsymbol{\Sigma}(\boldsymbol{\theta})$, which is the goal of this section.

To proceed, we first convert the constrained optimal control system $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ to an unconstrained system $\boldsymbol{\Sigma}(\boldsymbol{\theta}, \gamma)$ by adding all constraints to its control cost function using logarithm barrier function for each inequality and quadratic barrier function for each equality, and $\gamma > 0$ is a barrier parameter, that is,

$$\boldsymbol{\Sigma}(\boldsymbol{\theta}, \gamma): \quad \begin{aligned} \textit{control cost:} \quad J(\boldsymbol{\theta}, \gamma) = & \sum_{t=0}^{T-1} \Big(c_t(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) - \gamma \sum_{i=1}^{q_t} \ln\big(-g_{t,i}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta})\big) + \\ & \frac{1}{2\gamma} \sum_{i=1}^{s_t} \big(h_{t,i}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta})\big)^2 \Big) + c_T(\boldsymbol{x}_T, \boldsymbol{\theta}) - \\ & \gamma \sum_{i=1}^{q_T} \ln\big(-g_{T,i}(\boldsymbol{x}_T, \boldsymbol{\theta})\big) + \frac{1}{2\gamma} \sum_{i=1}^{s_T} \big(h_{T,i}(\boldsymbol{x}_T, \boldsymbol{\theta})\big)^2 \\ \textit{dynamics:} \quad \boldsymbol{x}_{t+1} = & \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) \quad \text{with} \quad \boldsymbol{x}_0 = \boldsymbol{x}_0(\boldsymbol{\theta}). \end{aligned} \tag{5}$$

The trajectory $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)} = \left\{\boldsymbol{x}^{(\boldsymbol{\theta},\gamma)}_{0:T}, \boldsymbol{u}^{(\boldsymbol{\theta},\gamma)}_{0:T-1}\right\}$ produced by the above unconstrained system $\boldsymbol{\Sigma}(\boldsymbol{\theta}, \gamma)$ is

$$\begin{aligned} \boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)} = \left\{\boldsymbol{x}^{(\boldsymbol{\theta},\gamma)}_{0:T}, \boldsymbol{u}^{(\boldsymbol{\theta},\gamma)}_{0:T-1}\right\} \in \arg\min_{\{\boldsymbol{x}_{0:T}, \boldsymbol{u}_{0:T-1}\}} \quad & J(\boldsymbol{\theta}, \gamma) \\ \text{s.t.} \quad \boldsymbol{x}_{t+1} = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t, \boldsymbol{\theta}) \quad \text{with} \quad & \boldsymbol{x}_0 = \boldsymbol{x}_0(\boldsymbol{\theta}), \end{aligned} \tag{SB($\boldsymbol{\theta}, \gamma$)}$$

that is, $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}$ is minimizing the new control cost $J(\boldsymbol{\theta}, \gamma)$ subject to only dynamics. Then we have the following important result for the safe unconstrained approximation of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ using $\boldsymbol{\Sigma}(\boldsymbol{\theta}, \gamma)$.

Theorem 2. *Let the conditions (i)-(iii) in Lemma 1 for differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ to the constrained optimal control system $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ hold. For any small $\gamma > 0$,*

(a) there exists a local isolated minimizing trajectory $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}$ that solves Problem SB($\boldsymbol{\theta}, \gamma$), and $\boldsymbol{\Sigma}(\boldsymbol{\theta}, \gamma)$ is well-defined at $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}$, i.e., $\boldsymbol{g}_t\big(\boldsymbol{x}^{(\boldsymbol{\theta},\gamma)}_t, \boldsymbol{u}^{(\boldsymbol{\theta},\gamma)}_t, \boldsymbol{\theta}\big) < \boldsymbol{0}$ and $\boldsymbol{g}_T\big(\boldsymbol{x}^{(\boldsymbol{\theta},\gamma)}_T, \boldsymbol{\theta}\big) < \boldsymbol{0}$;

(b) $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}$ is once-continuously differentiable with respect to $(\boldsymbol{\theta}, \gamma)$, and

$$\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)} \to \boldsymbol{\xi}_{\boldsymbol{\theta}} \quad \textit{and} \quad \frac{\partial \boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}}{\partial \boldsymbol{\theta}} \to \frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}} \quad \textit{as} \quad \gamma \to 0; \tag{6}$$

(c) *the trajectory derivative* $\frac{\partial \boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}}{\partial \boldsymbol{\theta}}$ *for* $\boldsymbol{\Sigma}(\boldsymbol{\theta},\gamma)$ *is a unique global minimizing trajectory to the auxiliary control system* $\overline{\boldsymbol{\Sigma}}\big(\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}\big)$ *corresponding to* $\boldsymbol{\Sigma}(\boldsymbol{\theta},\gamma)$.

The proof of the above theorem is given in Appendix C. It is worth noting that the above assertions requires no additional assumption except the same conditions (i)-(iii) for differentiblity of $\boldsymbol{\xi}_{\boldsymbol{\theta}}$. We make the following comments on the above results, using an illustrative cartpole example in Fig. 1. First, assertion (b) states that by choosing a small $\gamma > 0$, one can simply use $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}$ and $\frac{\partial \boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}}{\partial \boldsymbol{\theta}}$ of the *unconstrained optimal control system* $\boldsymbol{\Sigma}(\boldsymbol{\theta},\gamma)$ to approximate $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ of the original constrained system $\boldsymbol{\Sigma}(\boldsymbol{\theta})$, respectively. Second, notably, assertion (b) also states that the above approximations can be controlled for arbitrary accuracy by simply letting $\gamma \to 0$, as illustrated in the upper panels in Fig. 1. Third, more importantly, assertion (a) states that the above approximations are always safe in a sense that the approximation $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}$ with any small $\gamma > 0$ is guaranteed to satisfy all inequality constraints in the original $\boldsymbol{\Sigma}(\boldsymbol{\theta})$, as illustrated in the bottom panels in Fig. 1. Finally, similar to the claims in Theorem 1, assertion (c) states that the derivative $\frac{\partial \boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}}{\partial \boldsymbol{\theta}}$ for $\boldsymbol{\Sigma}(\boldsymbol{\theta},\gamma)$ is a global minimizing trajectory to its corresponding auxiliary control system $\overline{\boldsymbol{\Sigma}}\big(\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}\big)$, so one can apply the unconstrained PDP [61].

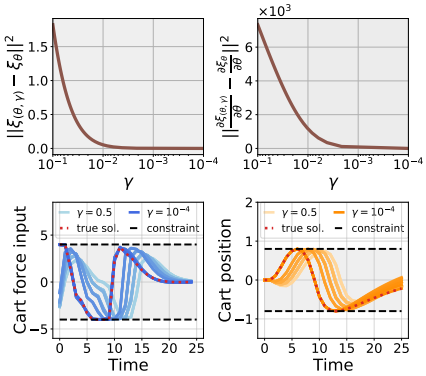


Figure 1: $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}$ and $\frac{\partial \boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}}{\partial \boldsymbol{\theta}}$ approximate $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ under different $\gamma > 0$.

In addition to the theoretical importance of Theorem 2, we also summarize its algorithmic advantage compared to directly handling the constrained system $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ and its auxiliary control system $\overline{\boldsymbol{\Sigma}}(\boldsymbol{\xi}_{\boldsymbol{\theta}})$. First, solving the unconstrained $\boldsymbol{\Sigma}(\boldsymbol{\theta},\gamma)$ is much easier than solving the constrained $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ as abundant trajectory optimization methods are available such as iLQR and DDP. Second, when applying Theorem 2 to approximate $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ using $\frac{\partial \boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}}{\partial \boldsymbol{\theta}}$, there is no need to identify the inactive and active inequality constraints, which can be difficult to implement due to numerical errors (see experiments later); Third, in contrast to Theorem 1, when using unconstrained $\boldsymbol{\Sigma}(\boldsymbol{\theta},\gamma)$ and $\overline{\boldsymbol{\Sigma}}\big(\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}\big)$, there is no need to deal with the multipliers $\boldsymbol{v}_{0:T}$ and $\boldsymbol{w}_{0:T}$. Finally, in $\boldsymbol{\Sigma}(\boldsymbol{\theta},\gamma)$, by absorbing the hard inequality constraints through barrier functions into the control cost, it introduces the 'softness' of constraints and eliminates the dis-continuous 'switch' between inactive/active inequalities over a large range of $\boldsymbol{\theta}$, leading to a more numerically stable algorithm, as we will show in experiments later. The algorithm that implements Theorem 2 is given in Algorithm 2 in Appendix E.2.

6 Safe PDP to Solve Problem P

To complete Safe PDP, we now give a result for Problem P to address Challenge 4) in Section 3.

Theorem 3. *Consider all functions defining the constrained optimal control system* $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ *are three-times continuously differentiable, and let the conditions (i)-(iii) in Lemma 1 for differentiblity of* $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ *hold in a neighborhood of* $\boldsymbol{\theta}^*$. *If the second-order condition that* $\boldsymbol{\theta}^*$ *is a local isolated minimizor to Problem P are satisfied, the gradients of all binding constraints* $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}^*}, \boldsymbol{\theta}^*)$=0, $\nabla_{\boldsymbol{\theta}} R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}^*}, \boldsymbol{\theta}^*)$, *are linearly independent, and the strict complementary holds at* $\boldsymbol{\theta}^*$, *then, the following assertions hold for any small* $\epsilon > 0$,

(a) *there exists a local isolated minimizor* $\boldsymbol{\theta}^*(\epsilon)$ *to Problem SP(*ϵ*), and the corresponding system trajectory* $\boldsymbol{\xi}_{\boldsymbol{\theta}^*(\epsilon)}$ *from* $\boldsymbol{\Sigma}(\boldsymbol{\theta}^*(\epsilon))$ *is safe, i.e.,* $R_i\big(\boldsymbol{\xi}_{\boldsymbol{\theta}^*(\epsilon)}, \boldsymbol{\theta}^*(\epsilon)\big) < 0$, $i = 1, 2, ..., l$;

(b) $\boldsymbol{\theta}^*(\epsilon)$ *is once-continuously differentiable with respect to* ϵ, *and*

$$\boldsymbol{\theta}^*(\epsilon) \to \boldsymbol{\theta}^* \qquad \text{as} \qquad \epsilon \to 0; \tag{7}$$

(c) *For any* $\boldsymbol{\theta}$ *near* $\boldsymbol{\theta}^*(\epsilon)$, *the trajectory* $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ *from* $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ *is safe, i.e.,* $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) < 0$, $i = 1, 2, ..., l$.

The proof of the above theorem is in Appendix D. The above result says that instead of solving the constrained Problem P, one can absorb all $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) \leq 0$ into the task loss through barrier functions and solve an unconstrained Problem SP(ϵ). Importantly, by controlling the barrier parameter $\epsilon \to 0$, the solution $\boldsymbol{\theta}^*(\epsilon)$ to the unconstrained Problem SP(ϵ) can converge to the true solution $\boldsymbol{\theta}^*$ to the

original Problem P. In Theorem 3, the most crucial are the following two safety assertions: first, from (a), the solution $\boldsymbol{\theta}^*(\epsilon)$ to Problem SP(ϵ) is always safe in a sense that the corresponding trajectory $\boldsymbol{\xi}_{\boldsymbol{\theta}^*(\epsilon)}$ satisfies $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}^*(\epsilon)}, \boldsymbol{\theta}^*(\epsilon))<0$, $i = 1, 2, ..., l$; and second, from (c), for any $\boldsymbol{\theta}$ near $\boldsymbol{\theta}^*(\epsilon)$, the corresponding trajectory $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ is also safe, i.e., $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) < 0$, $i = 1, 2, ..., l$. These two safety assertions combined give an ultimate safety assertion for Safe PDP: *when searching for the solution $\boldsymbol{\theta}^*$ to Problem P in its neighborhood by sequentially solving Problem SP(ϵ) using any gradient-based solvers, any intermediate $\boldsymbol{\theta}$ is safe.* The Safe PDP algorithm is given in Algorithm 3 in Appendix E.3.

7 Applications

We apply Safe PDP to solve some representative safety-critical learning/control tasks. For a specific task, one only needs to specify the parameterization details of $\boldsymbol{\Sigma}(\boldsymbol{\theta})$, task loss $\ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$, and constraints $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ in Problem P. The experiments are performed on the systems of different complexities in Table 1. All codes are available at `Blank_URL_for_Anonymity`.

Table 1: Experimental environments [61]

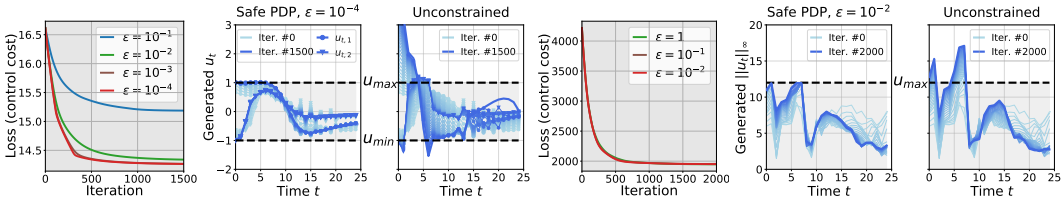
System $\boldsymbol{\Sigma}(\boldsymbol{\theta})$	Dynamics $\boldsymbol{f}(\boldsymbol{\theta}_{\text{dyn}})$	Control cost $J(\boldsymbol{\theta}_{\text{obj}})$	Constraints $\boldsymbol{g}(\boldsymbol{\theta}_{\text{cstr}})$
Cartpole	cart & pole masses and length	$c_t = \lVert\boldsymbol{u}\rVert_2^2 + \lVert\boldsymbol{\theta}'_{\text{obj}}(\boldsymbol{x}-\boldsymbol{x}_{\text{goal}})\rVert_2^2$, $c_T = \lVert\boldsymbol{\theta}'_{\text{obj}}(\boldsymbol{x}-\boldsymbol{x}_{\text{goal}})\rVert_2^2$	$\boldsymbol{g}_x(\boldsymbol{x}) \leq X_{\max}$, $\lVert\boldsymbol{u}\rVert_{2/\infty} \leq U_{\max}$, $\boldsymbol{\theta}_{\text{cstr}} = \{X_{\max}, U_{\max}\}$
Two-link Robot arm	length and mass of links		
6-DoF quadrotor	mass, wing length, inertia		
6-DoF rocket landing	rocket mass, length, inertia		

Note that for each system, $\boldsymbol{g}(\boldsymbol{\theta}_{\text{cstr}})$ includes the immediate constraints on both system input $\boldsymbol{u}$ and state $\boldsymbol{x}$ at any time instance; $\boldsymbol{g}_x$ is known; $\lVert\cdot\rVert_{2/\infty}$ is the 2 or ∞ norm; and time horizon T for all systems is around 50.

Problem I: Safe Policy Optimization aims to find a policy that minimizes a control cost J subject to constraints $\boldsymbol{g}$, while guaranteeing that *any intermediate policy during optimization should never violate the constraints*. To apply Safe PDP to solve such a problem for the systems in Table 1, we set:

$$\boldsymbol{\Sigma}(\boldsymbol{\theta}): \quad \begin{aligned} &\textit{dynamics:} \quad \boldsymbol{x}_{t+1} = \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t) \quad \text{with} \quad \boldsymbol{x}_0, \\ &\textit{policy:} \quad \boldsymbol{u}_t = \boldsymbol{\pi}_t(\boldsymbol{x}_t, \boldsymbol{\theta}), \end{aligned} \tag{8}$$

where dynamics $\boldsymbol{f}$ is learned from demonstrations in Problem III, and $\boldsymbol{\pi}(\boldsymbol{\theta})$ is represented by a (deep) feedforward neural network (NN) with $\boldsymbol{\theta}$ the NN parameter. In Problem P, the task loss $\ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ is set as $J(\boldsymbol{\theta}_{\text{obj}})$, and task constraints $R_i(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta})$ as $\boldsymbol{g}(\boldsymbol{\theta}_{\text{cstr}})$, with both $\boldsymbol{\theta}_{\text{obj}}$ and $\boldsymbol{\theta}_{\text{cstr}}$ known. Then, safe policy optimization is to solve Problem P using Safe PDP. The results for the robot arm and 6-DoF maneuvering quadrotor are in Fig. 2, and the other results, details, and videos are in Appendix F.1.



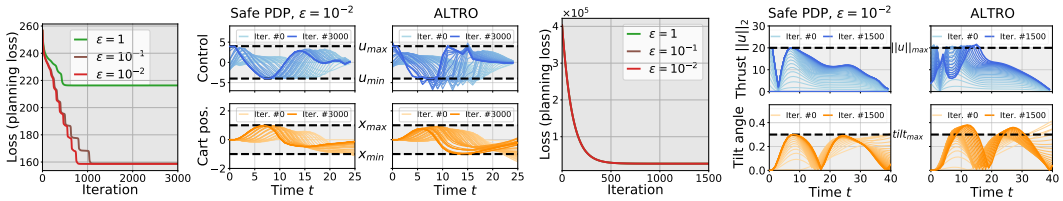
(a) Robot-arm loss (b) Constraint violation during opt. (c) Quadrotor loss (d) Constraint violation during opt.

Figure 2: Safe policy optimization for robot-arm (a)-(b) and 6-DoF maneuvering quadrotor (c)-(d).

Fig. 2a and 2c plot loss (control cost) versus gradient-descent iteration under different ϵ, showing that the NN policy achieves a good convergence when $\epsilon \leq 10^{-2}$ (as asserted by Theorem 3). Fig. 2b and 2d show all indeterminate control trajectories generated from the NN policy during entire iterations; we also mark the constraints $U_{\max}$ and compare with the unconstrained policy optimization under the same conditions. The results confirm that Safe PDP enables *to achieve an optimal policy while guaranteeing that any intermediate policy throughout optimization is safe*. Note that since exiting policy optimization methods (to authors' best knowledge, described in Section 1.1) cannot deal with immediate state and control constraints in this experiment, we have not given any other comparison.

Problem II: Safe Motion Planning searches for a dynamics-feasible trajectory that optimizes a criterion and avoids unsafe regions (obstacles), while guaranteeing that any intermediate trajectory during search must avoid the unsafe regions. To apply Safe PDP to solve such problem, we specialize $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ as (8) except that policy here is $\boldsymbol{u}_t = \boldsymbol{u}(t, \boldsymbol{\theta})$, which is represented by Lagrangian polynomial [75] with $\boldsymbol{\theta}$ the parameters (pivots). In Problem P, task loss is set as $J(\boldsymbol{\theta}_{\text{obj}})$, and task constraints as

$\boldsymbol{g}(\boldsymbol{\theta}_{\text{cstr}})$, with $\boldsymbol{\theta}_{\text{obj}}$ and $\boldsymbol{\theta}_{\text{cstr}}$ known in Table 1. The safe planning results using Safe PDP for cartpole and 6-DoF rocket landing are in Fig. 2, in comparison with ALTRO, a state-of-the-art constrained trajectory optimization method [21]. Other results and more details (videos) are in Appendix F.2.



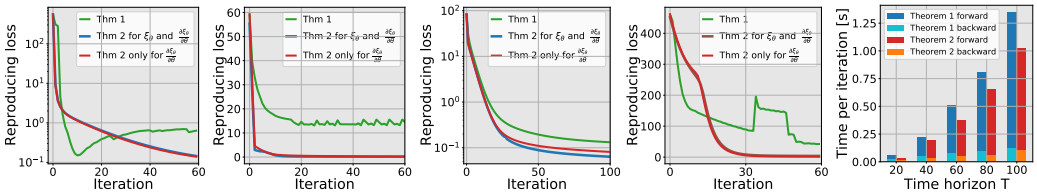
(a) Cartpole loss (b) Constraint violation during opt. (c) Rocket loss (d) Constraint violation during opt.

Figure 3: Safe motion planning for cartpole (a)-(b) and 6-DoF rocket powered landing (c)-(d).

Fig. 3a and 3c plot the task loss versus gradient-descent iteration, showing that the trajectory achieves a good convergence with $\epsilon \leq 10^{-2}$. Fig. 3b and 3d show all intermediate trajectories during entire iterations of Safe PDP and those of ALTRO, with constraints marked. The results confirms that Safe PDP can find an optimal trajectory while always respecting constraints throughout planning process.

Problem III: Learning MPC from Demonstrations. Suppose for all systems in Table 1, the control cost $J(\boldsymbol{\theta}_{\text{cost}})$, dynamics $\boldsymbol{f}(\boldsymbol{\theta}_{\text{dyn}})$ and constraints $\boldsymbol{g}_t(\boldsymbol{\theta}_{\text{cstr}})$ are all unknown and parameterized as in Table 1. We aim to jointly learn $\boldsymbol{\theta} = \{\boldsymbol{\theta}_{\text{cost}}, \boldsymbol{\theta}_{\text{dyn}}, \boldsymbol{\theta}_{\text{cstr}}\}$ from demonstrations $\boldsymbol{\xi}^{\text{demo}} = \{\boldsymbol{x}^{\text{demo}}_{0:T}, \boldsymbol{u}^{\text{demo}}_{0:T-1}\}$ of a true system/expert. In Problem P, set $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ as (1), consisting of $J(\boldsymbol{\theta}_{\text{cost}})$, $\boldsymbol{f}(\boldsymbol{\theta}_{\text{dyn}})$, and $\boldsymbol{g}_t(\boldsymbol{\theta}_{\text{cstr}})$ parameterized; set task loss $\ell(\boldsymbol{\xi}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) = \|\boldsymbol{\xi}^{\text{demo}} - \boldsymbol{\xi}_{\boldsymbol{\theta}}\|^2$, which quantifies the *reproducing loss* between $\boldsymbol{\xi}^{\text{demo}}$ and $\boldsymbol{\xi}_{\boldsymbol{\theta}}$; and there is no task constraints. By solving Problem P, we can learn $\boldsymbol{\Sigma}(\boldsymbol{\theta})$ such that its reproduced $\boldsymbol{\xi}_{\theta}$ is closest to given $\boldsymbol{\xi}^{\text{demo}}$. The demonstrations $\boldsymbol{\xi}^{\text{demo}}$ here are generated with $\boldsymbol{\theta}$ known (two episode trajectories for each system with time horizon T=50). The results of reproducing loss versus gradient-descent iteration are in Fig. 4, and more details/results (videos) are in Appendix F.3.

In Fig. 4a-4d, for each system, we use three strategies to obtain $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ for $\boldsymbol{\Sigma}(\boldsymbol{\theta})$: (A) use a solver [69] to obtain $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and use Theorem 1 to obtain $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$; (B) use Theorem 2 to approximate both $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$ by $\boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}$ and $\frac{\partial \boldsymbol{\xi}_{(\boldsymbol{\theta},\gamma)}}{\partial \boldsymbol{\theta}}$, respectively, γ=10^{-2}; and (C) use a solver to obtain $\boldsymbol{\xi}_{\boldsymbol{\theta}}$ and Theorem 2 only for $\frac{\partial \boldsymbol{\xi}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}$. Fig. 4a-4d show that for Strategies (B) and (C), the reproducing loss quickly converges to zeros, indicating that the dynamics, constraints, and control cost are successfully learned to reproduce the demonstrations. Fig. 4a-4d also show numerical instability of strategy (A); this is due to the discontinuous 'jumping switch' of inactive/active inequalities between iterations, and error of correctly identifying active inequalities (we identify them by checking $g_{t,i} > -\delta$ with δ a small threshold), as analyzed in Section 5. More analysis is given in Appendix F.3. Note that we are not aware of any existing methods that can handle jointly learning of cost, dynamics, and constraints here, and thus we have not given benchmark comparison. Fig. 4e gives timing results of Safe PDP.



(a) Cartpole (log-y) (b) Robot-arm (c) Quadrotor (log-y) (d) Rocket (e) Timing

Figure 4: Jointly learning dynamics, constraints, and control cost from demonstrations.

8 Conclusions

This paper develops a Safe Pontryagin Differentiable Programming methodology, which establishes a systematic and provable safe differentiable framework to solve a board class of safety-critical control and learning tasks with different types of constraints. Safe PDP approximates both the constrained solution and its gradient in backward pass using a more efficient unconstrained counterpart. We have established the results for the controlled accuracy of the approximations and the guarantee of strict constraint satisfaction throughout the approximation and optimization. We envision the potential of Safe PDP in addressing various safety-critical problems in learning, control, and robotics fields.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[2] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[3] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *International Conference on Learning Representations*, 2016.

[4] Arnab Nilim and Laurent El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

[5] Matthias Heger. Consideration of risk in reinforcement learning. In *International Conference on Machine Learning*, pages 105–111. 1994.

[6] Ronald A Howard and James E Matheson. Risk-sensitive markov decision processes. *Management science*, 18(7):356–369, 1972.

[7] Teodor Mihai Moldovan and Pieter Abbeel. Safe exploration in markov decision processes. *International Conference on Machine Learning*, 2012.

[8] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[9] Kemin Zhou and John Comstock Doyle. *Essentials of robust control*, volume 104. Prentice hall Upper Saddle River, NJ, 1998.

[10] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[11] Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

[12] Eitan Altman. Constrained markov decision processes with total cost criteria: Lagrangian approach and dual linear program. *Mathematical methods of operations research*, 48(3):387–417, 1998.

[13] Ming Yu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. Convergent policy optimization for safe reinforcement learning. *Advances in Neural Information Processing Systems*, 2019.

[14] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *International Conference on Machine Learning*, 18(1):6070–6120, 2017.

[15] Shalabh Bhatnagar and K Lakshmanan. An online actor–critic algorithm with function approximation for constrained markov decision processes. *Journal of Optimization Theory and Applications*, 153(3):688–708, 2012.

[16] Dongsheng Ding, Kaiqing Zhang, Tamer Basar, and Mihailo Jovanovic. Natural policy gradient primal-dual method for constrained markov decision processes. *Advances in Neural Information Processing Systems*, 33, 2020.

[17] Miguel Calvo-Fullana, Santiago Paternain, Luiz FO Chamon, and Alejandro Ribeiro. State augmented constrained reinforcement learning: Overcoming the limitations of learning with rewards. *arXiv preprint arXiv:2102.11941*, 2021.

[18] Yatin Nandwani, Abhishek Pathak, Parag Singla, et al. A primal dual formulation for deep learning with constraints. *Advances in Neural Information Processing Systems*, 2019.

[19] Maïtine Bergounioux, Kazufumi Ito, and Karl Kunisch. Primal-dual strategy for constrained optimal control problems. *SIAM Journal on Control and Optimization*, 37(4):1176–1194, 1999.

[20] Matthew R Kirchner, Gary Hewer, Jérôme Darbon, and Stanley Osher. A primal-dual method for optimal control and trajectory generation in high-dimensional systems. In *Conference on Control Technology and Applications*, pages 1583–1590, 2018.

[21] Taylor A Howell, Brian E Jackson, and Zachary Manchester. Altro: A fast solver for constrained trajectory optimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 7674–7679, 2019.

[22] Simon S Du and Wei Hu. Linear convergence of the primal-dual gradient method for convex-concave saddle point problems without strong convexity. In *International Conference on Artificial Intelligence and Statistics*, pages 196–205. PMLR, 2019.

[23] Chi Jin, Praneeth Netrapalli, and Michael Jordan. What is local optimality in nonconvex-nonconcave minimax optimization? In *International Conference on Machine Learning*, pages 4880–4889. PMLR, 2020.

[24] Yinlam Chow, Ofir Nachum, Aleksandra Faust, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1901.10031*, 2019.

[25] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International Conference on Machine Learning*, pages 22–31. PMLR, 2017.

[26] Tsung-Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge. Projection-based constrained policy optimization. *International Conference on Learning Representations*, 2020.

[27] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897. PMLR, 2015.

[28] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. *Advances in Neural Information Processing Systems*, 2018.

[29] Theodore J Perkins and Andrew G Barto. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research*, 3(Dec):803–832, 2002.

[30] Felix Berkenkamp, Matteo Turchetta, Angela P Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. *Advances in Neural Information Processing Systems*, 2017.

[31] Aleksandr Mikhailovich Lyapunov. The general problem of the stability of motion. *International journal of control*, 55(3):531–534, 1992.

[32] Peter Giesl and Sigurdur Hafstein. Review on computational methods for lyapunov functions. *Discrete & Continuous Dynamical Systems-B*, 20(8):2291, 2015.

[33] Somil Bansal, Mo Chen, Sylvia Herbert, and Claire J Tomlin. Hamilton-jacobi reachability: A brief overview and recent advances. In *IEEE Conference on Decision and Control*, pages 2242–2253, 2017.

[34] Peter Wieland and Frank Allgöwer. Constructive safety using control barrier functions. *IFAC Proceedings Volumes*, 40(12):462–467, 2007.

[35] Aaron D Ames, Xiangru Xu, Jessy W Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2016.

[36] Jaime F Fisac, Anayo K Akametalu, Melanie N Zeilinger, Shahab Kaynama, Jeremy Gillula, and Claire J Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 64(7):2737–2752, 2018.

[37] Sylvia L Herbert, Mo Chen, SooJean Han, Somil Bansal, Jaime F Fisac, and Claire J Tomlin. Fastrack: A modular framework for fast and guaranteed safe motion planning. In *IEEE Annual Conference on Decision and Control*, pages 1517–1522, 2017.

[38] Mo Chen, Jaime F Fisac, Shankar Sastry, and Claire J Tomlin. Safe sequential path planning of multi-vehicle systems via double-obstacle hamilton-jacobi-isaacs variational inequality. In *European Control Conference*, pages 3304–3309, 2015.

[39] Lawrence C Evans and Panagiotis E Souganidis. Differential games and representation formulas for solutions of hamilton-jacobi-isaacs equations. *Indiana University mathematics journal*, 33(5):773–797, 1984.

[40] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *European Control Conference*, pages 3420–3431, 2019.

[41] Jason Choi, Fernando Castaneda, Claire J Tomlin, and Koushil Sreenath. Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions. *arXiv preprint arXiv:2004.07584*, 2020.

[42] Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *AAAI Conference on Artificial Intelligence*, volume 33, pages 3387–3395, 2019.

[43] Alexander Robey, Haimin Hu, Lars Lindemann, Hanwen Zhang, Dimos V Dimarogonas, Stephen Tu, and Nikolai Matni. Learning control barrier functions from expert demonstrations. In *IEEE Conference on Decision and Control*, pages 3717–3724, 2020.

[44] Alexander Robey, Lars Lindemann, Stephen Tu, and Nikolai Matni. Learning robust hybrid control barrier functions for uncertain systems. *arXiv preprint arXiv:2101.06492*, 2021.

[45] Wanxin Jin, Zhaoran Wang, Zhuoran Yang, and Shaoshuai Mou. Neural certificates for safe control policies. *arXiv preprint arXiv:2006.08465*, 2020.

[46] Anthony V Fiacco and Garth P McCormick. *Nonlinear programming: sequential unconstrained minimization techniques*. SIAM, 1990.

[47] Yurii Nesterov and Arkadii Nemirovskii. *Interior-point polynomial algorithms in convex programming*. SIAM, 1994.

[48] Anders Forsgren, Philip E Gill, and Margaret H Wright. Interior methods for nonlinear optimization. *SIAM review*, 44(4):525–597, 2002.

[49] AEB Lim, JB Moore, and L Faybusovich. Linearly constrained lq and lqg optimal control. *IFAC Proceedings Volumes*, 29(1):1110–1115, 1996.

[50] SJ Wright. Structured interior point methods for optimal control. In *IEEE Conference on Decision and Control*, pages 1711–1716, 1991.

[51] Stephen J Wright. Interior point methods for optimal control of discrete time systems. *Journal of Optimization Theory and Applications*, 77(1):161–187, 1993.

[52] Christopher V Rao, Stephen J Wright, and James B Rawlings. Application of interior-point methods to model predictive control. *Journal of optimization theory and applications*, 99(3):723–757, 1998.

[53] Anders Hansson and S Boydt. Robust optimal control of linear discrete-time systems using primal-dual interior-point methods. In *American Control Conference*, volume 1, pages 183–187, 1998.

[54] Anders Hansson. A primal-dual interior-point method for robust optimal control of linear discrete-time systems. *IEEE Transactions on Automatic Control*, 45(9):1639–1655, 2000.

[55] Christian Feller and Christian Ebenbauer. Relaxed logarithmic barrier function based model predictive control of linear systems. *IEEE Transactions on Automatic Control*, 62(3):1223–1238, 2016.

[56] Julien Laurent-Varin, J Frederic Bonnans, Nicolas Bérend, Mounir Haddou, and Christophe Talbot. Interior-point approach to trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 30(5):1228–1238, 2007.

[57] John Hauser and Alessandro Saccon. A barrier function method for the optimization of trajectory functionals with constraints. In *IEEE Conference on Decision and Control*, pages 864–869, 2006.

[58] Paul Malisani, François Chaplais, and Nicolas Petit. An interior penalty method for optimal control problems with state and input constraints of nonlinear systems. *Optimal Control Applications and Methods*, 37(1):3–33, 2016.

[59] Alexander Domahidi, Aldo U Zgraggen, Melanie N Zeilinger, Manfred Morari, and Colin N Jones. Efficient interior point methods for multistage problems arising in receding horizon control. In *IEEE conference on decision and control*, pages 668–674, 2012.

[60] Andrei Pavlov, Iman Shames, and Chris Manzie. Interior point differential dynamic programming. *IEEE Transactions on Control Systems Technology*, 2021.

[61] Wanxin Jin, Zhaoran Wang, Zhuoran Yang, and Shaoshuai Mou. Pontryagin differentiable programming: An end-to-end learning and control framework. In *Advances in Neural Information Processing Systems*, 2020.

[62] Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. A review on bilevel optimization: from classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2017.

[63] J Pearson and R Sridhar. A discrete optimal control problem. *IEEE Transactions on automatic control*, 11(2):171–174, 1966.

[64] Anthony V Fiacco. Sensitivity analysis for nonlinear programming using penalty methods. *Mathematical programming*, 10(1):287–311, 1976.

[65] Walter Rudin et al. *Principles of mathematical analysis*, volume 3. McGraw-hill New York, 1976.

[66] Charles D Kolstad and Leon S Lasdon. Derivative evaluation and computational experience with large bilevel mathematical programs. *Journal of optimization theory and applications*, 65(3):485–499, 1990.

[67] Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pages 136–145. PMLR, 2017.

[68] Brandon Amos, Ivan Dario Jimenez Rodriguez, Jacob Sacks, Byron Boots, and J Zico Kolter. Differentiable mpc for end-to-end planning and control. *arXiv preprint arXiv:1810.13400*, 2018.

[69] Joel AE Andersson, Joris Gillis, Greg Horn, James B Rawlings, and Moritz Diehl. Casadi: a software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1):1–36, 2019.

[70] Athanasios Sideris and Luis A Rodriguez. A riccati approach to equality constrained linear quadratic optimal control. In *American Control Conference*, pages 5167–5172, 2010.

[71] Shuo Yang, Gerry Chen, Yetong Zhang, Frank Dellaert, and Howie Choset. Equality constrained linear optimal control with factor graphs. *arXiv preprint arXiv:2011.01360*, 2020.

[72] Forrest Laine and Claire Tomlin. Efficient computation of feedback control for equality-constrained lqr. In *International Conference on Robotics and Automation*, pages 6748–6754. IEEE, 2019.

[73] Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *International Conference on Informatics in Control, Automation and Robotics*, pages 222–229, 2004.

[74] David H Jacobson and David Q Mayne. Differential dynamic programming. 1970.

[75] Milton Abramowitz and Irene A Stegun. *Handbook of mathematical functions with formulas, graphs, and mathematical tables*, volume 55. US Government printing office, 1964.

[76] Michael Athans. The matrix minimum principle. *Information and control*, 11(5-6):592–606, 1967.

[77] Jean Dieudonné. *Foundations of modern analysis*. New York: Academic Press. Volume 1 of Treatise on Analysis, 2011.

[78] Rolf Johansson. *System modeling and identification*. Prentice-hall, 1993.

[79] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 2018.

[80] Wanxin Jin, Dana Kulić, Shaoshuai Mou, and Sandra Hirche. Inverse optimal control from incomplete trajectory observations. *The International Journal of Robotics Research*, 40(6-7):848–865, 2021.

[81] Wanxin Jin and Shaoshuai Mou. Distributed inverse optimal control. *Automatica*, 129, 2021.

[82] Wanxin Jin, Dana Kulić, Jonathan Feng-Shun Lin, Shaoshuai Mou, and Sandra Hirche. Inverse optimal control for multiphase cost functions. *IEEE Transactions on Robotics*, 35(6):1387–1398, 2019.

[83] Wanxin Jin, Todd D Murphey, and Shaoshuai Mou. Learning from incremental directional corrections. *arXiv preprint arXiv:2011.15014*, 2020.

[84] Wanxin Jin, Todd D Murphey, Dana Kulić, Neta Ezer, and Shaoshuai Mou. Learning from sparse demonstrations. *arXiv preprint arXiv:2008.02159*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Please see Sections 5 and 7 in the paper
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] Pleae find them in all Theorems in the paper.
   (b) Did you include complete proofs of all theoretical results? [Yes] Please find complete proofs for all theoretical results in the Appendix in the supplementary file.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] currently all codes are provided on request. We will release to public in proper time very soon.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please see the Appendix Section F in the supplementary file.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please see the Appendix Section F in the supplementary file.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

   (a) If your work uses existing assets, did you cite the creators? [Yes] Please see the citation of the experimental environments in Table 1 in the paper.

   (b) Did you mention the license of the assets? [N/A]

   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] Some video demos are included in the supplementary file.

   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]