UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Hamilton-Jacobi Reachability Estimation in Reinforcement Learning

Permalink

https://escholarship.org/uc/item/75d2j8bj

Author

Ganai, Milan

Publication Date

2024

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA SAN DIEGO

Hamilton-Jacobi Reachability Estimation in Reinforcement Learning

A Thesis submitted in partial satisfaction of the
requirements for the degree Master of Science

in

Computer Science

by

Milan Ganai

Committee in charge:

Professor Sicun Gao, Chair
Professor Henrik I Christensen
Professor Sylvia Lee Herbert

2024

Copyright

Milan Ganai, 2024

All rights reserved.

The Thesis of Milan Ganai is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

DEDICATION

To my parents.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to acknowledge Professor Sicun Gao for his support as a research advisor throughout my undergraduate and graduate studies at UC San Diego. I am thankful for the opportunity he provided me to enter into the adventure of research in AI, robotics, and decision making and the in-depth discussions in solving a host of research problem.

I would like to acknowledge Professor Sylvia Herbert for her guidance in chartering the domain of control theory and Hamilton-Jacobi Reachability as well as her expertise and insights in reviewing my publications. I also would like to acknowledge Professor Henrik Christensen for reviewing my thesis.

I would like to thank Zheng Gong and Chenning Yu for the long hours in brainstorming and doing awesome research. I also would like to thank the Automation Algorithms Lab for the fun travels, trying new food, and learning a lot.

Finally, I am deeply grateful to my parents for their endless love and support and for always encouraging me.

Chapters 1, 2, 3, and 5 have been submitted for publication of the material in "Hamilton-Jacobi Reachability in Reinforcement Learning: A Survey," M. Ganai; S. Gao; S. Herbert, 2024. The thesis author was the primary investigator and author of this paper.

Chapter 4, in full, is a reprint of the material as it appears in "Iterative Reachability Estimation for Safe Reinforcement Learning," M. Ganai, Z. Gong, C. Yu, S. Herbert, S. Gao. in Advances in Neural Information Processing Systems, 2023. The thesis author is the primary investigator and author of this paper.

VITA

2023 Bachelor of Science in Computer Science, University of California San Diego

2024 Master of Science in Computer Science, University of California San Diego

PUBLICATIONS

Milan Ganai, Chiaki Hirayama, Ya-Chien Chang, and Sicun Gao. Learning Stabilization Control from Observations by Learning Lyapunov-like Proxy Models. 2023 IEEE International Conference on Robotics and Automation (ICRA), pages 2913–2920, 2023.

Milan Ganai, Zheng Gong, Chenning Yu, Sylvia L Herbert, and Sicun Gao. Iterative Reachability Estimation for Safe Reinforcement Learning. In Advances in Neural Information Processing Systems, 2023.

Milan Ganai, Sylvia L Herbert, and Sicun Gao. Hamilton-Jacobi Reachability in Reinforcement Learning: A Survey. 2024.

ABSTRACT OF THE THESIS

Hamilton-Jacobi Reachability Estimation in Reinforcement Learning

by

Milan Ganai

Master of Science in Computer Science

University of California San Diego, 2024

Professor Sicun Gao, Chair

Recent literature has proposed approaches that learn control policies with high performance while maintaining safety guarantees. Synthesizing Hamilton-Jacobi (HJ) reachable sets has become an effective tool for verifying safety and supervising the training of reinforcement learning-based control policies for complex, high-dimensional systems. Previously, HJ reachability was limited to verifying low-dimensional dynamical systems – this is because the computational complexity of the dynamic programming approach it relied on grows exponentially with the number of system states. To address this limitation, in recent years, there have been methods that compute the reachability value function simultaneously with learning control policies to scale HJ reachability analysis while still maintaining a reliable estimate of

the true reachable set. These HJ reachability approximations are used to improve the safety, and even reward performance, of reinforcement learning (RL) based control policies and can solve challenging tasks such as those with dynamic obstacles and/or with lidar-based or vision-based observations. We first introduce the framework for HJ reachability estimation in reinforcement learning. Then, we review the recent developments in the field of HJ reachability estimation research for reliability in high-dimensional systems. Subsequently, we present a new framework called Reachability Estimation for Safe Policy Optimization that employs HJ reachability estimation for stochastic safety-constrained reinforcement learning and provide safety guarantees and optimal convergence analysis.

Chapter 1

Introduction

As autonomous control systems are deployed in the real world, there is a growing need to develop methods with rigorous safety guarantees. Verification-based approaches relying on control theoretic functions have been in the forefront among studied solutions. However, the large uncertainty and complex nature of real world dynamics limits the practical application of many of these approaches.

Hamilton-Jacobi (HJ) reachability analysis is a rigorous tool that verifies the safety and/or liveness of a dynamic system [8, 24]. For a specified model and target set, HJ reachability analysis is typically used to compute the set of initial states from which the system can reach a goal despite bounded disturbance. For safety analysis, HJ reachability can provide the set of initial states from which the system may be forced into the failure set despite best-case efforts (the complement of this set of initial states is, therefore, the safe set). This verification method provides guarantees on the safety properties of a system and the approach generalizes to various difficult problem settings. These include problems with nonlinear dynamics, reach-avoid problems with time-varying goals or constraints [41], problems that must be robust to bounded system uncertainties or disturbances [20, 23], and finding other certificate functions [46].

HJ reachability computation is based on finding a viscosity solution for the Hamilton-Jacobi-Bellman partial differential equation (HJBPDE) corresponding to a specified dynamics model and target set. Proposed approaches have accomplished this by discretizing the state

space and using dynamic programming mechanisms [10]. However, this approach has been practically deployed on systems with at most 6 dimensions [22]. The main challenge is that the computational complexity of these approaches is exponential in the state dimensions, rendering them intractable in relatively large dimension systems.

To address this issue on the curse of dimensionality, past works have proposed approaches that make strong assumptions such as convexity, order preserving dynamics, and mixed monotone systems [33, 34, 50] or exploit the system's structure [21, 41, 58, 59, 72, 74]. However, these approaches still do not necessarily scale well with the complexity encountered in the learning-based controls. Furthermore, they still require access to the model for active sampling and/or computation of gradients of the dynamics.

In this thesis, we focus on a recent line of work that learns the HJ reachability value function in conjunction with learning control policies. Particularly, recent approaches like [4, 42] demonstrated how to learn a discrete-time value function solution of the HJBPDE via a recursive Bellman formulation. These value functions describe the maximum reachability violation or reward (depending on the usage) that a particular control policy achieves from each state. This form of learning has opened a new direction of research in which the learned reachability value function can directly be incorporated in reach-avoid problems [56] and safety-constrained reinforcement learning [43,99]. While learning a certificate has been implemented for other safety verification functions (e.g. control barrier functions), significant benefits of learning reachability value functions include a) the ability to guarantee convergence to a valid solution of the HJBPDE of a particular control policies' dynamics, and b) not having to perform hyperparameter tuning for the loss function. Learned reachability value functions for learned control policies have been demonstrated to be effective in various challenging problems [42, 43, 54, 56, 99].

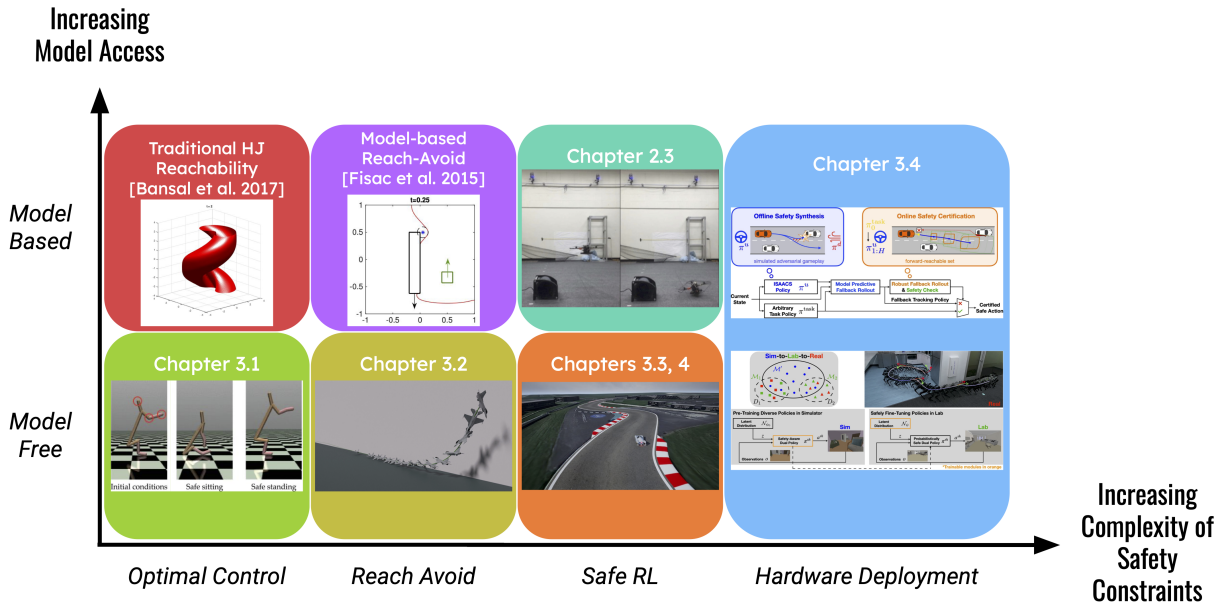

Figure 1.1. A layout of this thesis on approaches using HJ reachability for learning-based controls. In Chapters 2 and 3 we review background and survey papers in HJ reachability estimation. In Chapter 4 we present a novel Reachability Estimation for Safe Policy Optimization algorithm.

1.1 Motivations and Overview

While there are several recent surveys on related topics, none discuss the rapidly growing literature on HJ reachability for learned controls. Bansal et al.'s 2017 survey [8] reviews HJ reachability methods for high-dimensional reachability analysis (examples shown up to 10D) and includes a brief discussion on reachability analysis that use neural networks to solve HJBPDEs. Nonetheless, the approaches presented in the survey may not necessarily scale to the complexity encountered in systems controlled primarily with learned-based policies (>20D). Chen et al. 2018 [24] presents approaches to scale HJ reachability verification through system decomposition of nonlinear dynamics and applications in unmanned airspace management, but does not discuss learning-based HJ reachability techniques. The 2021 survey by Althoff et al. [6] covers methods that find a guaranteed overapproximation of the reachability set via set propagation; however, it leaves to future work HJ reachability methods for online verification of partially known environments, as well as systems involving neural networks. The recent survey by Dawson et

al. [35] covers topics on neural certificates – this class includes learning-based Lyapunov and Barrier functions [16, 17, 44, 76]. In this review we aim to provide an overview of estimating (i.e. via learning) HJ reachability specifically for learned controls. A schematic of the classes of methods we discuss in this thesis can be seen in Fig. 1.1. We structure this thesis in the following manner:

- In Chapter 2, we formally introduce reinforcement learning and HJ reachability analysis and discuss approaches that use traditional HJ reachability for learned control.
- In Chapter 3, we survey the recent progress made in learning-based HJ reachability estimation.
- In Chapter 4, we present a novel Reachability Estimation for Safe Policy Optimization algorithm for stochastic safety-constrained reinforcement learning with safety guarantees and optimal convergence analysis.
- In Chapter 5, we discuss the limitations of HJ reachability estimation approaches and lay out new research directions for future works in using HJ reachability estimation.

1.2 Acknowledgement

Chapter 1 has been submitted for publication of the material in "Hamilton-Jacobi Reachability in Reinforcement Learning: A Survey," M. Ganai; S. Gao; S. Herbert, 2024. The thesis author was the primary investigator and author of this paper.

Chapter 2

Background

2.1 Markov Decision Processes

A Markov decision process (MDP) is defined as $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, P, r, \gamma \rangle$, where

- $\mathcal{S} \subseteq \mathbb{R}^n$ and $\mathcal{A} \subseteq \mathbb{R}^{m_a}$ are the state and action spaces respectively,
- $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0,1]$ is the transition function capturing the environment dynamics,
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function associated with each state-action pair,
- γ is a discount factor in the range $[0,1)$,
- $\mathcal{S}_I \subseteq \mathcal{S}$ is the initial state set,
- $\Delta_0 : \mathcal{S}_I \to (0,1]$ is the initial state distribution, and
- $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$ is a stochastic policy that is a distribution capturing an action distribution given a state. Actions are sampled from this policy and affect the environment defined by the MDP.

In unconstrained RL, the goal is to learn an optimal policy π^* maximizing expected discounted sum of rewards, i.e.

$$\pi^* = \arg\max_{\pi} \mathop{\mathbb{E}}_{s\sim\Delta_0} V_r^{\pi}(s), \text{where} \tag{2.1}$$

$$V_r^{\pi}(s) := \mathop{\mathbb{E}}_{\xi\sim\pi,P(s)}[\sum_{s_t\in\xi} \gamma^t r(s_t,a_t)]. \tag{2.2}$$

Note: $\xi \sim \pi, P(s)$ indicates sampling trajectory ξ for horizon T starting from state s using policy π in the MDP with transition model P, and $s_t \in \xi$ is the t^{th} state in trajectory ξ. Similarly, $s' \sim \pi, P(s)$ indicates sampling the next state after state s using policy π with transition model P. We will use the notation s' to mean by default the next (sampled) state after the state s.

2.2 Dynamical Systems and HJ Reachability

In this thesis, we will consider continuous, fully observable dynamics that are either deterministic or stochastic with bounds. Consider a dynamical system $f : \mathcal{S} \times \mathcal{A} \times \mathcal{D} \to S$:

$$\frac{ds}{dt} = f(s,a,d) \tag{2.3}$$

in which the state is $s \in \mathcal{S} \subseteq \mathbb{R}^n$, the control (also known as action) is $a \in \mathcal{A}$, and the disturbance is $d \in \mathcal{D}$, where $\mathcal{A} \subseteq \mathbb{R}^{m_a}$ and $\mathcal{D} \subseteq \mathbb{R}^{m_d}$ are compact sets. We assume f is Lipschitz continuous in s and uniformly bounded. We also assume that the control and disturbance signals $a(\cdot)$ and $d(\cdot)$ are measurable [32]. In most cases, the works we cover either do not have a disturbance variable, or model disturbance as a random sampled value. If there is no disturbance, then the dynamical model is simply $f : \mathcal{S} \times \mathcal{A} \to S$.

Consider a Lipschitz surface function $h : \mathcal{S} \to \mathbb{R}^{\geq 0}$ which is the safety loss function that maps a state to a non-negative real value, which is called the constraint value, or simply cost. Note that $h(s) = 0$ if and only if there is no constraint violation at state s.

The failure set $\mathcal{F}$ is the set of states for which there is an instantaneous constraint

violation. Formally, the failure set is defined as the super-zero level set of h. In particular,

$$s \in \mathcal{F} \iff h(s) > 0. \tag{2.4}$$

On the other hand, a target set is the set of states for which it is desirable to reach, and it can be similarly defined. We will explore target sets in more depth in reach-avoid problems in Section 3.2.

For a deterministic dynamics, it is possible to determine if an initial state will lead to failure despite optimal actions. Then, the value function $V : \mathcal{S} \times \mathbb{R} \to \mathbb{R}$ and associated reachable set $\mathcal{R}(\mathcal{F},t)$ are defined as:

$$V(s,t) := \sup_{d(\cdot)} \inf_{a(\cdot)} \sup_{\tau \in [t,T]} h(s_\tau) \tag{2.5}$$

$$\mathcal{R}(\mathcal{F},t) := \{s \in \mathcal{S} : V(s,t) > 0\} \tag{2.6}$$

In effect, this optimization over the action signal minimizes the maximum possible reachable violation starting from any point in the state space. If the control never enters the failure set when starting from state s, the value function will be zero. Otherwise, the value function will be strictly positive. In the case of a finite horizon in time interval $t \in [0,T]$, dynamic programming can obtain the optimal control and value function. Specifically, this will be the solution to the time-dependent terminal-value Hamilton-Jacobi-Bellman variational inequality (HJBVI) [4]:

$$0 = \max\left\{h(s) - V(s,t), \frac{\partial V}{\partial t} + \min_{a\in\mathcal{A}} \max_{d\in\mathcal{D}} \nabla_s V^\top f(s,a,d)\right\},$$
$$V(s,T) = h(s), \forall s \in \mathcal{S} \tag{2.7}$$

Now as $T \to \infty$, if V converges to a fixed solution then $V(s,t)$ will be independent of t. Thus the time parameter can be dropped to obtain the optimal value function $V(s)$.

2.3 Traditional HJ reachability analysis for learned controls

We first briefly discuss traditional HJ reachability analysis techniques for reinforcement learning-based control. Recent papers propose approaches that evaluate the safety (or probe the safe space) of learning-based control by analytically computing solutions of the dynamics's HJBVI. These methods require having access to or reconstructing the system's model dynamics. With a model, approaches can compute gradients of the dynamics at any given state.

The work of [40] uses model-based HJ reachability analysis in conjunction with Bayesian-inference techniques to create a safety framework that can incorporate an arbitrary learning-based control algorithm. While there are no safety concerns, it permits a learned control policy to optimize for a particular task. Otherwise, it defaults to a safe policy computed via solving the HJBPDE. The safety choice of picking between these two policies is determined via safety analysis refined through Bayesian inferences from online data, particularly using Gaussian processes.

The work of [57] is a model-based approach based on backward reachability. In particular, it iteratively uses backward reachability from the final goal state to construct a set of initial state distributions under some approximate model dynamics. Then, at each iteration, it proposes using model-free methods to acquire a policy to get from an initial state (sampled uniformly from a growing backward reachable set) to the goal.

Another work [3] makes inferences about disturbances to perform reachability analysis. Particularly, the work uses Gaussian processes to construct the disturbance set from previous observations of the dynamics. This is used to solve the HJBPDE and compute an optimally safe control and safety value function. Then, a safe framework can be defined using any safety-aware learned (task-solving) control and this optimally safe control and safety value function. Namely, whenever the value function satisfies some safety threshold, then the safety-aware learned control is deployed. Otherwise, the default optimally safe controller is used.

We will primarily discuss learning-based methods for obtaining the HJ reachability value

function via reinforcement learning. We term this technique as HJ reachability estimation.

2.4 Acknowledgement

Chapter 2 has been submitted for publication of the material in "Hamilton-Jacobi Reachability in Reinforcement Learning: A Survey," M. Ganai; S. Gao; S. Herbert, 2024. The thesis author was the primary investigator and author of this paper.

Chapter 3

Survey of HJ Reachability Estimation Methods

In this chapter, we will survey learning-based methods for Hamilton-Jacobi reachability estimation. We organize the chapter as follows:

- In Section 3.1, we demonstrate how to learn HJ reachability online to acquire reinforcement learning-based control.
- In Section 3.2, we survey various HJ reachability-based/-inspired methods that solve reach-avoid tasks.
- In Section 3.3, we review approaches for model-free safe reinforcement learning in both deterministic and stochastic dynamics scenarios.
- In Section 3.4, we examine HJ reachability estimation-based methods that address robustness and uncertainty issues found in real world environments.

3.1 Learning Reachability in Model-free Settings

Overcoming the computational complexity of traditional HJ reachability analysis methods requires a scalable approach to acquire the HJ reachability value function. The recent literature has proposed a new direction of approximating the HJ reachability value function through learning-based approaches in the face of unknown dynamics. In particular, similar to a reward or

cost critic, an HJ reachability function can be learned in an online, recursive fashion. Within the RL framework, we can construct algorithms that obtain reachable sets via a data-driven, sampling-based manner that is 1) generalizable, since there is no need for direct access to the dynamics, and 2) scalable, in part due to the guaranteed convergence to a unique value function solution with gamma contraction mapping.

3.1.1 Bellman formulation

To learn an estimation of the HJ reachability value function in an online fashion, the value function must be equivalently defined with a backup operator in the form of the recursive Bellman update.

In particular, the works of [4, 42] demonstrate that the discrete approximation of (2.7) with no disturbances is:

$$V(s,t) = \max\left\{h(s), \min_{a\in\mathcal{A}} V(s+f(s,a)\Delta t, t+\Delta t)\right\} \tag{3.1}$$

Furthermore, as $T \to \infty$, if V converges, then V does not change with respect to time, so it satisfies the Bellman equation:

$$V(s) = \max\{h(s), \min_{a\in\mathcal{A}} V(s+f(s,a)\Delta t)\} \tag{3.2}$$

$$= \max\{h(s), \min_{a\in\mathcal{A}} V(s')\} \tag{3.3}$$

where s' is the next state after s in the trajectory. Using this Bellman reformulation, the HJ reachability value function of the optimal control can be learned using the recursive dynamic programming approach known as value iteration. Notice that if this method is used to obtain a value function and optimal policy in a stochastic setting (i.e. the transition function and/or the policy is probabilistic) it would return a value function capturing the expected maximum cost along a trajectory sampled from the policy and transition function. This is not useful or

well-defined for hard constraint tasks since a stochastic policy will likely enter a violation with some non-zero probability when starting from most states.

Nonetheless, it is still possible to use the Bellman recursive formulation for acquiring the HJ reachability value function to learn a meaningful tool for stochastic MDPs and policies using a special cost function [1, 43]. Consider the binary indicator cost function $\mathbb{1}_{h(s)>0}$ which returns 1 if there is a constraint violation at state s, and returns 0 otherwise. In this setting, the optimal control $\pi : \mathcal{S} \times \mathcal{A} \to [0,1]$ is the one that minimizes the likelihood of entering the set of constraint violation states along the trajectory under the stochastic MDP with transition likelihood function P. Formally, in the discrete-time setting, the optimal control and its associated value function $\phi : \mathcal{S} \to [0,1]$, called the reachability estimation function (REF), are defined by [1, 43]:

$$\phi(s) := \inf_{\pi(\cdot|\cdot)} \mathop{\mathbb{E}}_{\xi \sim \pi, P(s)} \sup_{s_t \in \xi} \mathbb{1}_{h(s)>0} \tag{3.4}$$

Although the value function is defined for stochastic dynamics (notice the expectation over the sampled trajectories), [43] exploits the binary nature of the instantaneous cost indicator function to create a Bellman recursive formulation of the REF:

$$\phi(s) = \max \left\{ \mathbb{1}_{h(s)>0}, \min_{\pi(\cdot|s)} \mathop{\mathbb{E}}_{s' \sim \pi, P(s)} \phi(s') \right\} \tag{3.5}$$

When this value function is learned for a particular control it can provide information on the probability that the control at any given state will reach a violation.

3.1.2 Discounted HJ value function for RL

Temporal difference learning is a preeminent class of model-free reinforcement learning algorithms that estimates the value function for a particular control policy. In other words, the value function $V^\pi(s)$ with Bellman operator $\mathcal{B}^\pi$ (i.e. the operator that defines the recursive Bellman formation), should be estimated for a particular control policy π. This can be done by

iteratively updating the value function with the temporal difference rule using trajectory samples collected online. At update k, for learning rate α, the temporal difference rule is [85, 86, 92]:

$$V_{k+1}^{\pi}(s) \leftarrow V_k^{\pi}(s) + \alpha(\mathcal{B}^{\pi}V_k^{\pi}(s) - V_k^{\pi}(s)). \tag{3.6}$$

In order to guarantee convergence to the unique solution of the Bellman equation, the Bellman operator $\mathcal{B}^{\pi}$ must induce a gamma contraction mapping in the space of value functions [36]. In general, time-discounting in the Bellman formulation of the value function enables the reachable set to be estimated as a fixed point in a contraction mapping [4].

To address this, the approach found in [4] proposes a modified discounted optimal control value function. For the defined cost function $h : \mathcal{S} \to \mathbb{R}^{\geq 0}$, the optimal control and value function are defined by:

$$V(s) := \inf_{\pi(\cdot)} \sup_{t \geq 0} h(s_t) e^{-\lambda t} \tag{3.7}$$

for some discount rate $\lambda \in \mathbb{R}^{>0}$.

Similar to the non-discounted Bellman formulation, this value function and its optimal control can be obtained by solving the Hamilton-Jacobi-Bellman variational inequality [4]:

$$0 = \max\left\{h(s) - V(s,t), \min_{a \in \mathcal{A}} \nabla_s V^{\top} f(s,a) - \lambda V(x)\right\} \tag{3.8}$$

This has the discrete-time solution:

$$V(s) = \max\{h(s), \min_{a \in \mathcal{A}} \gamma V(s')\} \tag{3.9}$$

where $\gamma = e^{-\lambda \Delta t}$ is the discount factor. The authors demonstrate the gamma contraction mapping for this discounted Bellman formulation for $\gamma \in (0,1)$, and thereby guarantee that temporal difference learning will converge to the unique value function solution.

The work of [42] proposes a different Bellman formulation for learning an estimation of

the HJ reachability value function:

$$V(s) = (1-\gamma)h(s) + \gamma \max\{h(s), \min_{a \in \mathcal{A}} V(s')\} \tag{3.10}$$

While this is not an exact discrete-time solution of the HJBVI in (3.8), the work of [42] proves this provides a tighter gamma contraction mapping than (3.9), and therefore temporal difference learning can converge to the value function solution faster. Notice that using the cost function as the binary indicator function $\mathbb{1}_{h(s)>0}$ in lieu of $h(s)$ would make (3.9) and (3.10) become identical Bellman formulations.

Using the discounted Bellman formulations, HJ reachability can be incorporated into reinforcement learning problems. In [42], the authors use the HJ reachability value function as the critic and the policy optimization algorithm REINFORCE [95] to solve control problems in environments like the lunar lander and the 18-dimensional jumping half-cheetah.

3.2 Solving Reach-Avoid Problems

Reach-avoid problems form a class of environments in which the goal is to control the agent to reach a target set of states while simultaneously avoiding a failure set of states [9, 12, 41, 69, 73]. We have previously discussed how HJ reachability has been used to solve the avoidance problem. Recent literature has demonstrated how to combine the reach problem and the avoid problem in HJ reachability simultaneously, as well as how to combine HJ reachability with other control theoretic functions to solve the reach-avoid problem in the online setting.

3.2.1 Learning HJ Reach-Avoid Value Function

The work of [41] establishes how to formally define reach-avoid problems. Specifically, the problem seeks to find the optimal control such that given a starting state, the agent can reach the target set of states $\mathcal{T}$ while avoiding the failure set of states $\mathcal{F}$. They define two cost functions

$l : \mathcal{S} \to \mathbb{R}$ and $g : \mathcal{S} \to \mathbb{R}$ such that for any state $s \in S$:

$$\begin{aligned} l(s) \leq 0 &\iff s \in \mathcal{T} \\ g(s) > 0 &\iff s \in \mathcal{F} \end{aligned} \tag{3.11}$$

Then with deterministic MDP, in discrete time, for a finite horizon time T, a payoff function for a deterministic control policy $\pi : \mathcal{S} \to \mathcal{A}$ can be defined as:

$$\mathcal{V}^{\pi}(s,T) = \min_{t \in [0...T]} \max \left\{ l(s_t), \max_{\tau \in [0...t]} g(s_\tau) \right\} \tag{3.12}$$

The outer maximum considers the possibility of ever reaching the target set. The inner maximum ensures that, during the time taken to reach the target set, there are no states in the trajectory that are in the failure set. Thus, for a given time T, if there exists a time t when the agent reaches a state s_t in the target set while avoiding the failure set, then the payoff function will be at most $l(s_t) \leq 0$ and therefore non-positive. However, if the agent always enters the failure set before the target set, then at any time t, there would always exist a time $w \in [0...t]$ such that $g(s_w) > 0$, and therefore the payoff is positive. Step-wise noise disturbance can be considered within the payoff function, and a dynamic programming value iteration approach to obtaining the payoff function for a particular control can be formulated [41].

Consider infinite horizon (i.e. $T \to \infty$). For the sake of simplifying notation, we can define:

$$\mathcal{V}^{\pi}(s) = \lim_{T \to \infty} \mathcal{V}^{\pi}(s,T) \tag{3.13}$$

As shown in a subsequent work [56], the optimal control and its associated value function can then be defined as the one that minimizes the payoff function of (3.12):

$$V(s) = \inf_{\pi(\cdot)} \mathcal{V}^{\pi}(s). \tag{3.14}$$

Observe that the sign of the payoff function can tell us if the control signal starting from state s will satisfy the reach-avoid condition. So, if and only if $V(s) \leq 0$, then there exists a control that can solve the reach avoid problem starting from state s.

Now, just as in the case for model-free learning of the HJ reachability function in Section 3.1.2, it is possible to learn the optimal HJ reach-avoid function. [56] provides a discounted (recall the importance of gamma contraction mapping) reach-avoid Bellman formulation suitable for learning online with temporal difference learning. Specifically,

$$\begin{aligned} V(s) = (1-\gamma)\max\{l(s), g(s)\} \\ + \gamma \max\left\{ \min\left\{ l(s), \min_{a \in \mathcal{A}} V(s') \right\}, g(s) \right\} \end{aligned} \tag{3.15}$$

where s' is the next state produced by the MDP upon taking action a from state s.

With this recursive reformulation of the value function, [56] uses the standard RL algorithm Deep Q-Network (DQN) [75] to obtain the corresponding optimal control policy. They test this algorithm on environments such as an attack-defense game with two Dubins cars, and the Lunar Landing environment.

3.2.2 Combing Reachability with Control Lyapunov for Stabilize-Avoid Problems

Within the class of reach-avoid problems are the stabilize-avoid problems, in which the goal is to find a control that avoids the failure set while stabilizing toward the target set. If the target set consists of equilibrium points, then standard reach-avoid algorithms can be used to solve the stabilize-avoid problems. However, in many cases, the target set may additionally consist of non-equilibrium points. To use the reach-avoid algorithms in the stabilize-avoid problem in this general case, the set of equilibrium points must be extracted from the target set. This extraction is difficult and may even be impossible if such a set does not exist. HJ reachability-inspired approaches can be combined with the control Lyapunov function to solve

Stabilize-Avoid problems.

In the work of [83], the stabilize-avoid problem is formulated as a constraint optimization problem. Particularly, for a deterministic MDP and using the cost functions $l : \mathcal{S} \to \mathbb{R}^{\geq 0}$ and $g : \mathcal{S} \to \mathbb{R}$ with properties of (3.11), the undiscounted value function for policy π is defined along the trajectory as:

$$V^{l,\pi}(s) := \sum_{t=0}^{\infty} l(s_t) \tag{3.16}$$

where $\{s_t\}, t \in \mathbb{Z}^{\geq 0}$ is the trajectory under π starting from state $s = s_0$. Furthermore, the optimal control problem is defined as:

$$\begin{aligned} &\min_{\pi} V^{l,\pi}(s) \\ &\text{s.t. } g(s_t) \leq 0, \forall t \geq 0 \end{aligned} \tag{3.17}$$

Under some assumptions based on bounding the cost function l and its dynamics under control π by some state measure, [83] proves that $V^{l,\pi}$ is a Lyapunov function. They also convert the constraint problem into the epigraph form [14]:

$$\begin{aligned} &\min_{z} z \\ &\text{s.t. } 0 \geq \min_{\pi} \max \left\{ \max_{t \in \mathbb{Z}^{\geq 0}} g(s_t), V^{l,\pi}(s) - z \right\} \end{aligned} \tag{3.18}$$

In effect, z acts as the accumulated l cost budget, and the goal is to minimize the maximum needed cost budget and ensure the agent avoids entering the failure set where $g(s) > 0$. The RHS of the constraint in this epigraph form can be learned as a value function parameterized by both the state and the cost budget. Namely, [83] learns this optimal control value function by applying a recursion similar to (3.15):

$$V(s,z) = \min_{a \in \mathcal{A}} \max\{g(s), V(s', z - l(s))\}. \tag{3.19}$$

The algorithm uses a standard policy gradient approach to learn this value function online, and then in a subsequent stage solves the problem of (3.18) by training via regression a neural network $z(s)$ that minimizes $V(s,z(s))$. This approach has been used to solve various complex stabilize-avoid problems including a 17 dimension F16 fighter jet [51] ground collision avoidance in a low-altitude corridor.

3.3 Model-free Safe RL

Safe reinforcement learning is a setting in which the goal is to maximize some cumulative rewards while constraining the costs (i.e. constraint violations) along a trajectory [15,45,48,103]. In previous sections, the problems were reduced to optimizing a single (potentially composite) value function. However, in safe reinforcement learning, the problem generally requires keeping track of two separate value functions, one for rewards and another for costs, and optimizing a composite expression involving both value functions. The reward value function V_r^π is specifically defined as the discounted cumulative rewards found in Section 2.1. However, the cost value function's definition is determined by the specific optimization framework.

Traditionally, safe reinforcement learning was solved within the constrained Markov decision process (CMDP) framework [7] in which the cost value function was the discounted cumulative costs similar to the reward value function:

$$V_c^\pi(s) := \mathop{\mathbb{E}}_{\xi\sim\pi,P(s)}[\sum_{s_t\in\xi}\gamma^t h(s_t)] \tag{3.20}$$

Then, for some environment-defined positive cost threshold χ, the CMDP-constrained optimization takes the form:

$$\begin{aligned} &\max_\pi \mathop{\mathbb{E}}_{s\sim\Delta_0}[V^\pi(s)] \\ &\text{s.t.} \mathop{\mathbb{E}}_{s\sim\Delta_0}[V_c^\pi(s)] \le \chi \end{aligned} \tag{CMDP}$$

Various approaches have been proposed to solve Safe RL in this framework. Trust-region approaches [2, 97, 98, 102] try to guarantee monotonic improvement in performance while ensuring constraint satisfaction. Primal-dual approaches [37, 67, 77, 88] use Lagrangian relaxation of the constraints to optimize an expression involving the reward and cost value functions. Outside of these two classes exist approaches like constraint-rectified policy optimization (CRPO) [96], which takes a policy gradient update step toward improving V_r^π if constraints are satisfied at a particular iteration, otherwise it takes steps to minimize V_c^π. This approach guarantees convergence to optimum under certain assumptions.

The main drawback of the CMDP framework is its lack of rigorous guarantees of persistent safety. This is because the framework permits some positive amount of constraint violations ($\chi > 0$), and so it cannot be used for state-wise constraint optimization problems. Another issue is that choosing a cost threshold χ for an environment requires tuning and/or prior familiarity with the environment. To address this, recent literature has proposed methods of using the safety guarantees provided by Hamilton-Jacobi reachability to redefine the problem into a constrained optimization within feasible (i.e. constraint-satisfying) states. We explore recent algorithms with frameworks for the deterministic and stochastic dynamics cases.

3.3.1 Deterministic Safe RL

When the MDP is deterministic, the HJ reachability value function can be learned online through the Bellman update from (3.10). Specifically, for a control policy π, define the HJ reachability value function recursively as:

$$V_h^\pi(s) = (1-\gamma)h(s) + \gamma \max\{h(s), V_h^\pi(s')\} \tag{3.21}$$

The reachability value function is used to probe whether a state is within the *feasible* set. This is the set of states starting from which the agent will never enter the failure (i.e. constraint violating) set(s) along its trajectory. Formally, for a particular control π, and its associated

reachability value function V_h^π, the feasible set is defined as:

$$\mathcal{S}_f^\pi := \{s \in \mathcal{S} : V_h^\pi(s) = 0\} \tag{3.22}$$

Some papers refer to this feasible set as the safe set, and is the complement of $\mathcal{R}(\mathcal{F})$ from (2.6). By learning the reward value function V_r^π and reachability value function V_h^π, a recent approach [19] solves safe control tasks by considering the two cases of whether a state is feasible or not and learning a different control for each case. Similar to the CRPO algorithm, during training, if the state is in the feasible set (with some tolerance ε) then an action is taken from the control that optimizes V_r^π and that control is updated. Otherwise if the state is infeasible, then an action is taken from the "safe" control which minimizes the maximum reachable violation, i.e. V_h^π, and this safe control is updated. This technique falls within the broader class of shielding [40], which is discussed in more detail in Section 3.4 This approach is notable for solving a high-dimensional, vision-based autonomous racing environment called Learn-to-Race [52].

However, to fully address the problems of CMDP (lack of safety guarantees stemming from tolerance of some constraint violation), environment-specific cost thresholds/tolerance should be avoided altogether. Instead, the recent literature [43, 99] has moved toward learning optimal (largest) feasible sets. The largest feasible set can be defined as:

$$\mathcal{S}_f := \{s \in \mathcal{S} : \exists \pi, V_h^\pi(s) = 0\} \tag{3.23}$$

In other words, the largest feasible is the set of states for which there exists a control policy that ensures no constraint violations along a trajectory starting from those states. The largest feasible set can also be written as:

$$\mathcal{S}_f = \bigcup_\pi S_f^\pi \tag{3.24}$$

By obtaining or having access to this largest feasible set, the hope is that the algorithms can learn

controls that overcome the conservative behavior seen in other control/energy-based approaches like CBFs [66, 70].

Let the binary function $\mathbb{1}_{s\in\mathcal{S}_f}$ indicate whether a state is in this largest feasible (returning 1) or not (returning 0). Then, the work of [99] proposes a novel optimization framework that considers optimization under two scenarios depending on whether the state is in $\mathcal{S}_f$, assuming one has access to this oracle $\mathbb{1}_{s\in\mathcal{S}_f}$. In particular, if state $s \in \mathcal{S}_f$, the goal would be to optimize for maximum reward value function starting from that state under the constraint that the trajectory continues to persistently remain within the feasible set (and thereby incur no future violations). On the other hand, if the state $s \notin \mathcal{S}_f$, then the goal is to find a control that minimizes the maximum reachable violation starting from that state. Formally, this optimization called Reachability Constrained Reinforcement Learning (RCRL) can be expressed as:

$$\begin{aligned} &\max_{\pi} \mathop{\mathbb{E}}_{s\sim\Delta_0}[V_r^{\pi}(s)\cdot\mathbb{1}_{s\in\mathcal{S}_f} - V_h^{\pi}(s)\cdot\mathbb{1}_{s\notin\mathcal{S}_f}] \\ &\text{s.t. } V_h^{\pi}(s) \leq 0, \forall s \in \mathcal{S}_I \cap \mathcal{S}_f. \end{aligned} \tag{RCRL}$$

The Lagrangian of (RCRL) can be formulated as:

$$\begin{aligned} \mathcal{L}(\pi,\lambda) = \mathop{\mathbb{E}}_{s\sim\Delta_0}[V_r^{\pi}(s)\cdot\mathbb{1}_{s\in\mathcal{S}_f} - V_h^{\pi}(s)\cdot\mathbb{1}_{s\notin\mathcal{S}_f}] \\ + \int_{\mathcal{S}_f\cap\mathcal{S}_I} \lambda(s) V_h^{\pi}(s) ds \end{aligned} \tag{3.25}$$

The main challenge in solving this optimization is being able to acquire the *largest* feasible set. To overcome this, [99] solves their optimization by providing guarantees in stochastic gradient descent optimization of the policies, critics, and Lagrangian multiplier via the stochastic approximation theory framework established in [13, 30], and used in [31].

[99] proposes finding a saddle point of the surrogate Lagrangian optimization of (RCRL) as:

$$\min_{\pi}\max_{\lambda} \mathop{\mathbb{E}}_{s\sim\Delta_0}[-V_r^{\pi}(s) + \lambda(s) V_h^{\pi}(s)] \tag{3.26}$$

The idea behind this formulation is that $\lambda(s)$ will eventually converge to a finite value for feasible states and diverge for infeasible states [67]. Recall that for feasible states s, $V_h^\pi(s) = 0$, so the optimization becomes simply minimizing $-V_r^\pi(s)$ regardless of the magnitude of $\lambda(s)$. However, for infeasible states, $V_h^\pi(s) > 0$, so the optimization minimizes $-V_r^\pi(s) + \lambda V_h^\pi(s)$ for very large λ. Notice, however, that since the Lagrangian multiplier diverges for infeasible states, $-V_r^\pi(s)$ can be ignored. So, the optimization is effectively minimizing $V_h^\pi(s)$.

If $\lambda(s)$ is the Lagrangian multiplier for the optimal control, then solving the surrogate Lagrangian optimization in (3.26) is equivalent to solving the Lagrangian of (3.25). [99] demonstrates this can be achieved primarily by configuring the learning rate schedules of the learned networks. Say, the critics maintain a step size schedule of $\{\zeta_1(k)\}$, the policy maintains a step size schedule of $\{\zeta_2(k)\}$, and the Lagrangian multiplier maintains a step size schedule of $\{\zeta_3 k\}$ for iteration k. Then, based on stochastic approximation theory [13, 30], if:

$$\begin{aligned} \sum_k \zeta_i(k) = \infty \text{ and } \sum_k \zeta_i(k)^2 < \infty, \forall i \in \{1,2,3\} \\ \text{and } \zeta_3(k) = o(\zeta_2(k)), \zeta_2(k) = o(\zeta_1(k)) \end{aligned} \tag{3.27}$$

then it is possible to prove that the updates of the critic, policy, and Lagrangian multiplier will result in convergence of the local optimal policy of RCRL *almost surely* (i.e. with likelihood 1). The reward and cost critic networks have a faster learning rate schedule than the policy networks and therefore converge to the current policy's optimal value functions. The Lagrangian multiplier network has a learning schedule slower than the policy network and therefore can be thought of as capturing the overall trends of feasibility. If during training there was a policy that was able to make a particular state in its feasible set, then $\lambda(s)$ will capture that information. If in the future, the policy no longer makes the state in the feasible set, the Lagrangian multiplier will increase and thereby penalize the policy. Using this approach, [99] is able to solve hard constraint problems in the Safety Gym [77] environment with static hazards and obstacles.

3.3.2 Stochastic Safe RL

Under a stochastic MDP, HJ reachability can still be a useful tool for guaranteeing optimal control with safety guarantees. We present in Section 3.1.1 how recent works define a HJ reachability value function called the Reachability Estimation Function (REF) for a binary cost function $\mathbb{1}_{h(s)>0}$ under stochastic dynamics. The optimal REF captures the minimum likelihood of entering the set of constraint violation states. In effect, the REF is the likelihood that a state is *infeasible* – we will therefore use the phrase *likelihood of feasibility* to mean $1-\phi(s)$ and *the likelihood of infeasibility* to mean $\phi(s)$.

The work of [43] proposes to use the REF function in defining the optimization formulation. In particular, in place of the deterministic feasibility indicator $\mathbb{1}_{s\in\mathcal{S}_f}$ they use the likelihood of feasibility $1-\phi(s)$, and instead of the deterministic infeasibility indicator $\mathbb{1}_{s\notin\mathcal{S}_f}$ they use the likelihood of infeasibility $\phi(s)$. Note these feasibility sets are the largest/optimal.

However, simply replacing the indicator function with $\phi(s)$ in the optimization of (RCRL) will not be a valid construction for the stochastic case since V_h^π is not well defined for stochastic dynamics. [43] addresses this by using the cumulative cost function V_c^π as defined in the CMDP framework in (3.20). In particular, they replace V_h^π with V_c^π in (RCRL).

In the constraint, $V_c^\pi(s)\leq 0$ is satisfied if and only if persistent safety (i.e. no constraint violations along the trajectory) is guaranteed for that state under control policy π. Therefore, $V_c^\pi(s)\leq 0$ can be used as a valid measure for constraining the agent to remain within the feasible set.

Furthermore, V_c^π provides important safety guarantees when the agent is in the infeasible set. Specifically, [43] proves that an optimal control minimizing V_c^π can verifiably *enter* the feasible set when starting in the infeasible set if there exists a control given sufficient time. Intuitively, consider that $V_c^\pi(s)$ is the (average) cumulative cost of a trajectory starting at s (ignore the discount factor by making say $\gamma=1$). If the control enters the feasible set, $V_c^\pi(s)$ is finite since there will be a point after will no more costs are accumulated. Otherwise if the control

remains in the infeasible set, then $V_c^\pi(s)$ is infinite since there will always be costs accumulated at some points in the trajectory. Thus, if there exists a control that enters the feasible set at state s, then the minimum cumulative cost for a policy starting from state s is finite, and thus the optimal control minimizing $V_c^\pi(s)$ will enter the feasible set. [43] provides a proof along these lines with consideration to the discount factor $\gamma \in [0,1)$.

Using the REF and the cumulative cost value function, [43] proposes an optimization formulation for safety constraint reinforcement learning that works for both stochastic and deterministic environments. Formally, their optimization called Reachability Estimation for Safe Policy Optimization (RESPO) can be expressed as:

$$\begin{aligned} \max_{\pi} \; & \mathop{\mathbb{E}}_{s\sim\Delta_0}[V_r^\pi(s)\cdot(1-\phi(s)) - V_c^\pi(s)\cdot\phi(s)] \\ \text{s.t. } & V_c^\pi(s) \le 0, \text{ w.p. } 1-\phi(s), \forall s \in S_I. \end{aligned} \tag{RESPO}$$

To learn the value function online, they create a discounted Bellman formulation to ensure gamma contraction mapping to demonstrate convergence to the solution (Section 3.1.2). Thus, they define a discounted Bellman formulation of the REF as:

$$\phi(s) = \max\{\mathbb{1}_{h(s)>0}, \gamma \min_{a\in\mathcal{A}} \mathop{\mathbb{E}}_{s'\sim P(s,a)} \phi(s')\} \tag{3.28}$$

The Lagrangian of (RESPO) is formulated as:

$$\mathop{\mathbb{E}}_{s\sim\Delta_0}\Big[[-V_r^\pi(s) + \lambda\cdot V_c^\pi(s)]\cdot(1-\phi(s)) + V_c^\pi(s)\cdot\phi(s)\Big] \tag{3.29}$$

Similar to (RCRL), the main challenge in solving RESPO is obtaining the optimal REF. [43] proposes solving this problem via the stochastic approximation theory framework [13, 30]. Similar to (3.27), say the learning rates of the critic value functions, the policy, REF, and

lagrangian multiplier are $\{\zeta_1(k)\}$, $\{\zeta_2(k)\}$, $\{\zeta_3(k)\}$, and $\{\zeta_4(k)\}$ respectively. Then if:

$$\begin{aligned} \sum_k \zeta_i(k) = \infty \text{ and } \sum_k \zeta_i(k)^2 < \infty, \forall i \in \{1,2,3,4\} \\ \text{and } \zeta_i(k) = o(\zeta_{i-1}(k)), \forall i \in \{2,3,4\} \end{aligned} \tag{3.30}$$

then [43] guarantees that the updates of the various learnable parameters will result in the policy network converging to the local optimal policy of RESPO *almost surely*. The reasoning is mostly similar to that of RCRL [99] except for the stochastic nature of the dynamics and ϕ. In particular, since the learning rate schedule for the REF ϕ is slower than that of the policy, [43] guarantees that ϕ will be the REF of the most optimal policy to the extent that the lagrangian multiplier λ allows (since λ is technically finite). RESPO learns stochastic policies that solve safety constrained problems in the Safe PyBullet framework [47], MuJoCo [91], and Safety Gym [77] in which there are various moving/movable obstacles in addition to stationary regions. Furthermore, [43] demonstrates how RESPO can incorporate and prioritize multiple hard and soft constraints to solve a multi-drone tunnel navigation environment. More details on this approach is explained in Chapter 4.

3.4 Robustness and real-world settings

While most of the applications of Hamilton-Jacobi Reachability we discussed so far solve problems in simulation, there has also been a line of work on learning verifiably safe controls in real-world settings. The main challenge in real-world settings is the presence of nondeterministic disturbances at each step. Take for instance quadrupedal robot control: the optimal control problem can be formulated as getting to region B in the fastest way possible, but other factors to consider include the presence of some unknown amount of wind or uncertain terrain.

The recent literature solves this primary by constructing a safety filter [53] criterion $\Delta : \mathcal{S} \times \Pi \times \mathcal{Q} \to \{0,1\}$ dependent on the state $s \in \mathcal{S}$, the task solving (i.e. performance optimizing) control $\pi^t \in \Pi$, and backup optimally safe q-value function $Q^u \in \mathcal{Q}$. They can then

define a composite policy π^{sh} that uses the safety filter criterion Δ to decide whether to use the task-solving control π^t or the backup optimally safe policy π^u corresponding to Q^u. This approach of using the backup safe policy to override the tasking-solving policy is known as the least restrictive control law or shielding in [5, 40] and also examined in [25, 64].

Hamilton-Jacobi reachability estimation methods have been used in constructing the safety filter criterion and/or the backup optimally safe policy. For instance, based on the work of [40], it is possible to construct the optimally safe q-value function in a Bellman formulation similar to that in (3.8):

$$Q^u(s,a) = (1-\gamma)h(s) + \gamma \max\left\{h(s), \min_{a' \in \mathcal{A}} Q^u(s',a')\right\} \tag{3.31}$$

and define the safety filter criterion with an indicator function as:

$$\Delta(s, \pi^t, Q^u) := \mathbb{1}\left\{Q^u(s, \pi^t(s)) \leq \varepsilon\right\} \tag{3.32}$$

for some threshold ε. Then the composite policy can be formally constructed as:

$$\pi^{sh}(s) = \begin{cases} \pi^t(s), & \Delta(s, \pi^t, Q^u) = 1 \\ \pi^u(s), & \text{otherwise} \end{cases} \tag{3.33}$$

3.4.1 Fully Learning-based control for Real-World Deployment

Using this framework, it is possible to acquire policies that are (almost) ready to be deployed in real-world scenarios. One difficulty in deploying these algorithms is that learned control often struggles to generalize in new, unseen environments in the real world. To address this distributional shift between the simulation-based training data and the real-world testing data, the work of [55] proposes a technique based on encouraging the generalization capabilities of the learned policies. They develop a 3-tiered approach: learning control policies in Simulation,

fine-tuning in a Lab, and then transferring the policies into the Real World. When training in Simulation, they use the HJ reachability-based shielding approach trained on RGB image vision-based observations. They augment this with a learning framework that optimizes for the diversity of robot learning behavior following the works of [38, 78]. The goal behavior in the simulation phase is to be able to reach the specified target through various paths. This can be done by conditioning the policy by some random latent variable representing a learned "skill" (i.e. taking a specific path to the target). By learning various ways (skills) to solve the problem, they can encourage the generalization capabilities of the learned control.

Subsequently, during the fine-tuning phase in the Lab environment, they can learn a prior distribution from which to sample the latent variables so as to find the best "skills," which were already learned in the simulation phase, needed to solve in some new lab environments. [55] proposes doing this by leveraging the PAC-Bayes Control framework [39, 68, 93] to certify the generalization of the corresponding posterior distribution. Overall, this approach was tested on hardware experiments with the quadrupedal robot in real world indoor spaces.

3.4.2 Learning-based Control Shielded with Forward Reachability in Real-world Deployment

While learning-based control has the benefit of being scalable, the learned policy may not be accurate for all points in the state space and in general lacks intrinsic guarantees of safety. The work of [54] addresses this problem by combining HJ reachability estimation and traditional HJ reachability analysis. While they use a shielding framework similar to [40, 55], they learn a backup optimally safe controller that is disturbance aware and then define a new composite policy that includes the task solving policy π^t, the safe controller π^u, and an additional safe control policy based-on locally computing the forward reachability set.

To obtain the disturbance-aware backup controller, recent work considers the problem of obtaining a safe control policy that is resilient to the worst-case disturbance at each step. Specifically, while learning a control π^u to solve the problem, [54] proposes simultaneously

treating the disturbance as an antagonist controlled with policy π^d. Then, in the typical game theoretic, adversarial fashion, the goal is to find a saddle point between both π^u and π^d. Formally, the optimal controls and associated value function can be defined with the Bellman formulation:

$$V(s) = (1-\gamma)h(s) + \gamma \min_{\pi^u} \max_{\pi^d} \mathop{\mathbb{E}}_{u,d} \max\left\{h(s), V(s')\right\} \tag{3.34}$$

The optimal control policies for this formulation are learned via the off-policy reinforcement learning algorithm Soft Actor-Critic algorithm [49].

While these learned controls cannot provide intrinsic safety guarantees, [54] constructs a composite policy that guarantees safety for H horizon steps. In particular, they linearize dynamics of the nominal local trajectory starting from state s obtained from the learned control. Then at some point s' along the trajectory, they use a linear quadratic regulator approach to obtain a locally linear tracking policy $K(s'-s)$ for H time into the future. Subsequently, they can define a safety criterion $\Delta : \mathcal{S} \times \Pi \times \mathbb{Z}^{\geq 0}$. $\Delta(s, \pi^t, H) = 1$ if after applying one step of the task policy π^t, tracking policy K can maintain safety under any disturbance for time horizon H – this is verified via forward HJ reachability analysis. Else $\Delta(s, \pi^t, H) = 0$. So, for a given state s_t and future time step $\tau \in \{0...H\}$ along the nominal trajectory starting from s_t, the composite policy can be defined as:

$$\pi^{sh}(s_{t+\tau}) = \begin{cases} \pi^t(s_t), & \Delta(s_{t+\tau}, \pi^t, H) = 1 \\ K(s_{t+\tau} - s_t), & \Delta(s_{t+\tau}, \pi^t, H) = 0 \wedge \tau \in \{1...H\} \\ \pi^u(s_t), & \text{otherwise} \end{cases} \tag{3.35}$$

Using this policy, [54] tests on a small robot car with uncertain dynamics.

3.5 Acknowledgement

Chapter 3 has been submitted for publication of the material in “Hamilton-Jacobi Reachability in Reinforcement Learning: A Survey,” M. Ganai; S. Gao; S. Herbert, 2024. The thesis author was the primary investigator and author of this paper.

Chapter 4

Iterative Reachability Estimation for Safe Reinforcement Learning

4.1 Introduction

Ensuring safety is important for the practical deployment of reinforcement learning (RL). Various challenges must be addressed, such as handling stochasticity in the environments, providing rigorous guarantees of persistent state-wise safety satisfaction, and avoiding overly conservative behaviors that sacrifice performance. We propose a new framework, Reachability Estimation for Safe Policy Optimization (RESPO), for safety-constrained RL in general stochastic settings. In the feasible set where there exist violation-free policies, we optimize for rewards while maintaining persistent safety. Outside this feasible set, our optimization produces the safest behavior by guaranteeing entrance into the feasible set whenever possible with the least cumulative discounted violations. We introduce a class of algorithms using our novel reachability estimation function to optimize in our proposed framework and in similar frameworks such as those concurrently handling multiple hard and soft constraints. We theoretically establish that our algorithms almost surely converge to locally optimal policies of our safe optimization framework. We evaluate the proposed methods on a diverse suite of safe RL environments from Safety Gym, PyBullet, and MuJoCo, and show the benefits in improving both reward performance and safety compared with state-of-the-art baselines.

4.2 Stochastic Hamilton-Jacobi Reachability for Reinforcement Learning

Classic HJ reachability considers finding the largest feasible set for deterministic environments. In this section, we apply a similar definition in [1, 84] and define the stochastic reachability problem.

4.2.1 Persistent Safety and HJ Reachability for Stochastic Systems

The instantaneous safety can be characterized by the safe set $\mathcal{S}_s$, which is the zero level set of the safety loss function $h : \mathcal{S} \mapsto \mathbb{R}_0^+$. The unsafe (i.e. violation) set $\mathcal{S}_v$ is the complement of the safe set.

Definition 1. Safe set and unsafe set: $\mathcal{S}_s := \{s \in \mathcal{S} : h(s) = 0\}, \mathcal{S}_v := \{s \in \mathcal{S} : h(s) > 0\}$.

We will write $\mathbb{1}_{s \in \mathcal{S}_v}$ as the *instantaneous violation indicator function*, which is 1 if the current state is in the violation set and 0 otherwise. Note that the safety loss function h is different from the instantaneous violation indicator function since h captures the magnitude of the violation at the state.

It is insufficient to only consider instantaneous safety. When the environment and policy are both deterministic, we easily have a unique trajectory for starting from each state (i.e. the future state is uniquely determined) under Lipschitz environment dynamics. In classic HJ reachability literature [8], for a deterministic MDP's transition model P_d and deterministic policy π_d, the set of states that guarantees persistent safety is captured by the zero sub-level set of the following value function:

Definition 2. Reachability value function $V_h^\pi : \mathcal{S} \mapsto \mathbb{R}_0^+$ is: $V_h^\pi(s) := \max_{s_t \in \tau \sim \pi_d, P_d(s)} h(s_t)$.

However, when there's a stochastic environment with transition model $P(\cdot|s,a)$ and policy $\pi(\cdot|s)$, the future states are not uniquely determined. This means for a given initial state and policy, there may exist many possible trajectories starting from this state. In this case, instead of

defining a binary function that only indicates the existence of constraint violations, we define the reachability estimation function (REF), which captures the probability of constraint violation:

Definition 3. The reachability estimation function (REF) $\phi^{\pi} : \mathcal{S} \mapsto [0,1]$ is defined as:

$$\phi^{\pi}(s) := \mathop{\mathbb{E}}_{\tau \sim \pi, P(s)} \max_{s_t \in \tau} \mathbb{1}_{(s_t | s_0 = s, \pi) \in \mathcal{S}_v}.$$

In a specific trajectory τ, the value $\max_{s_t \in \tau} \mathbb{1}_{(s_t | s_0 = s, \pi) \in \mathcal{S}_v}$ will be 1 if there exist constraint violations and 0 if there exists no violation, which is binary. Taking expectation over this binary value for all the trajectories, we get the desired probability. We define optimal REF based on an optimally safe policy $\pi^* = \arg\min_{\pi} V_c^{\pi}(s)$ (note that this policy may not be unique).

Definition 4. The optimal reachability estimation function $\phi^* : \mathcal{S} \mapsto [0,1]$ is: $\phi^*(s) := \phi^{\pi^*}(s)$.

Interestingly, we can utilize the fact the instantaneous violation indicator function produces binary values to learn the REF function in a bellman recursive form. The following will be used later:

Theorem 1. The REF can be reduced to the following recursive Bellman formulation:

$$\phi^{\pi}(s) = \max\{ \mathbb{1}_{s \in \mathcal{S}_v}, \mathop{\mathbb{E}}_{s' \sim \pi, P(s)} \phi^{\pi}(s') \},$$

where $s' \sim \pi, P(s)$ is a sample of the immediate successive state (i.e., $s' \sim P(\cdot|s, a \sim \pi(\cdot|s))$) and the expectation is taken over all possible successive states.

Proof.

$$
\begin{aligned}
\phi^{\pi}(s) &:= \mathop{\mathbb{E}}_{\tau\sim\pi,P(s)} \max_{s_t\in\tau} \mathbb{1}_{s_t^{\pi}\in S_v} \\
&= \mathop{\mathbb{E}}_{\tau\sim\pi,P(s)} \max\{\mathbb{1}_{s\in S_v}, \max_{s_t\in\tau\setminus\{s\}} \mathbb{1}_{s_t^{\pi}\in S_v}\} \\
&= \max\{\mathbb{1}_{s\in S_v}, \mathop{\mathbb{E}}_{\tau\sim\pi,P(s)} \max_{s_t\in\tau\setminus\{s\}} \mathbb{1}_{s_t^{\pi}\in S_v}\} \\
&= \max\{\mathbb{1}_{s\in S_v}, \mathop{\mathbb{E}}_{s'\sim\pi,P(s)} \mathop{\mathbb{E}}_{\tau'\sim\pi,P(s')} \max_{s_t\in\tau'} \mathbb{1}_{s_t^{\pi}\in S_v}\} \\
&= \max\{\mathbb{1}_{s\in S_v}, \mathop{\mathbb{E}}_{s'\sim\pi,P(s)} \phi^{\pi}(s')\}
\end{aligned}
$$

Note that we use the notation $\tau\sim\pi,P(s)$ to indicate a trajectory sampled from the MDP with transition probability P under policy π starting from state s, and use the notation $s'\sim\pi,P(s)$ to indicate the next immediate state from the MDP with transition probability P under policy π starting from state s. The third line holds because the indicator function is either 0 or 1, so if it's 1 then $\phi^{\pi}(s) = \mathbb{E}_{\tau\sim\pi,P(s)}\, 1 = 1$ else $\phi^{\pi}(s) = \mathbb{E}_{\tau\sim\pi,P(s)} \max_{s_t\in\tau\setminus\{s\}} \mathbb{1}_{s_t^{\pi}\in S_v}$.

□

Definition 5. The feasible set of a policy π based on $\phi^{\pi}(s)$ is defined as: $\mathcal{S}_f^{\pi} := \{s\in\mathcal{S} : \phi^{\pi}(s) = 0\}$.

Note, the feasible set for a specific policy is the set of states starting *from* which no violation is reached, and the safe set is the set of states *at* which there is no violation. We will use the phrase likelihood of being feasible to mean the likelihood of not reaching a violation, i.e. $1-\phi^{\pi}(s)$.

4.2.2 Comparison with RCRL

The RCRL approach [99] uses reachability to optimize and maintain persistent safety in the feasible set. Note, in below formulation, $\mathcal{S}_f$ is the optimal feasible set, i.e. that of a policy

$\arg\min_\pi V_h^\pi(s)$. The RCRL formulation is:

$$\max_\pi \mathop{\mathbb{E}}_{s\sim d_0}[V^\pi(s)\cdot \mathbb{1}_{s\in\mathcal{S}_f} - V_h^\pi(s)\cdot \mathbb{1}_{s\notin\mathcal{S}_f}], \text{ subject to } V_h^\pi(s)\leq 0, \forall s\in\mathcal{S}_I\cap\mathcal{S}_f. \qquad \text{(RCRL)}$$

The equation RCRL considers two different optimizations. When in the optimal feasible set, the optimization produces a persistently safe policy maximizing rewards. When outside this set, the optimization produces a control minimizing the maximum future violation, i.e. $\arg\min_\pi V_h^\pi(s)$. *However, this does not ensure (re)entrance into the feasible set even if such a control exists.*

RCRL performs constraint optimization on V_h^π with a neural network (NN) lagrange multiplier with state input [67]. When learning to optimize a Lagrangian dual function, the NN lagrange multiplier should converge to small values for states in the optimal feasible set and converge to large values for other states. Nonetheless, learning V_h provides a weak signal during training: if there is an improvement in safety along the trajectory not affecting the maximum violation, V_h^π remains the same for all states before the maximum violation in the trajectory. These improvements in costs can be crucial in guiding the optimization toward a safer policy. And optimizing with $V_h(s)$ can result in accumulating an unlimited number of violations smaller than the maximum violation. Also, a major issue with this approach is that *it's limited to deterministic MDPs and policies* because its reachability value function in the Bellman formulation does not directly apply to the stochastic setting. However, in general *stochastic* settings, estimating feasibility cannot be binary since for a large portion of the state space, even under the optimal policy, the agent may enter the unsafe set with a non-zero probability, rendering such definition too conservative and impractical.

4.3 Iterative Reachability Estimation for Safe Reinforcement Learning

In this paper, we formulate a general optimization framework for safety-constrained RL and propose a new algorithm to solve our constraint optimization by using our novel reachability estimation function. We present the deterministic case in Section 4.3.1 and build our way to the stochastic case in Section 4.3.2. We present our novel algorithm to solve these optimizations, involving our new reachability estimation function, in Section 4.3.3. We introduce convergence analysis in Section 4.3.4.

4.3.1 Iterative Reachability Estimation for Deterministic Settings

All state transitions and policies happen with likelihood 0 or 1 for the deterministic environment. Therefore, the probability of constraint violation for policy π from state s, i.e., $\phi^{\pi}(s)$, is in the set $\{0,1\}$. According to Definition 4, if there exists some policy π such that $\phi^{\pi}(s) = 0$, we have $\phi^{*}(s) = 0$. Otherwise, $\phi^{*}(s) = 1$. Notice that this captures definitive membership in the optimal feasible set $\phi^{*}(s) = \mathbb{1}_{s \in S_f^{\pi_s}}$, which is the feasible set of some safest policy $\pi_s = \arg\min_{\pi} V_c^{\pi}(s)$. Now, we divide our optimization in two parts: the infeasible part and the feasible part.

For the infeasible part, we want the agent to incur the least cumulative damage (discounted sum of costs) and, if possible, (re)enter the feasible set. Different from previous Reachability-based RL optimizations, by using the discounted sum of costs $V_c^{\pi}(s)$ we consider both magnitude and frequency of violations, thereby improving learning signal. The infeasible portion takes the form:

$$\max_{\pi} \mathop{\mathbb{E}}_{s \sim d_0} [-V_c^{\pi}(s)]. \tag{4.1}$$

For the feasible part, we want the policy to ensure the agent stays in the feasible set and maximize reward returns. This produces a constraint optimization where the cost value function

is constrained:

$$\max_{\pi} \mathop{\mathbb{E}}_{s\sim d_0}\left[V^{\pi}(s)\right], \text{ subject to } V_c^{\pi}(s) = 0, \forall s \in \mathcal{S}_I. \tag{4.2}$$

The following propositions justify using V_c^{π} as the constraint.

Proposition 1. The cost value function $V_c^{\pi}(s)$ is zero for state s if and only if the persistent safety is guaranteed for that state under the policy π.

Proof. (IF) Assume for a given policy π, the persistent safety is guaranteed, i.e. $h(s_t|s_0 = 0, \pi) = 0$ holds for all $s_t \in \tau$ for all possible trajectories τ sampled from the environment with control policy π. We then have:

$$V_c^{\pi}(s) := \mathop{\mathbb{E}}_{\tau\sim\pi, P(s)}[\sum_{s_t\in\tau} \gamma^t h(s_t)] = 0.$$

(ONLY IF) Assume for a given policy π, $V_c^{\pi}(s) = 0$. Since the image of the safety loss function $h(s)$ is non-negative real, and $V_c^{\pi}(s)$ is the expectation of the sum of non-negative real values, the only way $V_c^{\pi}(s) = 0$ is if $h(s_t|s_0 = 0, \pi) = 0$, $\forall s_t \in \tau$ for all possible trajectories τ sampled from the environment with control policy π. □

We define here $\mathcal{S}_f := \mathcal{S}_f^{\pi_s}$, the feasibilty set of some safest policy. Now, the above two optimizations can be unified with the use of the feasibility function $\phi^*(s)$:

$$\max_{\pi} \mathop{\mathbb{E}}_{s\sim d_0}\left[V^{\pi}(s)\cdot(1-\phi^*(s)) - V_c^{\pi}(s)\cdot\phi^*(s)\right], \text{ subject to } V_c^{\pi}(s) = 0, \forall s \in \mathcal{S}_I \cap \mathcal{S}_f. \tag{4.3}$$

Unlike other reachability based optimizations like RCRL, one particular advantage in Equation 4.3 is, with some assumptions, the guaranteed entrance back into feasible set with minimum cumulative discounted violations whenever a possible control exists. More formally, assuming infinite horizon:

Proposition 2. If $\exists\pi$ that produces trajectory $\tau = \{(s_i), i \in \mathbb{N}, s_1 = s\}$ in deterministic MDP $\mathcal{M}$ starting from state s, and $\exists m \in \mathbb{N}, m < \infty$ such that $s_m \in S_f^\pi$, then $\exists \varepsilon > 0$ where if discount factor $\gamma \in (1-\varepsilon, 1)$, then the optimal policy π^* of Equation 4.3 will produce a trajectory $\tau' = \{(s'_j), j \in \mathbb{N}, s'_1 = s\}$, such that $\exists n \in \mathbb{N}, n < \infty$, $s'_n \in S_f^{\pi^*}$ and $V_c^{\pi^*}(s) = \min_{\pi'} V_c^{\pi'}(s)$.

In other words the proposition is stating for some state s, if there is a policy that enters its feasible set in a finite number $(m-1)$ of steps, then by ensuring discount factor γ is close to 1 we can guarantee that the optimal policy π^* of Main paper Equation 4.3 will also enter the feasible set in a finite number of steps with the minimum cumulative discounted sum of the costs. Note that π^* will always produce trajectories with the minimum discounted sum of costs whether the state is in the feasible or infeasible set of the policy by virtue of its optimization which constrains V_c^π.

Proof. We consider two cases: (Case 1) $m = 1$ and (Case 2) $m > 1$.

Case 1 $m = 1$**:** In this case, there exists a policy π in which the the current state s is in the feasible set of that policy. By definition, that means that in a trajectory τ sampled in the MDP using that policy, starting from state s, there are no future violations incurred in τ. Thus $V_c^\pi(s) = 0$. Since π^* incurs the minimum cumulative violation, $V_c^{\pi^*}(s) = 0$ trivially. Therefore, s, the first state of the trajectory, is in the feasible set of π^*.

Case 2 $m > 1$**:** Since policy π^* produces the minimum cumulative discounted cost for a given state s, the core of this proof will be demonstrating that the minimum cumulative discounted cost of *entering* the feasible set (call this value H_E) is less than the minimum cumulative discounted cost of *not entering* the feasible set (call this value H_N), and therefore π^* will choose the route of entering the feasible set.

The proof will proceed by deriving a sufficient condition for $H_E < H_N$ by establishing bounds on them.

We place an upper bound on the minimum cumulative discounted cost of entering the feasible set H_E. Since $\exists\pi$ that enters the feasible set in $m-1$ steps, entering the feasible set can

be at most the highest possible cost that π incurs. Since the maximum cost at any state is $H_{\max}$, the upper bound is the discounted sum of $m-1$ steps of violations $H_{\max}$, or

$$H_E < \frac{H_{\max}(1-\gamma^{m-1})}{(1-\gamma)}$$

We place a lower bound on the minimum cumulative discounted cost of not entering the feasible set H_N. In this case, say in the sampled trajectory, the maximum gap between any two non-zero violations is w. By definition, the trajectory cannot have an infinite sequence of violation-free states since the trajectory never enters the feasible set. Therefore w is finite. Now recall $H_{\min}$ is the lower bound on the non-zero values of h. So the minimum cumulative discounted cost of not entering the feasible set must be at least the cost of the trajectory with a violation of $H_{\min}$ at intervals of w steps. That is:

$$\frac{H_{\min}(\gamma^w)}{(1-\gamma^w)} < H_N$$

Now $H_E < H_N$ will be true if the upper bound of H_E is less than the lower bound of H_N. In other words $H_E < H_N$ is true if:

$$\frac{H_{\max}(1-\gamma^{m-1})}{(1-\gamma)} < \frac{H_{\min}(\gamma^w)}{(1-\gamma^w)} \tag{4.4}$$

Rearranging, we get:

$$\frac{H_{\max}}{H_{\min}} < \frac{(1-\gamma)\cdot(\gamma^w)}{(1-\gamma^{m-1})\cdot(1-\gamma^w)} \tag{4.5}$$

Let's define the RHS of the Inequality 4.5 as the function $\upsilon(\gamma)$. Consider $\gamma \in (0,1)$. It is not difficult to demonstrate that $\upsilon(\gamma)$ in this domain range is a continuous function and that left directional limit $\lim_{\gamma\to 1^-} \upsilon(\gamma) = \infty$. This suggests that there is an open interval of values for γ (whose supremum is 1) for which $H_{\max}/H_{\min} < \upsilon(\gamma)$ and so $H_E < H_N$. So we establish that $\exists \varepsilon > 0$ such that for $\gamma \in (1-\varepsilon, 1)$, we satisfy the sufficient condition $H_E < H_N$ so that the

optimal policy will enter its feasible set.

Thus, we prove that if there is a policy entering its feasible set from state s, then there is a range of values for γ that are close enough to 1 ensuring that the optimal policy of Main paper Equation 4.3 will enter its feasible set in a finite number of steps with minimum discounted sum of costs.

□

4.3.2 Iterative Reachability Estimation for Stochastic Settings

In stochastic environments, for each state, there is some likelihood of entering into the unsafe states under any policy. Thus, we adopt the probabilistic reachability Definitions 3 and 4. Rather than using the binary indicator in the optimal feasible set to demarcate the feasibility and infeasibility optimization scenarios, we use the likelihood of infeasibility of the safest policy. In particular, for any state s, the optimal likelihood that the policy will enter the infeasible set is $\phi^*(s)$ from Definition 4.

We again divide the full optimization problem in stochastic settings into infeasible and feasible ones similar to Equations 4.1 and 4.2. However, we consider the infeasible formulation with likelihood the current state is in a safest policy's infeasible state, or $\phi^*(s)$. Similarly, we account for the feasible optimization formulation with likelihood the current state is in a safest policy's feasible set, $1-\phi^*(s)$. The complete Reachability Estimation for Safe Policy Optimization (RESPO) can be rewritten as:

$$\max_{\pi} \mathop{\mathbb{E}}_{s\sim d_0}[V^{\pi}(s)\cdot(1-\phi^*(s)) - V_c^{\pi}(s)\cdot\phi^*(s)], \text{ s.t., } V_c^{\pi}(s)=0, \text{ w.p. } 1-\phi^*(s), \forall s\in S_I. \tag{RESPO}$$

In sum, the RESPO framework provides several benefits when compared with other constrained Reinforcement Learning and reachability-based approaches. Notably, 1) it maintains persistent safety when in the feasible set unlike CMDP-based approaches, 2) compared with

other reachability-based approaches, RESPO considers performance optimization in addition to maintaining safety, 3) it maintains the behavior of a safest policy in the infeasible set and even reenters the feasible set when possible, 4) RESPO employs rigorously defined reachability definitions even in stochastic settings.

4.3.3 Overall Algorithm

We describe our algorithms by breaking down the novel components. Our algorithm predicts reachability membership to guide the training toward optimizing the right portion of the optimization equation (i.e., feasibility case or infeasibility case). Furthermore, it exclusively uses the discounted sum of costs as the safety value function – we can avoid having to learn the reachability value function while having the benefit of exploiting the improved signal in the cost value function.

Optimization in infeasible set versus feasible set.

If the agent is in the infeasible set, this is the simplest case. We want to find the optimal policy that maximizes $-V_c^{\pi}(s)$. This would be the only term that needs to be considered in optimization.

On the other hand, if the agent is in the feasible set, we must solve the constraint optimization $\max_{\pi} V^{\pi}(s)$ subject to $V_c^{\pi}(s) = 0$. This could be solved via a Lagrangian-based method:

$$\min_{\pi}\max_{\lambda} L(\pi,\lambda) = \min_{\pi}\max_{\lambda}\left(\mathop{\mathbb{E}}_{s\sim d_0}[-V^{\pi}(s) + \lambda V_c^{\pi}(s)]\right).$$

Now what remains is obtaining the reachability estimation function ϕ^*. First, we address the problem of acquiring optimal likelihood of being feasible. It is nearly impossible to accurately know before training if a state is in a safest policy's infeasible set. We propose learning a function guaranteed to converge to this REF (with some discount factor for γ-contraction mapping) by using the recursive Bellman formulation proved in Theorem 1.

We learn a function $p(s)$ to capture the probability $\phi^*(s)$. It is trained like a reachability function:

$$p(s) = \max\{\mathbb{1}_{s\in S_v}, \gamma \cdot p(s')\},$$

where S_v is the violation set, s' is the next sampled state, and γ is a discount parameter $0 \ll \gamma < 1$ to ensure convergence of $p(s)$. Furthermore, and crucially, we ensure the learning rate of this REF is on a slower time scale than the policy and its critics but faster than the lagrange multiplier.

Bringing the concepts covered above, we present our full optimization equation:

$$\min_{\pi}\max_{\lambda} L(\pi,\lambda) = \min_{\pi}\max_{\lambda}\left(\mathop{\mathbb{E}}_{s\sim d_0}\left[[-V^{\pi}(s) + \lambda \cdot V_c^{\pi}(s)]\cdot(1-p(s)) + V_c^{\pi}(s)\cdot p(s)\right]\right). \quad (4.6)$$

We show the design of our algorithm **RESPO** in an actor-critic framework in Algorithm 1. Note that the V and V_c have corresponding Q functions: $V^{\pi}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)} Q(s,a)$ and $V_c^{\pi}(s) = \mathbb{E}_{a\sim\pi(\cdot|s)} Q_c(s,a)$. We use operator Γ_{Θ} to indicate the projection of vector $\theta \in \mathbb{R}^n$ to the closest point in compact and convex set $\Theta \subseteq \mathbb{R}^n$. Specifically, $\Gamma_{\Theta} = \arg\min_{\hat{\theta}\in\Theta} ||\hat{\theta} - \theta||^2$. Γ_{Ω} is similarly defined.

Algorithm 1. RESPO Actor Critic

Require: Randomly initialized policy π_θ's parameters θ_0, reward critic Q's parameters η_0, cost critic Q_c's parameters κ_0, REF p's parameters ξ_0, Lagrange multiplier λ's parameters ω_0, horizon T

Require: Convex projection operators Γ_Θ and Γ_Ω, and reward and cost critic learning rate $\zeta_1(k)$, policy learning rate $\zeta_2(k)$, REF learning rate $\zeta_3(k)$, lagrange multiplier learning rate $\zeta_4(k)$

for $k = 0, 1, 2, \ldots$ **do**
  for $i = 0, 1, 2, \ldots$ **do**
    Sample trajectories $\tau_i : \{(s_j, a_j, s'_j, r_j, h_j)\} \sim \pi_\theta$
    Rew. Update $\eta_{k+1} = \eta_k - \zeta_1(k)\nabla_\eta Q(s_t, a_t) \cdot [Q(s_t, a_t) - (r(s_t, a_t) + \gamma Q(s_{t+1}, a_{t+1}))]$
    Cost Update $\kappa_{k+1} = \kappa_k - \zeta_1(k)\nabla_\kappa Q_c(s_t, a_t) \cdot [Q_c(s_t, a_t) - (h(s_t) + \gamma Q_c(s_{t+1}, a_{t+1}))]$
    Policy Update $\theta_{k+1} =$
    $\Gamma_\Theta\Big(\theta_k - \zeta_2(k)\gamma^t\Big[-Q(s_t, a_t)[1 - p(s_t)] + Q_c(s_t, a_t)[\lambda(1 - p(s_t)) + p(s_t)]\Big]\nabla_\theta \log \pi_\theta(a_t|s_t)\Big)$
    REF Update $\xi_{k+1} = \xi_k - \zeta_3(k)\nabla_\xi p(s_t) \cdot [p(s_t) - \max\{\mathbb{1}_{h(s_t)>0}, \gamma p(s_{t+1})\}]$
    Lagrange multiplier Update $\omega_{k+1} = \Gamma_\Omega\big(\omega_k - \zeta_4(k)Q_c(s_t, a_t)(1 - p(s_t))\nabla_\omega \lambda\big)$
  end for
end for

4.3.4 Convergence Analysis

We provide convergence analysis of our algorithm for Finite MDPs (finite bounded state and action space sizes, maximum horizon T, reward bounded by $R_{\max}$, and cost bounded by $H_{\max}$) under reasonable assumptions. We demonstrate our algorithm almost surely finds a locally optimal policy for our RESPO formulation, based on the following assumptions:

- **A1** *(Step size):* Step sizes follow schedules $\{\zeta_1(k)\}$, $\{\zeta_2(k)\}$, $\{\zeta_3(k)\}$, $\{\zeta_4(k)\}$ where:

$$\sum_k \zeta_i(k) = \infty \text{ and } \sum_k \zeta_i(k)^2 < \infty, \forall i \in \{1,2,3,4\}, \quad \text{and} \quad \zeta_j(k) = o(\zeta_{j-1}(k)), \forall j \in \{2,3,4\}.$$

The reward returns and cost returns critic value functions must follow the fastest schedule $\zeta_1(k)$, the policy must follow the second fastest schedule $\zeta_2(k)$, the REF must follow the second slowest schedule $\zeta_3(k)$, and finally, the lagrange multiplier should follow the slowest schedule

$\zeta_4(k)$.

- **A2** *(Strict Feasibility):* $\exists \pi(\cdot|\cdot;\theta)$ such that $\forall s \in \mathcal{S}_I$ where $\phi^*(s) = 0$, $V_c^{\pi_\theta}(s) \leq 0$.
- **A3** *(Differentiability and Lipschitz Continuity):* For all state-action pairs (s,a), we assume value and cost Q functions $Q(s,a;\eta), Q_c(s,a;\kappa)$, policy $\pi(a|s;\theta)$, and REF $p(s,a;\xi)$ are continuously differentiable in $\eta, \kappa, \theta, \xi$ respectively. Furthermore, $\nabla_\omega \lambda_\omega$ and, for all state-action pairs (s,a), $\nabla_\theta \pi(a|s;\theta)$ are Lipschitz continuous functions in ω and θ respectively.

Theorem 2. Given Assumptions **A1**-**A3**, the policy updates in Algorithm 1 will almost surely converge to a locally optimal policy for our proposed optimization in Equation RESPO.

We first provide an intuitive explanation behind why our REF learns to converge to the safest policy's REF, then a proof overview, and then the full proof.

Intuition behind REF convergence

The approach can be explained by considering what happens in the individual regions of space. Consider a deterministic environment for simplicity. As seen in Figure 4.1, there are two subsets of the initial state space: a safest policy's "true" feasible set $\mathcal{S}_{hI}$ and REF predicted feasible set $\mathcal{S}_{pI}$, and they create 4 regions in the initial state space $\mathcal{S}_I$: $\mathcal{W} = \overline{\mathcal{S}_{hI}} \cap \overline{\mathcal{S}_{pI}}$, $\mathcal{X} = \overline{\mathcal{S}_{hI}} \cap \mathcal{S}_{pI}$, $\mathcal{Y} = \mathcal{S}_{hI} \cap \overline{\mathcal{S}_{pI}}$, $\mathcal{Z} = \mathcal{S}_{hI} \cap \mathcal{S}_{pI}$. Consider a point during training when the lagrange multiplier λ is sufficiently large. For states in $\mathcal{W}$, the set of correctly classified infeasible states, the algorithm will simply minimize cumulative violations $V_c^{\pi_\theta}(s)$, and thereby remain as safe as possible since the policy and critics learning rates are faster than that of REF. $\mathcal{X}$, which is the set of infeasible states that are misclassified, is very small if we ensure the policy and REF are trained at much faster time scales than the multiplier and so when the agent starts in true infeasible states, it will by definition reach violations and therefore be labeled as infeasible. In $\mathcal{Y}$, the set of truly feasible states that are misclassified, the algorithm also minimizes cumulative violations, which by the definition of feasibility should be 0. It will then have no violations and enter the correctly predicted feasible set $\mathcal{Z}$. And when starting in states in $\mathcal{Z}$, the algorithm will

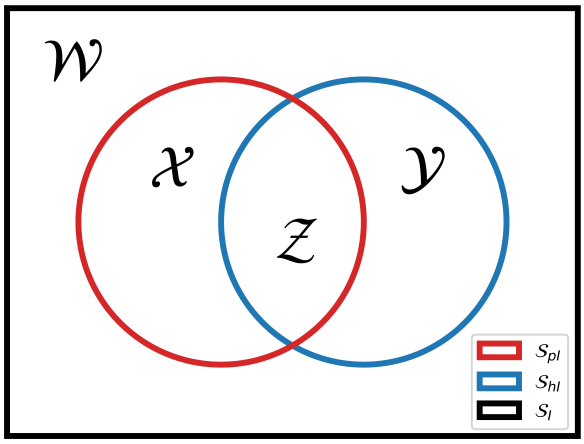

Figure 4.1. The predicted feasible set converges to a safest policy's feasible set since the misclassified regions $\mathcal{X}$ and $\mathcal{Y}$ are corrected over time.

optimize the lagrangian, and since the multiplier λ is sufficiently large, it will converge to a policy that optimizes for reward while ensuring safety, i.e. no future violations, and therefore the state will stay predictably feasible in $\mathcal{Z}$. In this manner, REF's predicted feasible set will converge to the optimal feasible set, and the agent will be safe and have optimal performance in the feasible set and be the safest behavior outside the feasible set. Thereby, the algorithm finds a locally optimal solution to the proposed optimization formulation.

Proof Overview

We show our algorithm convergence to the optimal policy by utilizing the proof framework of multi-time scale presented in [13, 30, 31, 99]. Specifically, we have 4 time scales for (1) the critics, (2) policy, (3) REF function, and (4) lagrange multiplier, listed in order from fastest to slowest. The overview of each timescale proof step is as follows:

1 We demonstrate the almost sure convergence of the critics to the corresponding fixed point optimal critic functions of the policy.

2 Using multi-timescale theory, we demonstrate the policy almost surely converges to a stationary point of a continuous time system, which we show has a Lyapunov function certifying its locally asymptotic stability at the stationary point.

3 We demonstrate the almost sure convergence of the REF function to the REF of the policy that is safe insofar as the lagrange multiplier is sufficiently large.

4 We demonstrate the almost sure convergence of the lagrange multiplier to a stationary point similar to the proof in the policy timecale.

Finally, we demonstrate that the stationary points for the policy and lagrange multiplier form a saddle point, and so by local saddle point theorem we almost surely achieve the locally optimal policy of our proposed optimization.

Proof Details

Proof. **Step 1** (convergence of the critics V_η and V_κ updates): From the multi-time scale assumption, we know that η and κ will convergence on a faster time scale than the other parameters θ, ξ, and ω. Therefore, we can leverage Lemma 1 of Chapter 6 of [13] to analyze the convergence properties while updating η_k and κ_k by treating θ, ξ, and ω as fixed parameters θ_k, ξ_k, and ω_k. In other words, the policy, REF, and lagrange multiplier are fixed while computing $Q^{\pi_{\theta_k}}(s,a)$ and $Q_c^{\pi_{\theta_k}}(s,a)$. With the Finite MDP assumption and policy evaluation convergence results of [86], and assuming sufficiently expressive function approximator (i.e. wide enough neural networks) to ensure convergence to global mininum, we can use the fact that the bellman operators $\mathcal{B}$ and $\mathcal{B}_c$ which are defined as

$$\mathcal{B}[Q](s,a) = r(s,a) + \gamma \mathop{\mathbb{E}}_{s',a' \sim \pi, P(s)}[Q(s',a')]$$

$$\mathcal{B}[Q_c](s,a) = h(s) + \gamma \mathop{\mathbb{E}}_{s',a' \sim \pi, P(s)}[Q_c(s',a')]$$

are γ-contraction mappings, and therefore as k approaches ∞, we can be sure that $Q(s,a;\eta_k) \to Q(s,a;\eta^*) = Q^{\pi_{\theta_k}}(s,a)$ and $Q_c(s,a;\kappa_k) \to Q_c(s,a;\kappa^*) = Q_c^{\pi_{\theta_k}}(s,a)$. So since η_k and κ_k converge to η^* and κ^*, we prove convergence of the critics in Time scale 1.

Step 2 (convergence of the policy π_θ update): Because ξ and ω updated on slower

time scales than θ, we can again use Lemma 1 of Chapter 6 of [13] and treat these parameters are fixed at ξ_k and ω_k respectively when updating θ_k. Additionally in Time scale 2, we have $||Q(s,a;\eta_k) - Q(s,a;\eta^*)|| \to 0$ and $||Q_c(s,a;\kappa_k) - Q_c(s,a;\kappa^*)|| \to 0$ almost surely. Now the update of the policy θ using the gradient from Equation 4.6 is:

$$\begin{aligned}
\theta_{k+1} &= \Gamma_\Theta[\theta_k - \zeta_2(k)(\nabla_\theta L(\theta, \xi_k, \omega_k)|_{\theta=\theta_k})] \\
&= \Gamma_\Theta[\theta_k - \zeta_2(k)[\gamma^t[-Q_\eta(s_t,a_t)[1-p_{\xi_k}(s_t)] \\
&+ Q_c(s_t,a_t)[\lambda_\omega(1-p_{\xi_k}(s_t)) + p_{\xi_k}(s_t)]]\nabla_\theta \log \pi(a_t|s_t;\theta)|_{\theta=\theta_k}]] \\
&= \Gamma_\Theta[\theta_k - \zeta_2(k)(\nabla_\theta L(\theta, \xi_k, \omega_k)|_{\theta=\theta_k,\eta=\eta^*,\kappa=\kappa^*} + \delta\theta_{k+1} + \delta\theta_\varepsilon)]
\end{aligned}$$

where

$$\begin{aligned}
\delta\theta_{k+1} = \sum_{s_i,a_i} \Big[& d_0(s_0) P^{\pi_{\theta_k}}(s_i,a_i|s_0)\gamma^i[-Q_\eta(s_i,a_i)[1-p_{\xi_k}(s_i)] \\
& + Q_c(s_i,a_i)[\lambda_\omega(1-p_{\xi_k}(s_i)) + p_{\xi_k}(s_i)]]\nabla_\theta \log \pi(a_i|s_i;\theta)|_{\theta=\theta_k}\Big] \\
& - \gamma^t[-Q_\eta(s_t,a_t)[1-p_{\xi_k}(s_t)] + Q_c(s_t,a_t)[\lambda_\omega(1-p_{\xi_k}(s_t)) + p_{\xi_k}(s_t)]] \\
& \cdot \nabla_\theta \log \pi(a_t|s_t;\theta)|_{\theta=\theta_k}
\end{aligned}$$

and

$$\begin{aligned}
\delta\theta_\varepsilon = \sum_{s_i,a_i} d_0(s_0) P^{\pi_{\theta_k}}(s_i,a_i|s_0) \Big[& \\
- \gamma^i[-Q(s_i,a_i;\eta_k)[1-p_{\xi_k}(s_i)] + Q_c(s_i,a_i;\kappa_k)[\lambda_\omega(1-p_{\xi_k}(s_i)) + p_{\xi_k}(s_i)]] & \\
\cdot \nabla_\theta \log \pi(a_i|s_i;\theta)|_{\theta=\theta_k} & \\
+ \gamma^i[-Q^{\pi_{\theta_k}}(s_i,a_i)[1-p_{\xi_k}(s_i)] + Q_c^{\pi_{\theta_k}}(s_i,a_i)[\lambda_\omega(1-p_{\xi_k}(s_i)) + p_{\xi_k}(s_i)]] & \\
\cdot \nabla_\theta \log \pi(a_i|s_i;\theta)|_{\theta=\theta_k}\Big] &
\end{aligned}$$

Lemma 1: We can first demonstrate that $\delta\theta_{k+1}$ is square integrable. In particular,

$$\begin{aligned}
&\mathbb{E}[||\delta\theta_{k+1}||^2|\mathcal{F}_{\theta,k}] \\
&\leq 2||\nabla_\theta \log\pi(a|s;\theta)|_{\theta=\theta_k}\mathbb{1}_{\pi(a|s;\theta_k)>0}||_\infty^2 \cdot \Bigg(||Q(s,a;\eta_k)||_\infty^2 \cdot ||1-p_{\xi_k}(s)||_\infty^2 \\
&+ ||Q_c(s,a;\kappa_k)||_\infty^2 \cdot \Big[||\lambda_\omega||_\infty^2 \cdot ||1-p_{\xi_k}(s)||_\infty^2 + ||p_{\xi_k}(s)||_\infty^2\Big]\Bigg) \\
&\leq 2\frac{||\nabla_\theta \pi(a|s;\theta)|_{\theta=\theta_k}||_\infty^2}{\min\{\pi(a|s;\theta_k)|\pi(a|s;\theta_k)>0\}} \cdot \Bigg(||Q(s,a;\eta_k)||_\infty^2 \cdot ||1-p_{\xi_k}(s)||_\infty^2 \\
&+ ||Q_c(s,a;\kappa_k)||_\infty^2 \cdot \Big[||\lambda_\omega||_\infty^2 \cdot ||1-p_{\xi_k}(s)||_\infty^2 + ||p_{\xi_k}(s)||_\infty^2\Big]\Bigg)
\end{aligned}$$

Note that $\mathcal{F}_{\theta,k} = \sigma(\theta_m, \delta\theta_m, m \leq k)$ is the filtration for θ_k generated by different independent trajectories [30]. Also note that the indicator function is used because the expectation of $||\delta\theta_{k+1}||^2$ is taken with respect to $P^{\pi_{\theta_k}}$ and $P^{\pi_{\theta_k}}(s,a|s_0) = 0$ if $\pi(a|s;\theta_k) = 0$. From the Assumptions on Lipschitz continuity and Finite MDPs reward and costs, we can bound the values of the functions and the gradients of functions. Specifically

$$\begin{aligned}
||\nabla_\theta \pi(a|s;\theta)|_{\theta=\theta_k}||_\infty^2 &\leq K_1(1+||\theta_k||_\infty^2), \\
||Q(s,a;\eta_k)||_\infty^2 &\leq \frac{R_{\max}}{1-\gamma}, \\
||Q_h(s,a;\kappa_k)||_\infty^2 &\leq \frac{H_{\max}}{1-\gamma}, \\
||\lambda_\omega||_\infty^2 &\leq \lambda_{\max}, \\
||1-p_{\xi_k}(s)||_\infty^2 &\leq 1, \\
||p_{\xi_k}(s)||_\infty^2 &\leq 1
\end{aligned}$$

where K_1 is a Lipschitz constant. Furthermore, note that because we are sampling, $\pi(a|s;\theta_k)$ will take on only a finite number of values, so its nonzero values will be bounded away from

zero. Thus we can say

$$\frac{1}{\min\{\pi(a|s;\theta_k)|\pi(a|s;\theta_k)>0\}} \leq K_2$$

for some large enough K_2. Thus using the bounds from these conditions, we can demonstrate

$$\mathbb{E}[||\delta\theta_{k+1}||^2|\mathcal{F}_{\theta,k}] \leq 2\cdot K_1(1+||\theta_k||_\infty^2)\cdot K_2(\frac{R_{\max}}{1-\gamma}\cdot 1+\frac{H_{\max}}{1-\gamma}\cdot(\lambda_{\max}\cdot 1+1)) < \infty$$

Therefore $\delta\theta_{k+1}$ is square integrable.

Lemma 2: Secondly, we can demonstrate $\delta\theta_\varepsilon \to 0$.

$$\begin{aligned}
\delta\theta_\varepsilon &= \sum_{s_i,a_i} d_0(s_0)P^{\pi_{\theta_k}}(s_i,a_i|s_0)\Big[\gamma^i\big[(Q(s_i,a_i;\eta_k)-Q^{\pi_{\theta_k}}(s_i))[1-p_{\xi_k}(s_i)] \\
&+(-Q_c(s_i,a_i;\kappa_k)+Q_c^{\pi_{\theta_k}}(s_i,a_i))[\lambda_\omega(1-p_{\xi_k}(s_i))+p_{\xi_k}(s_i)]\big]\nabla_\theta\log\pi(a_i|s_i;\theta)|_{\theta=\theta_k}\Big] \\
&\leq \sum_{s_i,a_i} d_0(s_0)P^{\pi_{\theta_k}}(s_i,a_i|s_0)\Big[\gamma^i\big[(Q(s_i,a_i;\eta_k)-Q(s_i,a_i;\eta^*))[1-p_{\xi_k}(s_i)] \\
&+(-Q_c(s_i,a_i;\kappa_k)+Q_c(s_i,a_i;\kappa^*))[\lambda_\omega(1-p_{\xi_k}(s_i))+p_{\xi_k}(s_i)]\big]\nabla_\theta\log\pi(a_i|s_i;\theta)|_{\theta=\theta_k}\Big] \\
&\leq \sum_{s_i,a_i} d_0(s_0)P^{\pi_{\theta_k}}(s_i,a_i|s_0)\Big[\gamma^i\big[||Q(s_i,a_i;\eta_k)-Q(s_i,a_i;\eta^*)||[1-p_{\xi_k}(s_i)] \\
&+||-Q_c(s_i,a_i;\kappa_k)+Q_c(s_i,a_i;\kappa^*)||[\lambda_\omega(1-p_{\xi_k}(s_i))+p_{\xi_k}(s_i)]\big]\nabla_\theta\log\pi(a_i|s_i;\theta)|_{\theta=\theta_k}\Big]
\end{aligned}$$

And because we have $||Q(s,a;\eta_k)-Q(s,a;\eta^*)|| \to 0$ and $||Q_c(s,a;\kappa_k)-Q_c(s,a;\kappa^*)|| \to 0$ almost surely, we can therefore say $\delta\theta_\varepsilon \to 0$.

Lemma 3: Finally, since $\hat{\nabla}_\theta J_\pi(\theta)|_{\theta=\theta_k}$ is a sample of $\nabla_\theta L(\theta,\xi_k,\omega_k)|_{\theta=\theta_k}$ based on the history of sampled trajectories, we conclude that $\mathbb{E}[\delta\theta_{k+1}|\mathcal{F}_{\theta,k}]=0$.

From the 3 above lemmas, the policy θ update is a stochastic approximation of a

continuous system $\theta(t)$ defined by [13]

$$\dot{\theta} = \Upsilon_{\Theta}[-\nabla_{\theta} L(\theta, \xi, \omega)] \tag{4.7}$$

in which

$$\Upsilon_{\Theta}[M(\theta] \stackrel{\Delta}{=} \lim_{0<\psi\to 0} \frac{\Gamma_{\Theta}(\theta + \psi M(\theta)) - \Gamma_{\Theta}(\theta)}{\psi}$$

or in other words the left directional derivative of $\Gamma_{\Theta}(\theta)$ in the direction of $M(\theta)$. Using the left directional derivative $\Upsilon_{\Theta}[-\nabla_{\theta} L(\theta, \xi, \omega)]$ in the gradient descent algorithm for learning the policy π_{θ} ensures the gradient will point in the descent direction along the boundary of Θ when the θ update hits its boundary. Using Step 2 in Appendix A.2 from [30], we have that $dL(\theta, \xi, \omega)/dt = -\nabla_{\theta} L(\theta, \xi, \omega)^T \cdot \Upsilon_{\Theta}[-\nabla_{\theta} L(\theta, \xi, \omega)] \leq 0$ and the value is non-zero if $||\Upsilon_{\Theta}[-\nabla_{\theta} L(\theta, \xi, \omega)]|| \neq 0$. Now consider the continuous system $\theta(t)$. For some fixed ξ and ω, define a Lyapunov function

$$\mathcal{L}_{\xi,\omega}(\theta) = L(\theta, \xi, \omega) - L(\theta^*, \xi, \omega)$$

where θ^* is a local minimum point. Then there exists a ball centered at θ^* with a radius ρ such that $\forall \theta \in \mathfrak{B}_{\theta^*}(\rho) = \{\theta | ||\theta - \theta^*|| \leq \rho\}$, $\mathcal{L}_{\xi,\omega}(\theta)$ is a locally positive definite function, that is $\mathcal{L}_{\xi,\omega}(\theta) \geq 0$. Using Proposition 1.1.1 from [11], we can show that $\Upsilon_{\Theta}[-\nabla_{\theta} L(\theta, \xi, \omega)]|_{\theta=\theta^*} = 0$ meaning θ^* is a stationary point. Since $dL(\theta, \xi, \omega)/dt \leq 0$, through Lyapunov theory for asymptotically stable systems presented in Chapter 4 of [60], we can use the above arguments to demonstrate that with any initial conditions of $\theta(0) \in \mathfrak{B}_{\theta^*}(\rho)$, the continuous state trajectory of $\theta(t)$ converges to θ^*. Particularly, $L(\theta^*, \xi, \omega) \leq L(\theta(t), \xi, \omega) \leq L(\theta(0), \xi, \omega)$ for all $t > 0$.

Using these aforementioned properties, as well as the facts that 1) $\nabla_{\theta} L(\theta, \xi, \omega)$ is a Lipschitz function (using Proposition 17 from [30]), 2) the step-sizes of Assumption on steps sizes, 3) $\delta\theta_{k+1}$ is a square integrable Martingale difference sequence and $\delta\theta_{\varepsilon}$ is a vanishing

error almost surely, and 4) $\theta_k \in \Theta, \forall k$ implying that $\sup_k ||\theta_k|| < \infty$ almost surely, we can invoke Theorem 2 of chapter 6 in [13] to demonstrate the sequence $\{\theta_k\}, \theta_k \in \Theta$ converges almost surely to the solution of the ODE defined by Equation 4.7, which additionally converges almost surely to the local minimum $\theta^* \in \Theta$.

Step 3 (convergence of REF p_ξ updates): Since ω is updated on a slower time scale that ξ, we can again treat ω as a fixed parameter at ω_k when updating ξ. Furthermore, in Time scale 3, we know that the policy has converged to a local minimum, particularly $||\theta_k - \theta^*(\xi_k, \omega_k)|| = 0$. Now the bellman operator for REF is defined by

$$\mathcal{B}_p[p](s) = \max\{\mathbb{1}_{s \in S_v}, \gamma \mathop{\mathbb{E}}_{s' \sim \pi, P(s)}[p(s')]\}.$$

We demonstrate this is a γ contraction mapping as follows:

$$\begin{aligned}
&|\mathcal{B}_p[p](s) - \mathcal{B}_p[\hat{p}](s)| \\
&= |\max\{\mathbb{1}_{s \in S_v}, \gamma \mathop{\mathbb{E}}_{s' \sim \pi, P(s)}[p(s')]\} - \max\{\mathbb{1}_{s \in S_v}, \gamma \mathop{\mathbb{E}}_{s' \sim \pi, P(s)}[\hat{p}(s')]\}| \\
&\leq |\gamma \mathop{\mathbb{E}}_{s' \sim \pi, P(s)}[p(s')] - \gamma \mathop{\mathbb{E}}_{s' \sim \pi, P(s)}[\hat{p}(s')]| \\
&= \gamma |\mathop{\mathbb{E}}_{s' \sim \pi, P(s)}[p(s') - \hat{p}(s')]| \\
&\leq \gamma \sup_s |p(s) - \hat{p}(s)| = \gamma ||p - \hat{p}||_\infty
\end{aligned}$$

So we can say that $p(s; \xi_k)$ will converge to $p(s; \xi^*)$ as $k \to \infty$ under the same assumptions of the Finite MDP and function approximator expressiveness in Step 1. Therefore, π_{θ_k} will also converge to $\pi^\diamond = \pi_{\theta^*(\xi^*, \omega_k)}$ as $k \to \infty$. And because π_θ is the sampling policy used to compute p, $p(s; \xi^*) = p^{\pi_{\theta^*(\xi^*, \omega_k)}}(s; \xi^*) = p^\diamond(s)$.

Notice that $\pi^\diamond$ is a locally minimum optimal policy for the following optimization (recall

λ_ω is treated as constant in this timescale):

$$\min_{\pi} \mathop{\mathbb{E}}_{s\sim d_0} \mathop{\mathbb{E}}_{a\sim\pi(\cdot|s)} \Big[-Q^\pi(s,a)\cdot[1-p^\diamond(s)] + Q_c^\pi(s,a)\cdot[(1-p^\diamond(s))\lambda_\omega + p^\diamond(s)] \Big]$$

and therefore also locally minimum optimal policy for optimization:

$$\min_{\pi} \mathop{\mathbb{E}}_{s\sim d_0} \mathop{\mathbb{E}}_{a\sim\pi(\cdot|s)} \Big[-Q^\pi(s,a) + Q_c^\pi(s,a)\cdot[\lambda_\omega + \frac{p^\diamond(s)}{(1-p^\diamond(s))}] \Big], \text{ if } p^\diamond(s) > 0$$
$$\mathop{\mathbb{E}}_{s\sim d_0} \min_{\pi} \mathop{\mathbb{E}}_{a\sim\pi(\cdot|s)} \Big[Q_c^\pi(s,a) \Big], \text{ if } p^\diamond(s) = 0$$

Since $\frac{p^\diamond(s)}{(1-p^\diamond(s))} \geq 0$, and the Q functions are always nonnegative, we can know that $\pi^\diamond$ is at least as safe as (i.e., its expected cumulative cost is at most that of) a locally optimal policy for the optimization:

$$\min_{\pi} \mathop{\mathbb{E}}_{s\sim d_0} \mathop{\mathbb{E}}_{a\sim\pi(\cdot|s)} \Big[-Q^\pi(s,a) + Q_c^\pi(s,a)\lambda_\omega \Big] \tag{4.8}$$

As λ_ω approaches $\lambda_{\max}$, which in turn approaches ∞, the local minimum optimal policies of Equation 4.8 approach those of the optimization $\pi^\triangle = \arg\min_\pi \mathbb{E}_{s\sim d_0} \mathbb{E}_{a\sim\pi(\cdot|s)} Q_c^\pi(s,a)\lambda_\omega = \arg\min_\pi \mathbb{E}_{s\sim d_0} \mathbb{E}_{a\sim\pi(\cdot|s)} Q_c^\pi(s,a)$. Therefore, the feasible set of the REF $p^\diamond$ will approach that of the REF $p^{\pi^\triangle}$.

Step 4 (convergence of lagrange multiplier λ_ω update): Since λ_ω is on the slowest time scale, we have that $||\theta_k - \theta^*(\omega)|| = 0$, $||\xi_k - \xi^*(\omega)|| = 0$, and $||Q_c(s,a;\kappa_k) - Q_c^{\pi_{\theta_k}}(s,a)|| = 0$ almost surely. Furthermore, due to the continuity of $\nabla_\omega L(\theta,\xi,\omega)$, we have that $||\nabla_\omega L(\theta,\xi,\omega)|_{\theta=\theta_k,\xi=\xi_k,\omega=\omega_k} - \nabla_\omega L(\theta,\xi,\omega)|_{\theta=\theta^*(\omega_k),\xi=\xi^*(\omega_k),\omega=\omega_k}|| = 0$ almost surely. The

update of the multiplier using the gradient for Equation is:

$$\begin{aligned}\omega_{k+1} &= \Gamma_\Omega[\omega_k + \zeta_4(k)(\nabla_\omega L(\theta,\xi,\omega)|_{\theta=\theta_k,\xi=\xi_k,\omega=\omega_k})] \\ &= \Gamma_\Omega[\omega_k + \zeta_4(k)(Q_c(s_t,a_t;\kappa_k)[1-p(s_t;\xi_k)]\nabla_\omega \lambda_\omega|_{\omega=\omega_k})] \\ &= \Gamma_\Omega[\omega_k + \zeta_4(k)(\nabla_\omega L(\theta,\xi,\omega)|_{\theta=\theta^*(\omega_k),\xi=\xi^*(\omega_k),\omega=\omega_k} + \delta\omega_{k+1})]\end{aligned}$$

where

$$\begin{aligned}\delta\omega_{k+1} &= -\nabla_\omega L(\theta,\xi,\omega)|_{\theta=\theta^*(\omega_k),\xi=\xi^*(\omega_k),\omega=\omega_k} + Q_c(s_t,a_t;\kappa_k)[1-p(s_t;\xi_k)]\nabla_\omega\lambda_\omega|_{\omega=\omega_k} \\ &= -\sum_{s_i,a_i} d_0(s_0)P^{\pi_{\theta_k}}(s_i,a_i|s_0)[Q_c^{\pi_{\theta^*}}(s_i,a_i)[1-p_{\xi^*}(s_i)]\nabla_\omega\lambda_\omega|_{\omega=\omega_k}] \\ &+ Q_c(s_t,a_t;\kappa_k)[1-p(s_t;\xi_k)]\nabla_\omega\lambda_\omega|_{\omega=\omega_k} \\ &= -\sum_{s_i,a_i} d_0(s_0)P^{\pi_{\theta_k}}(s_i,a_i|s_0)[Q_c^{\pi_{\theta^*}}(s_i,a_i)[1-p_{\xi^*}(s_i)]\nabla_\omega\lambda_\omega|_{\omega=\omega_k}] \\ &+ [Q_c(s_t,a_t;\kappa_k)[1-p(s_t;\xi_k)] - Q_c^{\pi_{\theta_k}}(s_t,a_t)[1-p(s_t;\xi_k)] + \\ &Q_c^{\pi_{\theta_k}}(s_t,a_t)[1-p(s_t;\xi_k)] - Q_c^{\pi_{\theta_k}}(s_t,a_t)[1-p^\diamond(s_t)] + \\ &Q_c^{\pi_{\theta_k}}(s_t,a_t)[1-p^\diamond(s_t)]]\nabla_\omega\lambda_\omega|_{\omega=\omega_k} \\ &= -\sum_{s_i,a_i} d_0(s_0)P^{\pi_{\theta_k}}(s_i,a_i|s_0)[Q_c^{\pi_{\theta^*}}(s_i,a_i)[1-p_{\xi^*}(s_i)]\nabla_\omega\lambda_\omega|_{\omega=\omega_k}] \\ &+ [(Q_c(s_t,a_t;\kappa_k) - Q_c^{\pi_{\theta_k}}(s_t,a_t))[1-p(s_t;\xi_k)] + \\ &Q_c^{\pi_{\theta_k}}(s_t,a_t)[p^\diamond(s_t) - p(s_t;\xi_k)] + \\ &Q_c^{\pi_{\theta_k}}(s_t,a_t)[1-p^\diamond(s_t)]]\nabla_\omega\lambda_\omega|_{\omega=\omega_k}]\end{aligned}$$

Now, just as in the θ update convergence, we can demonstrate the following lemmas:

Lemma 4: $\delta\omega_{k+1}$ is square integrable since

$$\mathbb{E}[||\delta\omega_{k+1}||^2|\mathcal{F}_{\omega,k}] \leq 2\cdot\frac{H_{\max}}{1-\gamma}\cdot 1\cdot K_3(1+||\omega_k||_\infty^2) < \infty$$

for some large Lipschitz constant K_3. Note that $\mathcal{F}_{\omega,k} = \sigma(\omega_m, \delta\omega_m, m \leq k)$ is the filtration for ω_k generated by different independent trajectories [30].

Lemma 5: Because $||Q_c(s_t, a_t; \kappa_k) - Q_c^{\pi_{\theta_k}}(s_t, a_t)||_\infty \to 0$ and $||p^\diamond(s_t) - p(s_t; \xi_k)||_\infty \to 0$ and $Q_c^{\pi_{\theta_k}}(s_t, a_t)[1 - p_{\xi^*}(s_t)]\nabla_\omega \lambda_\omega|_{\omega=\omega_k}$ is a sample of $Q_c^{\pi_{\theta^*}}(s_i, a_i)[1 - p_{\xi^*}(s_i)]\nabla_\omega \lambda_\omega|_{\omega=\omega_k}$, we conclude that $\mathbb{E}[\delta\omega_{k+1}|\mathcal{F}_{\omega,k}] = 0$ almost surely.

Thus, the lagrange multiplier ω update is a stochastic approximation of a continuous system $\omega(t)$ defined by [13]

$$\dot{\omega} = \Upsilon_\Omega[-\nabla_\omega L(\theta, \xi, \omega)|_{\theta=\theta^*(\omega), \xi=\xi^*(\omega)}] \tag{4.9}$$

with Martingale difference error of $\delta\omega_k$ and where Υ_Ω is the left direction derivative defined similar to that in Time scale 2 of the convergence of θ update. Using Step 2 in Appendix A.2 from [30], we have that

$$\begin{aligned} &dL(\theta^*(\omega), \xi^*(\omega), \omega)/dt \\ =&\nabla_\omega L(\theta, \xi, \omega)|_{\theta=\theta^*(\omega), \xi=\xi^*(\omega)}{}^T \cdot \Upsilon_\Omega[\nabla_\Omega L(\theta, \xi, \omega)|_{\theta=\theta^*(\omega), \xi=\xi^*(\omega)}] \geq 0 \end{aligned}$$

and the value is non-zero if

$$||\Upsilon_\Omega[\nabla_\omega L(\theta, \xi, \omega)|_{\theta=\theta^*(\omega), \xi=\xi^*(\omega)}]|| \neq 0.$$

For a local maximum point ω^*, define a Lyapunov function as

$$\mathcal{L}(\omega) = L(\theta^*(\omega), \xi^*(\omega), \omega^*) - L(\theta^*(\omega), \xi^*(\omega), \omega)$$

Then there exists a ball centered at ω^* with a radius ρ' such that $\forall\omega \in \mathfrak{B}_{\omega^*}(\rho') = \{\omega | ||\omega - \omega^*|| \leq \rho'\}$, $\mathcal{L}(\omega)$ is a locally positive definite function, that is $\mathcal{L}(\omega) \geq 0$. Also, $d\mathcal{L}(\omega(t))/dt = -dL(\theta^*(\omega), \xi^*(\omega), \omega)/dt \leq 0$ and is equal only when

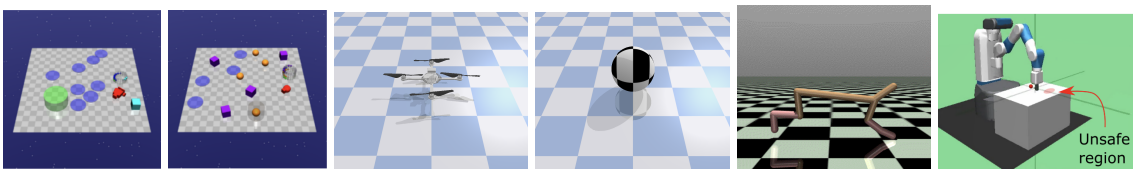

Figure 4.2. We compare the performance of our algorithm with other SOTA baselines in Safety Gym (left two figures), Safety PyBullet (middle two figures), and Safety MuJoCo (right two figures).

$\Upsilon_\Omega[\nabla_\omega L(\theta,\xi,\omega)|_{\theta=\theta^*(\omega),\xi=\xi^*(\omega)}] = 0$, so therefore ω^* is a stationary point. By leveraging Lyapunov theory for asymptotically stable systems presented in Chapter 4 of [60] we can demonstrate that for any initial conditions of $\omega \in \mathfrak{B}_{\omega^*}(\rho')$, the continuous state trajectory of $\omega(t)$ converges to the locally maximum point ω^*.

Using these aforementioned properties, as well as the facts that 1) $\nabla_\omega L(\theta^*(\omega),\xi^*(\omega),\omega)$ is a Lipschitz function, 2) the step-sizes of Assumption on steps sizes, 3) $\{\omega_{k+1}\}$ is a stochastic approximation of $\omega(t)$ with a Martingale difference error, and 4) convex and compact properties in projections used, we can use Theorem 2 of chapter 6 in [13] to demonstrate the sequence $\{\omega_k\}$ converges almost surely to a locally maximum point ω^* almost surely, that is $L(\theta^*(\omega),\xi^*(\omega),\omega^*) \geq L(\theta^*(\omega),\xi^*(\omega),\omega)$.

From Time scales 2 and 3 we have that $L(\theta^*(\omega),\xi^*(\omega),\omega) \leq L(\theta,\xi,\omega)$ while from Time scale 4 we have that $L(\theta^*(\omega),\xi^*(\omega),\omega^*) \geq L(\theta^*(\omega),\xi^*(\omega),\omega)$. Thus, $L(\theta^*(\omega),\xi^*(\omega),\omega) \leq L(\theta^*(\omega),\xi^*(\omega),\omega^*) \leq L(\theta,\xi,\omega^*)$. Therefore, $(\theta^*,\xi^*,\omega^*)$ is a local saddle point of (θ,ξ,ω). Invoking the saddle point theorem of Proposition 5.1.6 in [11], we can conclude that $\pi(\cdot|\cdot;\theta^*)$ is a locally optimal policy for our proposed optimization formulation. □

4.4 Experiments

Baselines. The baselines we compare are CMDP-based or solve for hard constraints. The CMDP baselines are Lagrangian-based Proximal Policy Optimization (**PPOLag**) based on [88], Constraint-Rectified Policy Optimization (**CRPO**) [96], Penalized Proximal Policy Optimization (**P3O**) [101], and Projection-Based Constrained Policy Optimization (**PCPO**) [98]. The hard constraints baselines are **RCRL** [99], **CBF** with constraint $\dot{h}(s) + \nu \cdot h(s) \leq 0$, and Feasibile

Actor-Critic (**FAC**) [67]. We classify **FAC** among the hard constraint approaches because we make its cost threshold $\chi = 0$ in order to better compare using NN lagrange multiplier with our REF approach in **RESPO**. We include the unconstrained Vanilla **PPO** [81] baseline for reference.

Benchmarks. We compare **RESPO** with the baselines in a diverse suite of safety environments. We consider high-dimensional environments in Safety Gym [77] (namely Point-Button and CarGoal), Safety PyBullet [47] (namely DroneCircle and BallRun), and Safety MuJoCo [91], (namely Safety HalfCheetah and Reacher). We also show our algorithm in a multi-drone environment with *multiple hard and soft constraints*.

4.4.1 Main Experiments in Safety Gym, Safety PyBullet, and MuJoCo

We compare our algorithm with SOTA benchmarks on various high-dimensional (up to 76D observation space), complex environments in the stochastic setting, i.e., where the environment and/or policy are stochastic. Particularly, we examine environments in Safety Gym, Safety PyBullet, and Safety MuJoCo. The environments provide reward for achieving a goal behavior or location, while the cost is based on tangible (e.g., avoiding quickly moving objects) and non-tangible (e.g., satisfying speed limit) constraints. Environments like PointButton require intricate behavior where specific buttons must be reached while avoiding multiple moving obstacles, stationary hazards, and wrong buttons.

Overall, **RESPO** achieves the best balance between optimizing reward and minimizing cost violations across all the environments. Specifically, our approach generally has the highest reward performance (see the red lines from the top row of Figure 4.3) among the safety-constrained algorithms while maintaining reasonably low to 0 cost violations (like in HalfCheetah in Figure 4.5). When **RESPO** performs the second highest, the highest-performing safety algorithm always incurs several times more violations than **RESPO** – for instance, **RCRL** in PointButton or **PPOLag** in Drone Circle. Non-primal-dual CMDP approaches, namely **CRPO**, **P3O**, and **PCPO** generally satisfy their cost threshold constraints, but their reward performances rarely

exceed that of **PPOLag**. **RCRL** generally has extremes of high reward and high cost, like in BallRun, or low reward and low cost, like in CarGoal. **FAC** and **CBF** generally have conservative behavior that sacrifices reward performance to minimize cost.

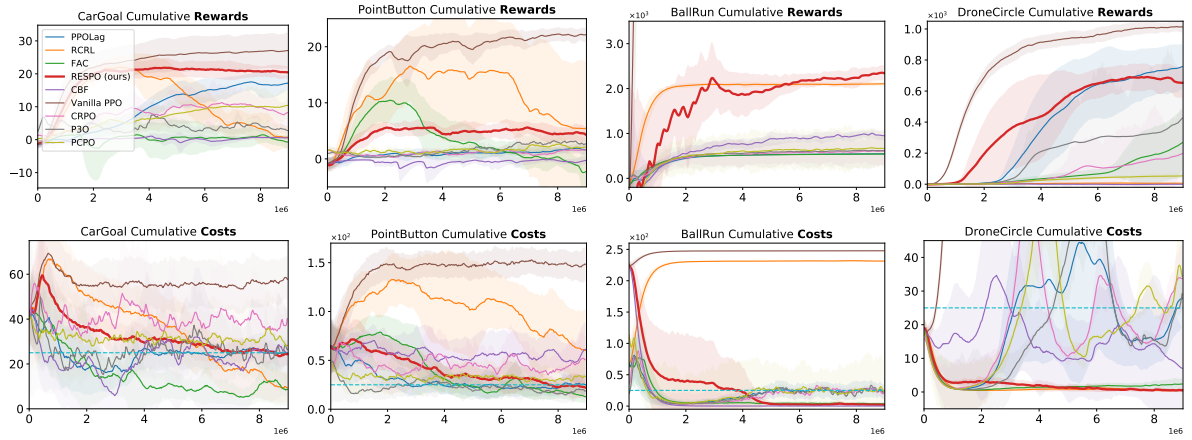

Figure 4.3. Comparison of RESPO with baselines in Safety Gym and PyBullet environments. The plots in the first row show performance measured in rewards (higher is better); those in second row show cost (lower is better). RESPO (red curves) generally achieves the best balance of maximizing reward and minimizing cost.

4.4.2 Hard and Soft Constraints

We also demonstrate **RESPO**'s performance in an environment with multiple hard and soft constraints. The environment requires controlling two drones to pass through a tunnel one at a time while respecting certain distance requirements. The reward is given for quickly reaching the goal positions. The two hard constraints involve **(H1)** ensuring neither drone collides into the wall and **(H2)** the distance between the two drones is more than 0.5 to ensure they do not collide. The soft constraint is that the two drones are within 0.8 of each other to ensure real-world communication. It is preferable to prioritize hard constraint **H1** over hard constraint **H2**, since colliding with the wall may have more serious consequences to the drones rather than violations of an overly precautious distance constraint.

Our approach, in the leftmost of Figure 4.4, successfully reaches the goal while avoiding the wall obstacles in all time steps. We are able to prioritize this wall avoidance constraint over

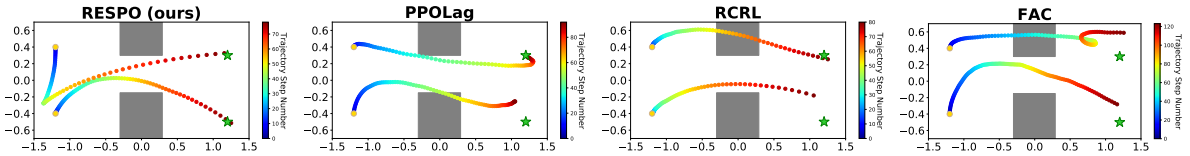

Figure 4.4. Comparison of algorithms in Hard & Soft Constraints multi-Drone control. Starting at gold circles, drones must enter the tunnel one at a time and reach green stars. Trajectory colors correspond to time. RESPO reaches goal, satisfies hard constraints, and usually respects soft constraints.

the second hard constraint. This can be seen particularly in between the blue to cyan time period where the higher Drone makes way for the lower Drone to pass through but needs to make a drop to make a concave parabolic trajectory to the goal. Nonetheless, the hard constraints are almost always satisfied, thereby producing the behavior of allowing one drone through the tunnel at a time. The soft constraints are satisfied at the beginning and end but are violated, reasonably, in the middle of the episode since only one drone can pass through the tunnel at a time, thereby forcing the other drone into a standby mode.

4.4.3 Ablation Studies

We also perform ablation studies to experimentally confirm the design choices we made based on the theoretically established convergence and optimization framework. We particularly investigate the effects of changing the learning rate of our reachability function as well as changing the optimization framework. We present the results of changing the learning rate for REF in Figure 4.6 while our results for the ablation studies on our optimization framework can be seen in Figure 4.7.

In Figure 4.6, we show the effects of making the learning rate of REF slower and faster than the one we use in accordance with Assumption 1. From these experiments, changing the learning rate in either direction produces poor reward performance. A fast learning rate makes the REF converge to the likelihood of infeasibility for the current policy, which can be suboptimal. But a very slow learning rate means the function takes too long to converge – the lagrange multiplier may meanwhile become very large, thus making it too difficult to optimize for reward

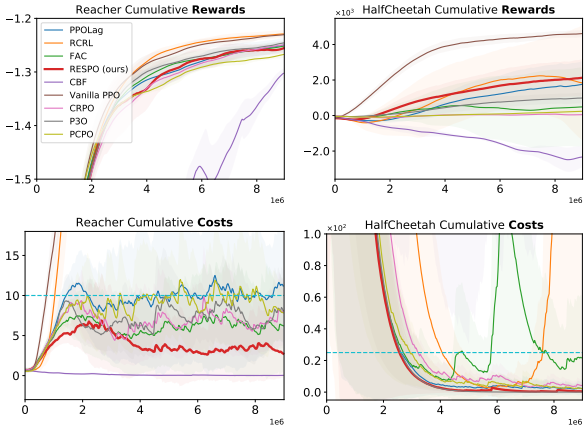

Figure 4.5. Comparison of RESPO with baselines in MuJoCo. Higher rewards (first row plots) and lower costs (second row plots) are better. In HalfCheetah, RESPO has highest reward among safety baselines, with 0 violations. In Reacher, RESPO has good rewards, low costs.

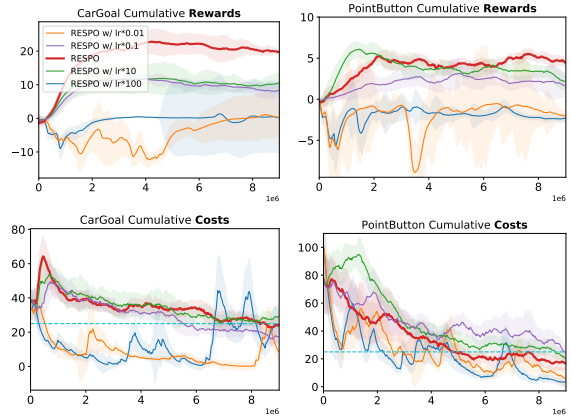

Figure 4.6. Ablation study on the learning rate of REF. Higher rewards (first row plots) are better; lower costs (second row plots) are better. When changing REF's learning rate to violate timescale assumptions, REF produces suboptimal feasible sets.

returns. In both scenarios, the algorithm with modified learning rates produces conservative behavior that sacrifices reward performance.

In Figure 4.7, we compare **RESPO** with RCRL implemented with our REF and **PPOLag** in the CMDP framework with cost threshold $\chi = 0$ to ensure hard constraint satisfaction. The difference between **RESPO** and the RCRL-based ablation approach is that the ablation still uses V_h^π instead of V_c^π. The ablation aproach's high cumulative cost can be attributed to the limitations of using V_h^π – particularly, the lower sensitivity of V_h^π to safety improvement and its lack of guarantees on feasible set (re)entrance. **PPOLag** with $\chi = 0$ produces low violations but also very low reward performance that's close to zero. Naively using V_c^π in a hard constraints framework leads to very conservative behavior that sacrifices reward performance. Ultimately, this ablation study experimentally highlights the importance of learning our REF *and* using value function V_c^π in our algorithm's design.

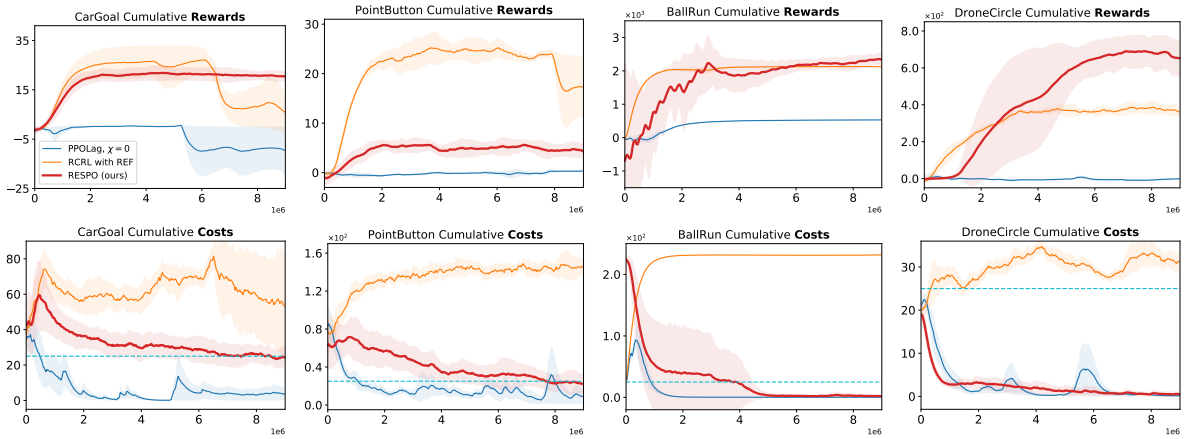

Figure 4.7. Ablation study on optimization framework. Top row plots show performance measured in reward (higher is better). Bottom row plots show cost (lower is better). This demonstrates both REF and V_c^π are crucial in our design and work in tandem to contribute to RESPO's efficacy.

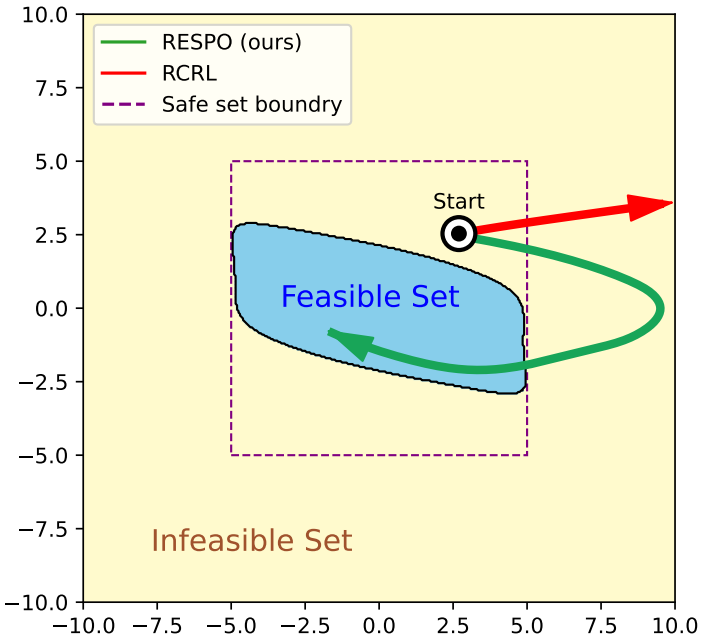

Figure 4.8. Comparison of the trajectories in the Double Integrator Environment of an agent controlled by RCRL (in red) and our proposed algorithm RESPO (in green) when starting from the safe but infeasible set. Our approach actively enters the feasible set (blue region), while RCRL fails to do so.

4.4.4 Double Integrator

We use the Double Integrator environment as a motivating example to demonstrate how performing constrained optimization using solely reachability-based value functions as in **RCRL** can produce nonoptimal behavior when the agent is outside the feasiblity set. Double Integrator has a 2 dimensional observation space $[x_1, x_2]$, 1 dimension action space $a \in [-0.5, 0.5]$, system dynamics is $\dot{s} = [x_2, a]$, and constraint as $||s||_\infty \leq 5$. Particularly, we make the cost as 1 if $||s||_\infty > 5$, and 0 otherwise to emphasize the importance of capturing the frequency of violation during training.

We train an **RCRL** controller and **RESPO** controller in this environment, and the results are visualized in Figure 4.8. The color scheme indicates the learned reachability value across the state space while the black line demarcates the border of the zero level set. We present the behavior of the trajectories of **RCRL** and **RESPO**. Because the **RCRL** optimizes for reachability value function when outside the feasible set, it simply minimizes the maximum violation, which as can be seen does not result in the agent reentering the feasible set since it is uniformly equal to or near 1 in the infeasible set. This is since it permits many violations of magnitude same or less than that of the maximum violation. On the other hand, **RESPO** optimizes for cumulative damage by considering total sum of costs, thereby re-entering the feasible set.

4.5 Discussion and Conclusion

In summary, we proposed a new optimization formulation and a class of algorithms for safety-constrained reinforcement learning. Our framework optimizes reward performance for states in least-violation policy's feasible state space while maintaining persistent safety as well as providing the safest behavior in other states by ensuring entrance into the feasible set with minimal cumulative discounted costs. Using our proposed reachability estimation function, we prove our algorithm's class of actor-critic methods converge a locally optimal policy for our proposed optimization. We provide extensive experimental results on a diverse suite of

environments in Safety Gym, PyBullet, and MuJoCo, and an environment with multiple hard and soft constraints, to demonstrate the effectiveness of our algorithm when compared with several SOTA baselines.

4.6 Acknowledgement

Chapter 4, in full, is a reprint of the material as it appears in "Iterative Reachability Estimation for Safe Reinforcement Learning," M. Ganai, Z. Gong, C. Yu, S. Herbert, S. Gao. in Advances in Neural Information Processing Systems, 2023. The thesis author is the primary investigator and author of this paper.

Chapter 5

Limitations and Future Works

5.1 Current Limitations

Hamilton-Jacobi reachability estimation has demonstrated great performance in a variety of problem formulations, even scaling up to vision-based data while providing some forms of safety guarantees. Nonetheless, there are some limitations to these approaches.

Like most learning-based approaches, acquiring the HJ reachability estimation value functions requires obtaining many samples to compute a good estimation. This may be difficult to do when trying to guarantee safety in an online framework where the number of attempts is limited. Furthermore, while recent works can guarantee convergence to the optimally safe control and value function as shown in [43, 99], learning-based methods have issues including catastrophic forgetting [79] that make it difficult to guarantee safety within a limited number of training steps/samples.

The valid definition and formulation of the HJ reachability estimation may also be limited in the possible behaviors that it can capture. For instance, when learning the reachability formulation, [4, 42] had to define it in a discounted Bellman formulation. One way this was done was by defining a different optimal control problem as in (3.7) that incorporated discounted costs. However, the exact Bellman formulation (shown in (3.9)) to solve this had a loose gamma contraction mapping, thereby taking longer to converge to the value function solution. The other, most frequently used approach from (3.10) was defining a different Bellman formulation

which had a tighter gamma contraction mapping – while this is a good approximation of the true Bellman formulation solution, it is not an exact reachability value function solution. Furthermore, in either case, the optimal control was redefined with discounting so the optimal control may potentially be in conflict with the true undiscounted optimal control.

Another limitation is that the reachability value functions, especially those learned via the Bellman formulation, are rigorously defined only for deterministic dynamics or non-deterministic dynamics with known bounds [4]. Methods like those found in [54, 100] that consider stochastic noise/disturbance require learning an additional model or disturbance policy. Probabilistic reachability approaches meant for stochastic environments such as [1, 26, 27, 43, 80, 89] can only use HJ reachability when the cost function is redefined in a binary manner. Other stochastic reachability approaches require direct access to some form of a dynamics or control model like a probabilistic density function of the adversary's predicted control [94].

Also, as explored in [43], when the agent is outside the feasible set, the reachability value function does not guarantee reentrance back into the feasible set. In particular, the control may incur a potentially infinite number of costs smaller than the maximum cost along the trajectory. This can be addressed by creating a new cost function.

Finally, learning HJ reachability in a model-free manner is limited by assumptions of the online learning of the Bellman formulation. In particular, there exist novel HJ Bellman variational inequalities such as the Control Barrier Value Function variational inequality (CBFVI) [28] whose solutions are provably both a HJ reachability value function *and* a Control Barrier Function. The discrete-time solution of the CBFVI is similar to that found in (3.9) but requires $\gamma \geq 1$. However, if we want to learn the value function online via Bellman recursion, we need to ensure gamma contraction mapping which requires $\gamma \in [0, 1)$. Because there is no feasible overlap in the solution space for γ, learning a Control Barrier Function with HJ reachability estimation online remains an open challenge.

5.2 Future Works

HJ reachability estimation for learning-based control is a rapidly growing field and has much more to offer. Future work includes addressing concerns about its limitations as well as extending new topics in reinforcement learning and HJ reachability.

One important domain in learned control is single lifetime reinforcement learning [18] or lifelong learning [90] in which the goal is to solve a task without resetting the environment. In the safety version of this setting, the algorithms need to be able to learn controls on the go while not terminating or entering a deadly state. In this scenario, safety is a priority during exploration – thus there remains the open problem of ensuring safety and goal reachability *during* the training process or from data so as to safely complete the task in one trial.

Another topic to explore is HJ reachability estimation in the Koopman-Hopf framework [82]. The Hopf formula for HJ reachability analysis is an approach proposed to solve high-dimensional tasks [29, 34, 61] but is limited to linear time-varying systems. Koopman theory [63, 71] is a mechanism of mapping nonlinear dynamics into some linear dynamics in a very high-dimensional latent space. There has been some work on using Koopman and reachability analysis together [62], but the work of [82] is novel in proposing to combine the Hopf reachability framework and Koopman theory to solve problems up to 10-dimensions. There has been recent work improving the scalability of Koopman-based methods through learning-based mechanisms [65, 87]. This leaves room for future research in further scaling Koopman-Hopf reachability analysis and applying this technique to learning-based control.

5.3 Acknowledgement

Chapter 5 has been submitted for publication of the material in "Hamilton-Jacobi Reachability in Reinforcement Learning: A Survey," M. Ganai; S. Gao; S. Herbert, 2024. The thesis author was the primary investigator and author of this paper.

Bibliography

[1] Alessandro Abate, Maria Prandini, John Lygeros, and Shankar Sastry. Probabilistic reachability and safety for controlled discrete time stochastic hybrid systems. *Automatica*, 44(11):2724–2734, 2008.

[2] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pages 22–31. PMLR, 2017.

[3] Anayo K Akametalu, Jaime F Fisac, Jeremy H Gillula, Shahab Kaynama, Melanie N Zeilinger, and Claire J Tomlin. Reachability-based safe learning with gaussian processes. In *53rd IEEE Conference on Decision and Control*, pages 1424–1431. IEEE, 2014.

[4] Anayo K. Akametalu, Shromona Ghosh, Jaime F. Fisac, Vicenc Rubies-Royo, and Claire J. Tomlin. A minimum discounted reward hamilton–jacobi formulation for computing reachable sets. *IEEE Transactions on Automatic Control*, 69(2):1097–1103, 2024.

[5] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[6] Matthias Althoff, Goran Frehse, and Antoine Girard. Set propagation techniques for reachability analysis. *Annual Review of Control, Robotics, and Autonomous Systems*, 4:369–395, 2021.

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

[8] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin. Hamilton-Jacobi reachability: A brief overview and recent advances. In *Conf. on Decision and Control*, 2017.

[9] EN Barron. Differential games with maximum cost. *NONLINEAR ANAL. THEORY METHODS APPLIC.*, 14(11):971–989, 1990.

[10] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[11] Dimitri P Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48(3):334–334, 1997.

[12] Olivier Bokanowski and Hasnaa Zidani. Minimal time problems with moving targets and obstacles. *IFAC Proceedings Volumes*, 44(1):2589–2593, 2011.

[13] Vivek S Borkar. *Stochastic approximation: a dynamical systems viewpoint*, volume 48. Springer, 2009.

[14] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[15] Lukas Brunke, Melissa Greeff, Adam W Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5:411–444, 2022.

[16] Ya-Chien Chang and Sicun Gao. Stabilizing neural control using self-learned almost lyapunov critics. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1803–1809, 2021.

[17] Ya-Chien Chang, Nima Roohi, and Sicun Gao. Neural lyapunov control. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[18] Annie Chen, Archit Sharma, Sergey Levine, and Chelsea Finn. You only live once: Single-life reinforcement learning. *Advances in Neural Information Processing Systems*, 35:14784–14797, 2022.

[19] Bingqing Chen, Jonathan Francis, Jean Oh, Eric Nyberg, and Sylvia L Herbert. Safe autonomous racing via approximate reachability on ego-vision. *arXiv preprint arXiv:2110.07699*, 2021.

[20] Mo Chen, Somil Bansal, Jaime F Fisac, and Claire J Tomlin. Robust sequential trajectory planning under disturbances and adversarial intruder. *IEEE Transactions on Control Systems Technology*, 27(4):1566–1582, 2018.

[21] Mo Chen, Sylvia Herbert, and Claire J Tomlin. Fast reachable set approximations via state decoupling disturbances. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 191–196. IEEE, 2016.

[22] Mo Chen, Sylvia L Herbert, Mahesh S Vashishtha, Somil Bansal, and Claire J Tomlin. Decomposition of reachable sets and tubes for a class of nonlinear systems. *Trans. on Automatic Control*, 2018.

[23] Mo Chen, Qie Hu, Jaime F Fisac, Kene Akametalu, Casey Mackin, and Claire J Tomlin. Reachability-based safety and goal satisfaction of unmanned aerial platoons on air highways. *Journal of Guidance, Control, and Dynamics*, 40(6):1360–1373, 2017.

[24] Mo Chen and Claire J Tomlin. Hamilton–jacobi reachability: Some recent theoretical advances and applications in unmanned airspace management. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:333–358, 2018.

[25] 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 *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3387–3395, 2019.

[26] Hao-Tien Chiang, Nick Malone, Kendra Lesser, Meeko Oishi, and Lydia Tapia. Aggressive moving obstacle avoidance using a stochastic reachable set based potential field. In *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*, pages 73–89. Springer, 2015.

[27] Hao-Tien Chiang, Nick Malone, Kendra Lesser, Meeko Oishi, and Lydia Tapia. Path-guided artificial potential fields with stochastic reachable sets for motion planning in highly dynamic environments. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 2347–2354. IEEE, 2015.

[28] Jason J Choi, Donggun Lee, Koushil Sreenath, Claire J Tomlin, and Sylvia L Herbert. Robust control barrier–value functions for safety-critical control. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 6814–6821. IEEE, 2021.

[29] Yat Tin Chow, Jérôme Darbon, Stanley Osher, and Wotao Yin. Algorithm for overcoming the curse of dimensionality for state-dependent hamilton-jacobi equations. *Journal of Computational Physics*, 387:376–409, 2019.

[30] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.

[31] 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.

[32] Earl A Coddington, Norman Levinson, and T Teichmann. Theory of ordinary differential equations, 1956.

[33] Samuel Coogan and Murat Arcak. Efficient finite abstraction of mixed monotone systems. In *Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control*, pages 58–67, 2015.

[34] Jérôme Darbon and Stanley Osher. Algorithms for overcoming the curse of dimensionality for certain hamilton–jacobi equations arising in control theory and elsewhere. *Research in the Mathematical Sciences*, 3(1):19, 2016.

[35] Charles Dawson, Sicun Gao, and Chuchu Fan. Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control. *IEEE Transactions on Robotics*, 2023.

[36] Eric V Denardo. Contraction mappings in the theory underlying dynamic programming. *Siam Review*, 9(2):165–177, 1967.

[37] Jingliang Duan, Zhengyu Liu, Shengbo Eben Li, Qi Sun, Zhenzhong Jia, and Bo Cheng. Adaptive dynamic programming for nonaffine nonlinear optimal control problem with state constraints. *Neurocomputing*, 484:128–141, 2022.

[38] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *International Conference on Learning Representations*, 2019.

[39] Alec Farid, Sushant Veer, and Anirudha Majumdar. Task-driven out-of-distribution detection with statistical guarantees for robot learning. In *Conference on Robot Learning*, pages 970–980. PMLR, 2022.

[40] 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.

[41] Jaime F Fisac, Mo Chen, Claire J Tomlin, and S Shankar Sastry. Reach-avoid problems with time-varying dynamics, targets and constraints. In *Hybrid Systems: Computation and Control*. ACM, 2015.

[42] Jaime F Fisac, Neil F Lugovoy, Vicenç Rubies-Royo, Shromona Ghosh, and Claire J Tomlin. Bridging hamilton-jacobi safety analysis and reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8550–8556. IEEE, 2019.

[43] Milan Ganai, Zheng Gong, Chenning Yu, Sylvia L Herbert, and Sicun Gao. Iterative reachability estimation for safe reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[44] Milan Ganai, Chiaki Hirayama, Ya-Chien Chang, and Sicun Gao. Learning stabilization control from observations by learning lyapunov-like proxy models. *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2913–2920, 2023.

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

[46] Zheng Gong, Muhan Zhao, Thomas Bewley, and Sylvia Herbert. Constructing control lyapunov-value functions using hamilton-jacobi reachability analysis. *IEEE Control Systems Letters*, 7:925–930, 2022.

[47] Sven Gronauer. Bullet-safety-gym: A framework for constrained reinforcement learning. 2022.

[48] Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, and Alois Knoll. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*, 2022.

[49] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[50] Michael R Hafner and Domitilla Del Vecchio. Computation of safety control for uncertain piecewise continuous systems on a partial order. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 1671–1677. IEEE, 2009.

[51] Peter Heidlauf, Alexander Collins, Michael Bolender, and Stanley Bak. Verification challenges in f-16 ground collision avoidance and other automated maneuvers. In *ARCH@ ADHS*, pages 208–217, 2018.

[52] James Herman, Jonathan Francis, Siddha Ganju, Bingqing Chen, Anirudh Koul, Abhinav Gupta, Alexey Skabelkin, Ivan Zhukov, Max Kumskoy, and Eric Nyberg. Learn-to-race: A multimodal control environment for autonomous racing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9793–9802, 2021.

[53] Kai-Chieh Hsu, Haimin Hu, and Jaime Fernández Fisac. The safety filter: A unified view of safety-critical control in autonomous systems. *arXiv preprint arXiv:2309.05837*, 2023.

[54] Kai-Chieh Hsu, Duy Phuong Nguyen, and Jaime Fernàndez Fisac. Isaacs: Iterative soft adversarial actor-critic for safety. In Nikolai Matni, Manfred Morari, and George J. Pappas, editors, *Proceedings of the 5th Annual Learning for Dynamics and Control Conference*, volume 211 of *Proceedings of Machine Learning Research*. PMLR, 15–16 Jun 2023.

[55] Kai-Chieh Hsu, Allen Z. Ren, Duy P. Nguyen, Anirudha Majumdar, and Jaime F. Fisac. Sim-to-lab-to-real: Safe reinforcement learning with shielding and generalization guarantees. *Artificial Intelligence*, page 103811, 2022.

[56] Kai-Chieh Hsu, Vicenç Rubies-Royo, Claire J. Tomlin, and Jaime F. Fisac. Safety and liveness guarantees through reach-avoid reinforcement learning. In *Proceedings of Robotics: Science and Systems*, Virtual, 7 2021.

[57] Boris Ivanovic, James Harrison, Apoorva Sharma, Mo Chen, and Marco Pavone. Barc: Backward reachability curriculum for robotic reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 15–21. IEEE, 2019.

[58] Shahab Kaynama and Meeko Oishi. Schur-based decomposition for reachability analysis of linear time-invariant systems. In *Proceedings of the 48h IEEE Conference on Decision*

and Control (CDC) held jointly with 2009 28th Chinese Control Conference, pages 69–74. IEEE, 2009.

[59] Shahab Kaynama and Meeko Oishi. A modified riccati transformation for decentralized computation of the viability kernel under lti dynamics. *IEEE Transactions on Automatic Control*, 58(11):2878–2892, 2013.

[60] H.K. Khalil. *Nonlinear Systems*. Pearson Education. Prentice Hall, 2002.

[61] Matthew R Kirchner, Robert Mar, Gary Hewer, Jérôme Darbon, Stanley Osher, and Yat Tin Chow. Time-optimal collaborative guidance using the generalized hopf formula. *IEEE Control Systems Letters*, 2(2):201–206, 2017.

[62] Niklas Kochdumper and Stanley Bak. Conformant synthesis for koopman operator linearized control systems. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 7327–7332. IEEE, 2022.

[63] Bernard O Koopman. Hamiltonian systems and transformation in hilbert space. *Proceedings of the National Academy of Sciences*, 17(5):315–318, 1931.

[64] Karen Leung, Edward Schmerling, Mengxuan Zhang, Mo Chen, John Talbot, J Christian Gerdes, and Marco Pavone. On infusing reachability-based safety assurance within planning frameworks for human–robot vehicle interactions. *The International Journal of Robotics Research*, 39(10-11):1326–1345, 2020.

[65] Bethany Lusch, J Nathan Kutz, and Steven L Brunton. Deep learning for universal linear embeddings of nonlinear dynamics. *Nature communications*, 9(1):4950, 2018.

[66] Haitong Ma, Jianyu Chen, Shengbo Eben, Ziyu Lin, Yang Guan, Yangang Ren, and Sifa Zheng. Model-based constrained reinforcement learning using generalized control barrier function. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4552–4559. IEEE, 2021.

[67] Haitong Ma, Yang Guan, Shegnbo Eben Li, Xiangteng Zhang, Sifa Zheng, and Jianyu Chen. Feasible actor-critic: Constrained reinforcement learning for ensuring statewise safety. *arXiv preprint arXiv:2105.10682*, 2021.

[68] Anirudha Majumdar, Alec Farid, and Anoopkumar Sonar. Pac-bayes control: learning policies that provably generalize to novel environments. *The International Journal of Robotics Research*, 40(2-3):574–593, 2021.

[69] Kostas Margellos and John Lygeros. Hamilton–jacobi formulation for reach–avoid differential games. *IEEE Transactions on automatic control*, 56(8):1849–1861, 2011.

[70] David Q Mayne, James B Rawlings, Christopher V Rao, and Pierre OM Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.

[71] Igor Mezić. Koopman operator, geometry, and learning of dynamical systems. *Not. Am. Math. Soc.*, 68(7):1087–1105, 2021.

[72] Ian M Mitchell. Scalable calculation of reach sets and tubes for nonlinear systems with terminal integrators: a mixed implicit explicit formulation. In *Proceedings of the 14th international conference on Hybrid systems: computation and control*, pages 103–112, 2011.

[73] Ian M Mitchell, Alexandre M Bayen, and Claire J Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on automatic control*, 50(7):947–957, 2005.

[74] Ian M Mitchell and Claire J Tomlin. Overapproximating reachable sets by hamilton-jacobi projections. *journal of Scientific Computing*, 19:323–346, 2003.

[75] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[76] Zhizhen Qin, Tsui-Wei Weng, and Sicun Gao. Quantifying safety of learning-based self-driving control using almost-barrier functions. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12903–12910. IEEE, 2022.

[77] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 7(1):2, 2019.

[78] Allen Ren, Sushant Veer, and Anirudha Majumdar. Generalization guarantees for imitation learning. In *Conference on Robot Learning*, pages 1426–1442. PMLR, 2021.

[79] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

[80] Hossein Sartipizadeh, Abraham P Vinod, Behçet Açikmeşe, and Meeko Oishi. Voronoi partition-based scenario reduction for fast sampling-based stochastic reachability computation of linear systems. In *2019 American Control Conference (ACC)*, pages 37–44. IEEE, 2019.

[81] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

[82] Will Sharpless, Nikhil Shinde, Matthew Kim, Yat Tin Chow, and Sylvia Herbert. Koopman-hopf hamilton-jacobi reachability and control. *arXiv preprint arXiv:2303.11590*, 2023.

[83] Oswin So and Chuchu Fan. Solving stabilize-avoid optimal control via epigraph form and deep reinforcement learning. In *Proceedings of Robotics: Science and Systems*, 2023.

[84] Sean Summers and John Lygeros. Verification of discrete time stochastic hybrid systems: A stochastic reach-avoid decision problem. *Automatica*, 46(12):1951–1961, 2010.

[85] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3:9–44, 1988.

[86] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA, 2018.

[87] Naoya Takeishi, Yoshinobu Kawahara, and Takehisa Yairi. Learning koopman invariant subspaces for dynamic mode decomposition. *Advances in neural information processing systems*, 30, 2017.

[88] Chen Tessler, Daniel Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2019.

[89] Adam J Thorpe, Vignesh Sivaramakrishnan, and Meeko MK Oishi. Approximate stochastic reachability for high dimensional systems. In *2021 American Control Conference (ACC)*, pages 1287–1293. IEEE, 2021.

[90] Sebastian Thrun. A lifelong learning perspective for mobile robot control. In *Intelligent robots and systems*, pages 201–214. Elsevier, 1995.

[91] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

[92] John N Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine learning*, 16:185–202, 1994.

[93] Sushant Veer and Anirudha Majumdar. Probably approximately correct vision-based planning using motion primitives. In *Conference on Robot Learning*, pages 1001–1014. PMLR, 2021.

[94] Abraham P Vinod, Baisravan HomChaudhuri, Christoph Hintz, Anup Parikh, Stephen P Buerger, Meeko MK Oishi, Greg Brunson, Shakeeb Ahmad, and Rafael Fierro. Multiple pursuer-based intercept via forward stochastic reachability. In *2018 Annual American Control Conference (ACC)*, pages 1559–1566. IEEE, 2018.

[95] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

[96] Tengyu Xu, Yingbin Liang, and Guanghui Lan. Crpo: A new approach for safe reinforcement learning with convergence guarantee. In *International Conference on Machine Learning*, pages 11480–11491. PMLR, 2021.

[97] Long Yang, Jiaming Ji, Juntao Dai, Yu Zhang, Pengfei Li, and Gang Pan. Cup: A conservative update policy algorithm for safe reinforcement learning. *arXiv preprint arXiv:2202.07565*, 2022.

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

[99] Dongjie Yu, Haitong Ma, Shengbo Li, and Jianyu Chen. Reachability constrained reinforcement learning. In *International Conference on Machine Learning*, pages 25636–25655. PMLR, 2022.

[100] Dongjie Yu, Wenjun Zou, Yujie Yang, Haitong Ma, Shengbo Eben Li, Jingliang Duan, and Jianyu Chen. Safe model-based reinforcement learning with an uncertainty-aware reachability certificate. *arXiv preprint arXiv:2210.07553*, 2022.

[101] Linrui Zhang, Li Shen, Long Yang, Shi-Yong Chen, Bo Yuan, Xueqian Wang, and Dacheng Tao. Penalized proximal policy optimization for safe reinforcement learning. In *International Joint Conference on Artificial Intelligence*, 2022.

[102] Yiming Zhang, Quan Vuong, and Keith Ross. First order constrained optimization in policy space. *Advances in Neural Information Processing Systems*, 33:15338–15349, 2020.

[103] Weiye Zhao, Tairan He, Rui Chen, Tianhao Wei, and Changliu Liu. State-wise safe reinforcement learning: A survey. *arXiv preprint arXiv:2302.03122*, 2023.