Handling Long-Term Constraints And Uncertainty in Safe Reinforcement Learning

Umgang mit langfristigen Beschränkungen und Ungewissheit beim sicheren Reinforcement Learning

Master thesis by Jonas Günster

Date of submission: July 30, 2024

1. Review: M.Sc. Puze Liu
2. Review: Ph.D. Davide Tateo
3. Review: Prof. Ph.D. Jan Peters

Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

Erklärung zur Abschlussarbeit gemäß § 22 Abs. 7 APB TU Darmstadt

Hiermit erkläre ich, Jonas Günster, dass ich die vorliegende Arbeit gemäß § 22 Abs. 7 APB der TU Darmstadt selbstständig, ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Ich habe mit Ausnahme der zitierten Literatur und anderer in der Arbeit genannter Quellen keine fremden Hilfsmittel benutzt. Die von mir bei der Anfertigung dieser wissenschaftlichen Arbeit wörtlich oder inhaltlich benutzte Literatur und alle anderen Quellen habe ich im Text deutlich gekennzeichnet und gesondert aufgeführt. Dies gilt auch für Quellen oder Hilfsmittel aus dem Internet.

Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 30. Juli 2024

Jonas Günster

Abstract

Robots as well as other autonomous systems are becoming increasingly prevalent in our daily lives. To control these systems in complex and uncertain environments, Reinforcement Learning is a promising approach. However, Reinforcement Learning lacks safety guarantees, which are crucial for deployment in the real world. Especially in autonomous control or robot-human collaboration, strict constraint adherence is paramount. Safe Reinforcement Learning methods aim to provide these safety guarantees with two distinct approaches. The model-based approach exploits additional knowledge such as constraints and dynamics to ensure safety. While the dynamics are readily available in the robotics domain, assuming knowledge of the constraints is problematic because they can be difficult to compute analytically. Additionally encoding long-term safety constraints apriori is difficult in underactuated or stochastic environments, where avoiding actions have to be taken long before a given constraint is reached. Model-free approaches, on the other hand, do not require this additional knowledge. Instead, they learn safe behaviour through countless unsafe environment interactions, which are only feasible in simulation. Thus, the deployment of model-free approaches to the real world is difficult due to the sim-to-real gap. In this thesis, we bridge the gap between model-based and model-free approaches with a model-based safe reinforcement learning algorithm that can enforce long-term safety constraints as well as capture and handle uncertainty. Our approach is based on the ATACOM algorithm, which uses the robot dynamics and predefined constraints to project actions onto a safe manifold. We incorporate online constraint learning into the ATACOM approach to handle long-term safety and uncertainty. We explicitly model the constraints' uncertainty in a distributional reinforcement learning perspective, which results in a risk-aware policy. We evaluate our approach in three distinct environments: a cart-pole control task, a differential drive robot navigation task, and a robot air hockey task. In these environments, our approach is competitive or superior to state-of-the-art methods in final performance. Additionally, we maintain safer behaviors throughout training and converge to a safer policy.

Zusammenfassung

Roboter und andere autonome Systeme sind in unserem Alltag immer häufiger anzutreffen. Um diese Systeme in komplexen und unsicheren Umgebungen zu steuern, ist Reinforcement Learning ein vielversprechender Ansatz. Allerdings fehlt es dem Reinforcement Learning an Sicherheitsgarantien, die für den Einsatz in der realen Welt entscheidend sind. Vor allem bei der autonomen Steuerung oder der Zusammenarbeit zwischen Roboter und Mensch ist die strikte Einhaltung von Beschränkungen von größter Notwendigkeit. Sichere Reinforcement Learning-Methoden versuchen, diese Sicherheitsgarantien mit zwei unterschiedlichen Ansätzen zu gewährleisten. Der modellbasierte Ansatz nutzt zusätzliches Wissen wie Beschränkungen und Dynamik, um Sicherheit zu gewährleisten. Während die Dynamik in der Robotik bereits verfügbar ist, ist es problematisch, die Kenntnis der Beschränkungen vorauszusetzen, da diese analytisch schwer zu berechnen sind. Außerdem ist es in unteraktuierten oder stochastischen Umgebungen schwierig, langfristige Sicherheitsbedingungen im Voraus zu kodieren, da Ausweichmanöver lange vor Erreichen einer bestimmten Beschränkung durchgeführt werden müssen. Modellfreie Ansätze hingegen benötigen dieses zusätzliche Wissen nicht. Stattdessen lernen sie sicheres Verhalten durch unzählige unsichere Umgebungsinteraktionen, die nur in der Simulation möglich sind. Daher ist der Einsatz modellfreier Ansätze in der realen Welt aufgrund der Diskrepanz zwischen Simulation und Realität schwierig. In dieser Arbeit überbrücken wir die Lücke zwischen modellbasierten und modellfreien Ansätzen mit einem modellbasierten sicheren Reinforcement-Learning-Algorithmus, der sowohl langfristige Sicherheitsbedingungen durchsetzen als auch Unsicherheiten erfassen und behandeln kann. Unser Ansatz basiert auf dem ATACOM-Algorithmus, der die Roboterdynamik und vordefinierte Beschränkungen verwendet, um Steuersignale auf eine sichere Menge zu projizieren. Wir integrieren Online-Constraint-Learning in den ATACOM-Ansatz, um langfristige Sicherheit und Unsicherheit zu behandeln. Wir modellieren explizit die Ungewissheit der Beschränkungen in einer verteilungsorientierten Reinforcement-Learning-Perspektive, was zu einer risikobewussten Strategie führt. Wir evaluieren unseren Ansatz in drei verschiedenen Umgebungen: eine Aufgabe zur Steuerung von Cart-Poles, eine Aufgabe zur Navigation von Robotern mit Differentialantrieb und eine Aufgabe beim Roboter-Air-Hockey. In diesen Umgebungen ist unser Ansatz konkurrenzfähig oder den modernsten Methoden in der Endleistung überlegen. Außerdem behalten wir während des gesamten Trainings sicherere Verhaltensweisen bei und konvergieren zu einer sichereren Strategie.

Contents

Figures and Tables

List of Figures

List of Tables

1. Introduction

1.1. Motivation

In recent years Reinforcement Learning (RL) has shown remarkable success in solving complex tasks. For example Mnih et al. [33] proposed an agent that performs on par with professional video game testers in a suite of 51 Atari games. However, these games are mostly straightforward problems that don't require sophisticated strategies. In Games with a greater strategic depth, such as Dota 2 [7], Starcraft II [46], and Go [38], RL agents have surpassed even the best human performance and compete in a league of their own. In more practical applications, RL has been deployed in robotic manipulation [22] for grasping tasks such as opening doors or folding laundry. Another field in which RL agents thrive is autonomous driving [8].

In summary RL shows a lot of promise in automating tedious and dangerous tasks like driving or folding laundry for humans. However, a major challenge for RL is adhering to safety constraints. The RL framework does not provide any safety guarantees, which are crucial for deployment in the real world. In domains like autonomous driving or robot-human collaboration, constraint violations can lead to catastrophic failures and harm people or cause damage. Unfortunately, simply encoding these constraints into the reward function is not sufficient, as the agent will maximize the sum of rewards. Thus short-term constraint violations are worth it for the agent if it leads to high rewards in the long term. Instead, the constraint needs to be respected at all times regardless of the potential reward.

The field of Safe Reinforcement Learning (SafeRL) solves this problem by learning policies that maximize task performance while satisfying the safety requirements. The SafeRL problem usually falls into one of the two categories: Safe Exploration (SafeExp) and Constrained Markov Decision Processes (CMDP).

SafeExp aims to ensure the agent's safety during the exploration phase and formulates the safety problem as a stepwise constraint that the agent should not violate at each step. Solving the SafeExp problem requires additional prior knowledge, such as constraints, robot dynamics, or previously collected datasets. While the SafeExp algorithms have been deployed successfully on complex real-world tasks [44, 39, 29, 30], most of these approaches suffer from many drawbacks preventing their application to complex or out-of-the-lab tasks. First, safety specifications defined as constraints entail an in-depth understanding of the environment and dynamics. Designing and validating safety constraints requires extensive expertise and experience. Second, real-world applications contain various uncertainty sources, such as sensor noise, model error, environmental disturbance, and partial observability, which are often neglected in the design of constraints. Third, the optimization techniques for constrained optimization problems are limited and often require a specific problem structure, such as quadratic programming with linear complementarity constraints. Lastly, the learning algorithms should also guarantee long-term safety, which is often neglected [16] or simplified to a tractable setting [44, 20].

CMDP approaches, on the other hand, learn the safe policies directly from the interaction with the environment. Thus, they cannot guarantee safety during the exploration phase but converge on a safe policy. This approach does not need handcrafted constraints and can handle long-term safety. However, the sim-to-real transfer can be problematic for safety, as the reality gap may cause catastrophic constraint violations.

1.2. Goal of the Thesis

In this thesis, we aim to bridge the gap between SafeExp methods and model-free SafeRL approaches. We argue that incorporating prior knowledge for safety-critical robotics applications can be beneficial, as prior knowledge, such as kinematics and dynamics, is often well-studied and readily available. Furthermore, in robotics, we can assume that the dynamics of the robot itself are known, while knowledge of the environment is unknown. Under this assumption, we combine techniques from model-free SafeRL methods and SafeExp approaches, showing how to exploit the knowledge of the dynamics while learning of unknown, long-term constraints.

To achieve this goal, we extend the Acting on the TAngent Space of the COnstraint Manifold (ATACOM) approach by dropping some key assumptions of the original formulation, such as known constraints and local safety, allowing learning of long-term safety constraints directly from data. Furthermore, we explicitly model the constraints' uncertainty in a

distributional RL perspective, which provides us with a way to estimate the total uncertainty of the model.

We provide a novel algorithm, Distributional ATACOM (D-ATACOM), that combines the ATACOM framework with distributional learned constraints, ensuring long-term safety and properly dealing with constraint uncertainty. D-ATACOM produces a risk-aware policy by restricting the level of acceptable risk, allowing the agent to explore more cautiously whenever the constraints are uncertain or the policy is close to violating them.

Our experiments demonstrate that D-ATACOM achieves a safer performance during the training phase and reaches a similar or better performance at the end of training. For tasks where the optimal policy stays within constraints, D-ATACOM explores more cautiously at a cost of slower learning speed. Instead, D-ATACOM show faster and safer behaviors during training whenever there is a conflict between the policy optimization problem and the constraint satisfaction one. Additionally, our method does not require excessive parameter tuning, as it includes automatic tuning rules for the most important hyperparameters.

2. Foundations

In this chapter, we will introduce the concepts of Reinforcement Learning, Safe Reinforcement Learning and Distributional Reinforcement Learning. Additionally, we present the method of ATACOM and discuss the related work in the field of Safe Reinforcement Learning.

2.1. Reinforcement Learning

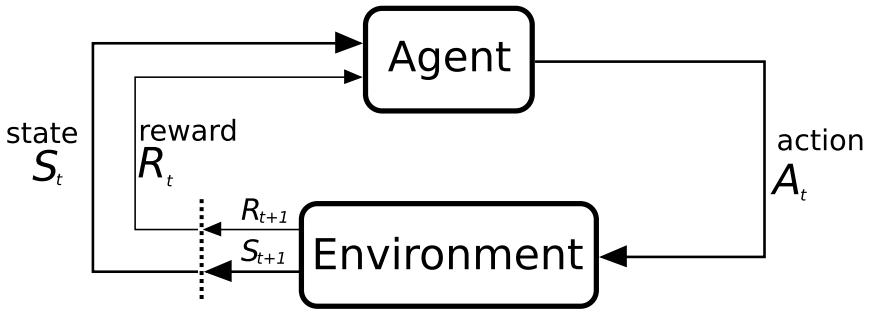


Figure 2.1.: Diagram of the agent environment interaction. The agent interacts with the environment based on the state s with an action a

Reinforcement learning is an area of machine learning, where an agent learns to interact with an environment by exploring it. Instead of the labeled data utilized in supervised learning, reinforcement learning collects data via an agent. As shown in Figure 2.1 the

agent observes that the environment is currently in state s_t. Based on this observation he chooses an action a_t. After taking the action in the state s_t, the agent reach a new state s_{t+1}. Additionally, there is a reward function, which provides a local measure of the agent's performance. The resulting reward can be seen as a rating for the action a_t given the state s_t.

To formalize this concept mathematically, we assume that the environment can be modeled as a Markov Decision Process (MDP) [41], which is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. The state space $\mathcal{S}$ contains all possible states the environment can be in. The action space $\mathcal{A}$ contains all possible actions the agent can take. The transition function $\mathcal{P}$ describes the probability of transitioning from one state to another given an action. The reward function $\mathcal{R}$ provides the reward for transitioning from one state to another. The discount factor γ determines how much the agent values future rewards.

The goal of the agent is to maximize the discounted sum of rewards. This is achieved by learning a policy π, which maps states to actions. The agent learns the policy by interacting with the environment and updating the policy based on the rewards received. However, this formulation lacks the concept of safety, which is crucial for real-world applications. As explained in Section 1.1, encoding constraints into the reward function is not sufficient either. Instead, we introduce the concept of Safe Reinforcement Learning to address this issue.

2.2. Safe Reinforcement Learning

We formulate the safety problem in the framework of CMDP [2, 3]. A CMDP is defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, k, \gamma)$ with a state space $\mathcal{S}$, an action space $\mathcal{A}$, a stochastic function $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ that represents the transition probability from a state to another state by an action, a reward function $r(s, a) \in [r_{\min}, r_{\max}]$, a constraint function $k(s) \in [k_{\min}, k_{\max}]$, and a discount factor $\gamma \in [0, 1)$.

The goal of SafeRL algorithms is to produce a policy π that maximizes the expected return while adhering to a safety constraint $\mathcal{F}(s)$:

$$\max_{\pi} \quad \mathop{\mathbb{E}}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right], \qquad \text{s.t.} \quad \mathcal{F}(s) \leqslant 0 \tag{2.1}$$

Depending on the perspective of the safety problem [10] $\mathcal{F}$ can take different forms, such as

$$k(s_t) \leqslant 0, \forall t \tag{2.2a}$$

$$\mathbb{P}(k(s_t) > 0) \leqslant \eta_c, \forall t \tag{2.2b}$$

$$\mathop{\mathbb{E}}_{\pi}\left[\sum_t \gamma^t k(s_t)|s_t\right] \leqslant \eta_e \tag{2.2c}$$

The first constraint in (2.2a) describes the hard constraint, to be satisfied at each time step. However, we cannot enforce these constraints in the setting of stochastic environments. To address this issue, we can use chance constraints, as shown in (2.2b), to restrict the probability of the violations to be smaller than a threshold η_c. Both (2.2a) and (2.2b) are stepwise constraints that focus on safety at the current time step. The last type of constraint, as shown in (2.2c), forces the cumulative cost of the trajectory to be smaller than a threshold η_e.

In this thesis, we define safety and feasibility following [50] as:

Definition 1. *Consider a constraint function $k : \mathcal{S} \to \mathbb{R}$ and a policy $\pi : \mathcal{S} \to \mathcal{A}$.*

i. A state s is Safe *if $k(s) \leqslant 0$.*
ii. The Safe Set *is defined as $\mathcal{S}_S = \{s \in \mathcal{S} : k(s) \leqslant 0\}$.*
iii. The Unsafe Set *is the complementary set $\bar{\mathcal{S}}_S = \mathcal{S} \backslash \mathcal{S}_S$.*
iv. A state s is Feasible *under a policy π if $k(s_t) \leqslant 0$ for all $t \in \{0, 1, \ldots, \infty\}, s_0 = s, a_t = \pi(s_t)$.*
v. A policy π is Feasible *if all initial states are feasible under π.*

To ensure that the policy is *Feasible* we use the constraint formulation of form (2.2c). To account for prediction errors and uncertainties, we combine this constraint formulation with the distributional formulation of (2.2b). We will introduce a method to learn this distributional long-term safety constraint in the following section.

2.2.1. Distributional Reinforcement Learning

Unlike the typical RL setting that considers the expected value of the reward (the cumulative cost in our case), distributional RL treats the reward (cost) as a random variable and, therefore, the value function describes the distributions of the random cumulative return

(cumulative cost). Distribution RL has shown superior performance in many benchmarking tasks, such as Atari Games [5, 15, 14] and Mujuco tasks [19] since the distributional value function contains more information beyond the first moment. The value function is represented as a random variable Z^π instead of a scalar of the expected value Q^π, the random variable Bellman equation has a similar form [1] $Z^\pi(s) \overset{\mathcal{D}}{=} R(s) + \gamma Z^\pi(S')$ where the random variable S' follows the distribution under the policy π and the dynamics $\mathcal{P}$. The distribution of the random value function can be represented by different types of models, such as the network with fixed support [5, 15], Gaussian Network [43] and Implicit Quantile Network [14]. This thesis focuses on Gaussian Networks. However, we are not limited to the model type and show further experiments using Implicit Quantile Networks.

2.3. Safe Learning on the Constraint Manifold

We briefly introduce the ATACOM approach [28, 29, 30], which forms the basis of our approach. ATACOM addresses stepwise hard constraint, as defined in Equation (2.2a). Furthermore, ATACOM assumes that the dynamic system of the robot is a given nonlinear affine system, $\dot{\boldsymbol{s}} = f(\boldsymbol{s}) + G(\boldsymbol{s})\boldsymbol{a}$. ATACOM constructs the *Constraint Manifold* by introducing a slack variable $\boldsymbol{\mu}$ as $\mathcal{M} \coloneqq \{(\boldsymbol{s}, \boldsymbol{\mu}) \in \mathcal{D} : c(\boldsymbol{s}, \boldsymbol{\mu}) = \mathbf{0}\}$ with $c(\boldsymbol{s}, \boldsymbol{\mu}) = k(\boldsymbol{s}) + \boldsymbol{\mu}$. We assume that $\boldsymbol{\mu}$ is equipped with a dynamic system $\dot{\boldsymbol{\mu}} = \alpha(\boldsymbol{\mu})\boldsymbol{u}_\mu$ and α is a class $\mathcal{K}$ function[2]. Using the concept of Constraint Manifold, a safe controller can be obtained by setting $\frac{\mathrm{d}}{\mathrm{dt}} c(s, \mu) = 0$. The resulting controller has the following form

$$\begin{bmatrix} \boldsymbol{a} \\ \boldsymbol{u}_\mu \end{bmatrix} = -\boldsymbol{J}_u^\dagger \boldsymbol{\psi} - \lambda \boldsymbol{J}_u^\dagger \boldsymbol{c} + \boldsymbol{B}_u \boldsymbol{u}, \tag{2.3}$$

with $J_u(\boldsymbol{s}, \boldsymbol{\mu}) = \begin{bmatrix} J_G(\boldsymbol{s}) & A(\boldsymbol{\mu}) \end{bmatrix}$, $J_G(\boldsymbol{s}) = J_k(\boldsymbol{s})G(\boldsymbol{s})$, $J_k(\boldsymbol{s}) = \frac{d}{ds}k(\boldsymbol{s})$ and the Constraint Drift $\psi(\boldsymbol{s}) = J_k(\boldsymbol{s})f(\boldsymbol{s})$ induced by the system drift $f(\boldsymbol{s})$. $\boldsymbol{B}_u$ is a set of basis vectors tangent to the manifold. The first and the second terms on the right-hand side are the contraction term that compensates for the drift and retracts the system to the manifold; the last term is the tangential term that drives the system along the constraint manifold. An RL agent only needs to learn a policy for the tangential action $\boldsymbol{u} \sim \pi(s)$, while the

[1] $A \overset{\mathcal{D}}{=} B$ denotes that two random variable A and B are equal in distributions.
[2] class $\mathcal{K}$ function: (1) continuous; (2) strictly increasing; (3) $\alpha(0) = 0$.

safety is guaranteed by the controller structure. ATACOM can be used in combination with various types of RL methods, as this technique only modifies the action space of the agent. However, ATACOM requires predefined deterministic constraints. In this work, we extend ATACOM for distributional constraints and propose a new method to simultaneously learn the policy and the long-term constraint online.

2.4. Related Work

Many algorithms try to solve the constraint optimization problem of the CMDP. The RL agent aims to maximize the expected return while maintaining the expected cost below a threshold [1, 11, 45, 32, 40, 18, 4, 13, 52]. This type of constraint has been extended to different variants, such as the risk-sensitive constraint [9, 51, 26, 49], in which the agent aims to keep the worst-case cost, which is computed from the probabilistic constraints with a risk function like Conditional Value-at-Risk (CVaR) or Mean-deviation, below a threshold. Another approach is to directly keep the probability of the constraint violation below a threshold [47, 34, 35].

Different types of constrained optimization techniques are applied in the policy update process, such as the trust-region method [1, 26]. In trust-region methods, the policy update is constrained by the Kullback-Leibler divergence between the old and the new policy. To ensure safety another constraint is added to the policy update such that only feasible Policies are considered. Another method is the interior point method [32], which uses a barrier function to constraint the policy update. The barrier function is a function that approaches infinity as the constraint is violated. The Lagrangian relaxation method [2, 45, 40, 18, 9, 51] is another method to solve the constrained optimization problem. The Lagrangian relaxation method adds a Lagrangian multiplier to the policy update to penalize the constraint violation. The Lagrangian multiplier is updated in each iteration to ensure that the constraint is satisfied. Furthermore, the Lyapunov function is also used to derive a policy improvement procedure [11, 12, 37]. A Lyapunov function is a type of scalar potential function that keeps track of the energy that a system continually dissipates. Besides modeling physical energy, Lyapunov functions can also represent abstract quantities, such as the steady-state performance of a Markov process. The Lyapunov function is used to translate the global discounted sum of cost to a local step-wise surrogate constraint, which is then used to ensure safety in the policy update.

Notably, learning the value function of the accumulative cost has gained incremental attraction as it addresses long-term safety. Recent works have shared a common view

that the value function of constraint provides a predictive estimation of safety, such as the feasibility value [53, 50], control barrier function [42, 31], and safety critic [48, 49].

In this thesis, we leverage the idea of the safety value function. Instead of penalizing the policy by adding a Lagrangian multiplier, we combine the safety value function with a model-based exploration method to derive a safe policy.

3. Methodology

In this chapter, we present D-ATACOM, a data-driven approach that guarantees long-term safety. Our method reduces constraint violations during learning as quickly as possible while converging to a final policy that guarantees long-term safety. To introduce this algorithm, we first establish a method to estimate the constraint for long-term safety, and then we show how to integrate the time-varying constraint into the ATACOM learning framework.

3.1. Feasibility Value Function

To ensure long-term constraint satisfaction, we introduce the concept of *FVF*, which describes the expected cumulative constraint violation under a policy π with an infinity horizon. Formally, we define the feasibility value function under a policy π as

$$V_{\mathrm{F}}^{\pi}(s) = \mathop{\mathbb{E}}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \max(k(s_t), 0) \,\middle|\, s_0 = s\right] \tag{3.1}$$

We assume the constraint $k(s) \in [k_{\min}, k_{\max}]$ is bounded and consequently $V_{\mathrm{F}}^{\pi} \in [0, \frac{k_{\max}}{1-\gamma}]$. When k is an indicator function of the constraint violation, the FVF is analogous to the Constraint Decay Function (CoDF) introduced in [50], defined as $F^{\pi}(s) = \gamma^{N_{\pi}(s)}$, with $N_{\pi}(s)$ the number of steps to the first constraint violation. Unlike the CoDF that assumes the unsafe state is absorbing, the FVF do not terminate the episode at the unsafe state, allowing the agent to retract the state back to safe in the future and therefore $V_{\mathrm{F}}^{\pi}(s) \geqslant F^{\pi}(s)$. The feasibility value function estimates the expected discounted cumulative cost of $\max(k(s), 0)$ under policy π. We can use the standard Bellman operator $(\mathcal{B}^{\pi} V_{\mathrm{F}})(s) = \max(k(s), 0) + \gamma V_{\mathrm{F}}(s)$ to update the estimate. For continuous state-action space, a common choice is to use a neural network to approximate the value function and update the value function using TD learning. When $V_{\mathrm{F}}^{\pi}(s) = 0$, we have $\max(k(s), 0) = 0$ indicating

$k(s) \leqslant 0$. Thus, the *Feasibile Set* $\mathcal{S}_F = \{s \in \mathcal{S} : V_{\mathrm{F}}^{\pi}(s) = 0\}$ is a subset of the Safe Set. Ensuring the feasibility value function to be zero is sufficient to guarantee stepwise safety.

3.2. Stochastic Constraint

The definition of FVF in (3.1) does not capture the stochasticity of the cost. If the constraint has a higher variance or a heavy-tailed distribution, the mean-based constraint may not be sufficient to ensure safety, as shown in Fig. 3.1. This problem can be alleviated by constructing stochastic constraints exploiting the theory of *distributional RL* [6]. We can model the FVF $V_{\mathrm{F}}^{\pi}(s)$ as a distribution value function, i.e., the cumulative cost at each state is a random variable instead of the expectation.

Different parametric models have been used to approximate the target distribution $\max(k(s), 0) + \gamma V_{\mathrm{F}}^{\pi}(s)$. In this thesis, we use two approaches to model the FVF. For the first approach, we model the distribution as a Gaussian. However, this Gaussian assumption is arbitrary and might not properly model the true distribution. Thus we also choose to directly model the Cumulative Distribution Function (CDF) with Implicit Quantile Networks (IQN). This approach does not assume a specific distribution and thus can capture more heavy-tailed cost distributions.

3.2.1. Gaussian Approximation

For the Gaussian Approximation we use Gaussian support up to the 2nd-order moment [43, 48], i.e. we assume $V_{\mathrm{F}}^{\pi}(s) \sim \mathcal{N}\left(\mu^F(s), \Sigma^F(s)\right)$ to approximate the distribution. We can compute the mean of the target distribution $\mu^F(s) = k'(s) + \gamma \mu^F(s')$ and the variance

$$\Sigma^F(s) = k'(s)^2 + 2\gamma k'(s) \mathop{\mathbb{E}}_{s' \sim \mathcal{P}^{\pi}} \left[\Sigma^F(s')\right] + \gamma^2 \mathop{\mathbb{E}}_{s' \sim \mathcal{P}^{\pi}} \left[\Sigma^F(s') + \left(\mu^F(s')\right)^2\right] - \left(\mu^F(s)\right)^2$$

Here, $k'(s) = \max(k(s), 0)$ and $\mathcal{P}^{\pi}(s'|s)$ is the transition probability under policy π. The TD error between the target distribution $\mathcal{N}(\mu^F(s), \Sigma^F(s))$ and the parameterized distribution $\mathcal{N}(\mu_\phi^F(s), \Sigma_\phi^F(s))$ with respect to the 2-Wasserstein distance can be computed as

$$\mathcal{L}_F = \|\mu^F(s) - \mu_\phi^F(s)\|^2 + \mathrm{Tr}\left(\Sigma^F(s) + \Sigma_\phi^F(s) - 2\left(\Sigma^F(s)^{1/2}\Sigma_\phi^F(s)\Sigma^F(s)^{1/2}\right)^{1/2}\right). \quad (3.2)$$

Since the value function is one-dimensional, we simplify Equation (3.2) as

$$\mathcal{L}_F = \|\mu^F(s) - \mu^F_\phi(s)\|^2 + \left\|\sqrt{\Sigma^F(s)} - \sqrt{\Sigma^F_\phi(s)}\right\|^2 .$$

Given that $V^F_\pi \geqslant 0$, we use a *Softplus* activation for the mean and a *Exponetial* parameterization for the standard deviation to ensure positiveness.

3.2.2. Implicit Quantile Network

IQN [14, 49] is a parametric model representing the quantile function of the distribution, which takes a quantile value τ as input and outputs a threshold value z so that the probability of Z being less or equal to z is τ. Let $\eta^\tau_\phi(s)$ be the quantile function at $\tau \in [0, 1]$ for the random feasibility value at state s. The TD error between two samples $\tau, \tau' \sim U([0, 1])$ for the transition (s, a, s', r, k) is

$$d^{\tau,\tau'}_\phi = k'(s) + \gamma\eta^{\tau'}(s') - \eta^\tau_\phi(s)$$

The IQN model can be optimized via the Huber quantile regression loss

$$\mathcal{L}_\tau(d) = |\tau - \mathbb{I}\{d\}|\mathcal{L}_k(d), \quad \text{where } \mathcal{L}_k(d) = \begin{cases} d^2/2k, & |d| < k \\ |d| - k/2, & \text{otherwise} \end{cases} \tag{3.3}$$

We use the network structure as proposed in [14] with cosine embeddings for the τ and a multilayer perceptron for the state embeddings.

With the approximators for the FVF in place, we will now introduce the risk measures we use to ensure safety during learning.

3.3. Chance Constraint with (Conditional) Value-at-Risk

Value-at-Risk (VaR) and CVaR quantify the risk of a random variable. The VaR is the smallest value such that the probability of Z is bigger than a value α. The CVaR risk measure is the mean of the α-tail of the distribution of Z.

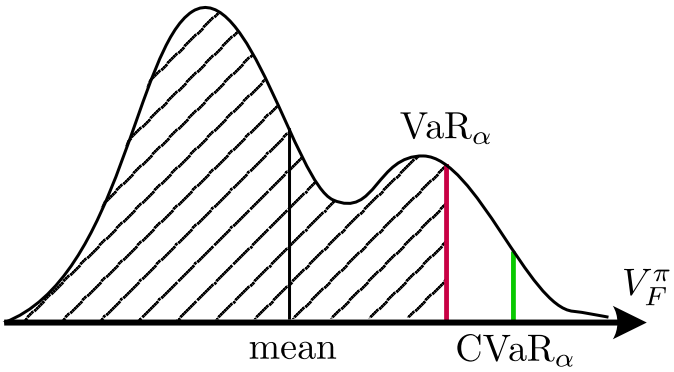


Figure 3.1.: Distribution of V_F^π and illustration of mean, VaR (red), and CVaR (green). The shaded area shows the cumulative probability α.

$$\mathrm{VaR}_\alpha(Z) = \inf\{z \in \mathbb{R} | F(z) \geqslant \alpha\}, \tag{3.4a}$$

$$\mathrm{CVaR}_\alpha(Z) = \mathbb{E}\left[z | z \geqslant \mathrm{VaR}_\alpha(Z)\right]. \tag{3.4b}$$

where $F(z)$ is the CDF of the random variable Z. When F is continuous and strictly increasing, the VaR is uniquely defined as $\mathrm{VaR}_\alpha(Z) = F^{-1}(\alpha)$, i.e., the quantile function. The VaR and CVaR offer a risk-aware constraint formulation by restricting the VaR or CVaR to be smaller than a threshold δ. The CVaR constraint for the Gaussian approximator of FVF is

$$\mathrm{CVaR}_\alpha^F(s) \coloneqq \mu^F(s) + \frac{1}{1-\alpha}\varphi(\Phi^{-1}(\alpha))\Sigma^F(s) \leqslant \delta, \tag{3.5}$$

where φ and Φ are the Probability Density Function (PDF) and the CDF of the standard normal distribution, respectively. The constant α determines the probability of constraint satisfaction, thus, the risk is $1-\alpha$. Considering the CVaR constraints (3.5), the constrained optimization problem as described in Equation (2.1) can be solved with ATACOM. Since ATACOM exploits the prior knowledge of the robot dynamics, the RL agent will converge to a safe policy as the estimation of FVF is getting accurate.

3.3.1. Adaptive constraint threshold estimate

While ideally, we would like to have the FVF to be always equal to zero, setting $\delta = 0$ is neither a practical choice since the network's mean is always bigger than 0, nor beneficial

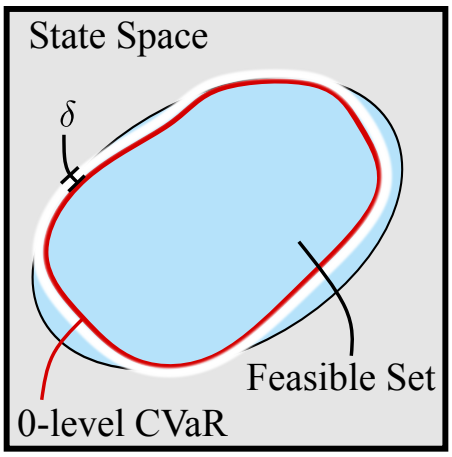


Figure 3.2.: Illustration of the feasible set (light blue), the learned FVF at 0-level red and threshold δ. The threshold δ provides a small feasible region (white) to explore within a small cost budget.

for the training of FVF as it restricts the exploration. The threshold δ trades off the constraint violation and the exploration, sometimes requiring further engineering.

To alleviate the engineering effort, we propose an adaptive scheme that updates the δ based on the current episodic cost and the estimation of the FVF. We use a Softplus parametrization to keep the δ positive. The δ parameter is updated during the learning process after each episode using the following loss

$$\mathcal{L}_\delta = \frac{1}{H} \sum_{i=0}^{H} L_{\text{Huber}} \left(d_c(s_i), \text{CVaR}_\alpha(s_t) - \delta \right) \tag{3.6}$$

with the horizon of the episode H, the accepted discounted cost budget $\bar{C}$ and the the predictive cost $\text{CVaR}_\alpha(s_t)$. The term $d_c(s_i) = \sum_{t=i}^{H} \gamma^{t-i} k'(s_t) - \bar{C}$ computes the difference between the emperical discounted cost starting from s_i. We use *Huber Loss* to obtain a more robust update against outliers. Intuitively, the δ are tuned such that the CVaR constraint is increased by the accepted cost budget. As the FVF estimation is becoming accurate, the discounted cost and the CVaR will converge to zero since D-ATACOM ensures safety, the Huber Loss in the end minimizes the difference between δ and $\bar{C}$.

3.4. Policy Iteration with Learnable Constraint using ATACOM

As introduced in Section 2.3, ATACOM constructs a safe action space by determining the basis vectors of the tangent space of the constraint manifold. However, previous work assumes that the constraint is given, fixed, and deterministic. This assumption is no longer valid when the constraint is trained during the learning process. Since the constraint function changes during training, the safe action space changes accordingly, leading to a non-stationary MDP, which causes the failure of the training. Specifically, let $\boldsymbol{a} \in \mathcal{A}$ be the action applied to the environment and the $\boldsymbol{u} \in \mathcal{U}$ be the control input obtained from the policy $\boldsymbol{u} \sim \pi(s)$. ATACOM constructs an affine mapping $W : \mathcal{U} \to \mathcal{A}$, defined in (2.3), that maps an action in safe space into the original one. Combining ATACOM with the actor-critic framework, a value function estimator $Q_{\boldsymbol{\omega}}(\boldsymbol{s}, \boldsymbol{u})$ is trained to approximate the expected return. Then, the policy $\pi_{\boldsymbol{\theta}}(\boldsymbol{s})$ is updated by maximizing the $Q_{\boldsymbol{\omega}}(\boldsymbol{s}, \boldsymbol{u})$. However, when the constraint is updated during the training process, the action mapping W will also change. The same action $\boldsymbol{u}$ will result in different $\boldsymbol{a}$ at different steps. This variation of the action space leads to an unstable update of $Q_{\boldsymbol{\omega}}(\boldsymbol{s}, \boldsymbol{u})$ and $\pi_{\boldsymbol{\theta}}(\boldsymbol{s})$.

In the following, we will demonstrate how to address this problem for the Soft Actor Critic (SAC) algorithm [24]. However, it is possible to use the same methodology to extend most Deep RL algorithms, e.g., DDPG [27] TD3 [21], PPO [36]. To solve this issue, we learn the value function of the original action space $Q_{\boldsymbol{\omega}}(\boldsymbol{s}, \boldsymbol{a})$, which is invariant to the constraints. Since $Q_{\boldsymbol{\omega}}$ is represented by a neural network and can be differentiated, we can use the reparameterization trick to obtain the gradient for $\boldsymbol{\theta}$ (similar to TD3 and SAC [24]), the objective and policy gradient can be obtained as

$$\max_{\pi_\theta} J^\pi = \mathop{\mathbb{E}}_{s, \boldsymbol{u} \sim \pi_{\boldsymbol{\theta}}(s)} \left[Q_{\boldsymbol{\omega}}(s, W(\boldsymbol{u})) \right], \qquad \nabla_{\boldsymbol{\theta}} J_\pi = \nabla_a Q_{\boldsymbol{\omega}}(s, a) \nabla_{\boldsymbol{u}} W(\boldsymbol{u}) \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}}(s).$$

Note that in SAC, the soft Q-function includes the entropy term $H(W(\pi_{\boldsymbol{\theta}}(s)))$ to encourage exploration. The entropy is computed as

$$H(\pi_{\boldsymbol{\theta}}(s)) = - \mathop{\mathbb{E}}_{\pi_{\boldsymbol{\theta}}(s)} \left[\log W(\pi_{\boldsymbol{\theta}}(s)) \right].$$

This means the entropy term is indeed constraint-dependent. Thus, updating the constraints may change the entropy of the policy, consequently, the estimated value function is not proper anymore. We argue that practically, the variation of the entropy bonus has a negligible effect on the training stability because the entropy term is scaled down by a coefficient. Furthermore, the online training of the value function allows us to quickly

Algorithm 1 D-ATACOM with constraint learning

Initialize: FVF network parameters ϕ, number of steps N, threshold δ, cost budget $\bar{C}$.
for $1 \cdots N$ **do**
  Construct $\mathrm{CVaR}_\alpha^F(s_t)$ using $\mu_\theta^F(s_t)$, $\Sigma_\theta^F(s_t)$ from Equation (3.5).
  Draw action in safe action space u_t and obtain the actual action a_t from Equation (2.3).
  Observe s_{t+1}, r_t, k_t from the environment.
  Save replay buffer $(s_t, a_t, r_t, k_t, s_{t+1}) \rightarrow \mathcal{D}$ and $(s_t, k_t, s_{t+1}) \rightarrow \mathcal{D}_f$ if $k_t > 0$.
  If the episode terminates, update δ using Equation (3.6).
  Sample a batch of transitions (s, a, r, k, s') from $\mathcal{D} \cup \mathcal{D}_f$.
  Update ϕ using Equation (3.2), $\phi \leftarrow \phi - \alpha_\phi \nabla_\phi \mathcal{L}_F$
  Update value function and policy π with RL, the update with SAC is shown in Algorithm 2.
end for

Algorithm 2 SAC implementation for D-ATACOM

Initialize: Batch of transitions $\mathcal{B} = (s, a, r, k, s')$, policy parameters θ and Q-function parameters ψ_1, ψ_2.
Draw next action in safe action space u' and obtain the actual next action a' and B'_u from Equation (2.3).
Compute FVF adjusted log probability $\log p' = \log \pi_\theta(a'|s') - \log |B'_u|$
Update Q-functions with the TD loss $\mathcal{L}_\psi = \frac{1}{|\mathcal{B}|}(Q_\psi(s,a) - r - \gamma(Q_\psi(s', a') + \alpha \log p'))^2$.
Draw actions u_θ and obtain a_θ that are differentiable w.r.t. θ, obtain B_u from Equation (2.3).
Compute FVF adjusted log probability, $\log p_\theta = \log \pi_\theta(a|s) - \log |B_u|$
Update policy with the gradient $-\nabla_\theta \frac{1}{|\mathcal{B}|}\left(Q_\psi(s, a_\theta) + \alpha \log p_\theta\right)$

adapt to the new policy entropy. Algorithm 1 demonstrates the general structure of the D-ATACOM with constraint learning. Algorithm 2 shows the implementation of SAC with the ATACOM framework.

During training, we keep a replay buffer of limited size. As training progresses, the agent will behave safer and encounter fewer constraint violations, flushing away unsafe transitions when using a single replay buffer. Instead, we would like the agent to remember the failures and avoid being overly optimistic, using a separate, smaller, *Failure Buffer* $\mathcal{D}_f$ to store the unsafe transitions. In each data batch, we sample a proportional number of data coming from this buffer.

4. Experimental Evaluation

In this chapter, we compare the performance of our approach in three different environments with different characteristics. We compare with SafeRL baselines such as the *LagSAC* [23] and the *WCSAC* [48]. All environments and algorithms based on SAC are implemented using the MushroomRL framework [17]. We use the implementations provided by OmniSafe [25] for PPO-based algorithms. We conducted a hyperparameter search on the learning rates, cost budget, and accepted risk with 10 random seeds. We present the learning performance with 25 seeds for the hyperparameters with the best trade-off between performance and safety. In additional experiments, we also compare the learned behavior and the impact of certain hyperparameters on the learning performance.

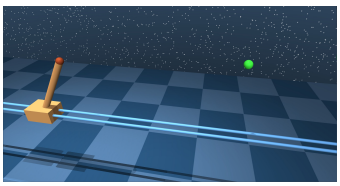
4.1. Learning Performance

For the Learning Performance, we conducted a hyperparameter search on the learning rates, cost budget, and accepted risk with 10 random seeds. We present the learning performance with 25 seeds for the hyperparameters with the best trade-off between performance and safety. In additional experiments, we also compare the learned behavior and the impact of certain hyperparameters on the learning performance.

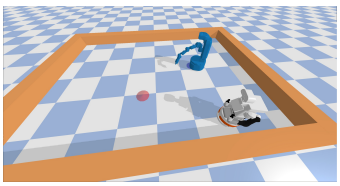
Further hyperparameter search experiments and tables with all used hyperparameters are in Appendix A.

4.1.1. Cartpole Environment

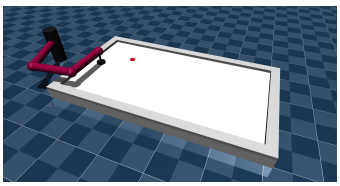
The cartpole environment, depicted in Figure 4.1a is a classic control problem with the goal of moving the pole tip to a desired position (green point) by controlling a cart. The pole has a length of 1 unit and is initialized in an upright position on the cart. The cart



(a) Cartpole Environment



(b) Navigation Environment



(c) Air Hockey Environment

Figure 4.1.: The three Environments used for evaluation of all algorithms

can move on a rail 10 units long. The cart is initialized on the left side of the rail, and the goal is to move the cart towards the goal position of the pole tip on the right rail's side while keeping the pole upright. Despite the simplicity of the environment, designing a feasible long-term constraint is very challenging due to the actuator and cart position limits.

The state space of the environment is $s = [x, \sin\theta, \cos\theta, \dot{x}, \dot{\theta}]^T$ where x is the position of the cart, $\dot{x}$ is the velocity of the cart, θ is the angle of the pole with the vertical axis, and $\dot{\theta}$ is the angular velocity of the pole. The action space is $a \in [-1, 1]$ where the action is the force applied to the cart.

The reward function given a goal position x_G and pole tip position x_T is defined as $r(s) = \text{clip}(1 - \frac{\|x_G - x_T\|}{4}, 0, 1)$. The constraint function prevents the pole from deviating more than π from the vertical axis. Thus we define the cost function as $c(s) = \max\left(\frac{\theta}{0.5\pi} - 1, 0\right)$.

As we can see from the results in Figure 4.2, our approach is the best-performing one among the SafeRL baselines in terms of learning speed, while achieving small constraint violations. SAC achieves a policy with higher performance, but this policy heavily violates the safety constraints as it completely disregards the pole angle constraint while reaching the goal. The RCPO algorithm [45] achieves better performance, but violations are comparable with SAC. Notice that D-ATACOM requires a feasible cost budget to generate feasible actions since a single constraint violation will result in a high sum cost and the complete maximum violations. We investigate the impact of the cost budget parameter in the Cartpole task in experiment 4.2.1.

4.1.2. Navigation

The Navigation task consists of two robots, one differential-driven TIAGo++ (white) that learns a navigation policy to a goal while avoiding the Fetch robot (blue), as shown in

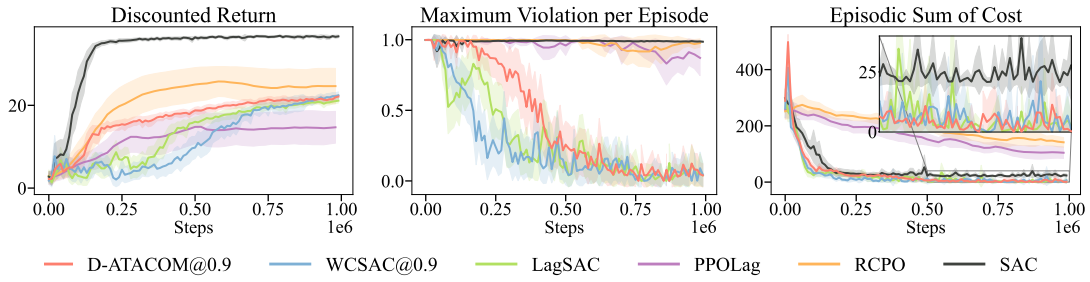


Figure 4.2.: Learning Curves for the Cartpole Environment

Figure 4.1b. The Fetch robot constantly moves its robotic arm in a periodic motion, such that the end-effector draws a lemniscate into the air in front of the robot. Additionally, the Fetch robot constantly moves to a randomly assigned target position using a hand-crafted policy that ignores the TIAGo. The agent controls the TIAGo robot to reach the target position while avoiding the Fetch robot, which serves as a dynamic obstacle. In this setting, the impact of the arm's motion on the constraint is not explicit, which generates stochasticity on the constraint.

The state space consists of the cartesian position and velocity of the two robots, the target position of the TIAGo, the previous action, and the cartesian position and velocity of Fetch's end-effector. The action space is the linear velocity in the x-direction and angular velocity around the z-axis of the TIAGo robot. These are converted into the left and right wheel velocities.

Given the distance to the goal d_G, the current orientation θ and the goal orientation θ_G

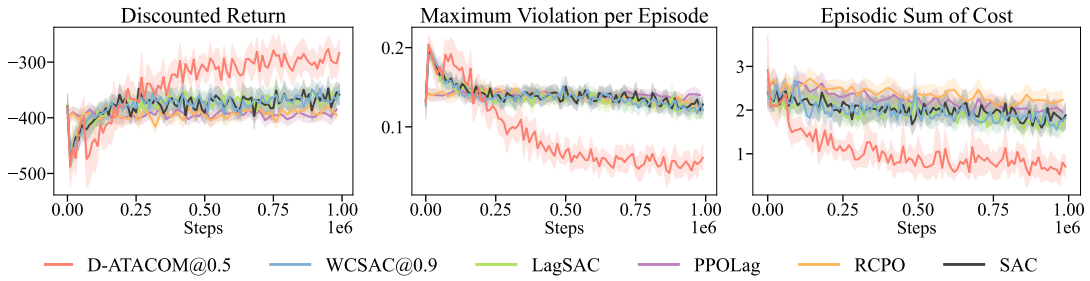


Figure 4.3.: Learning Curves for the Navigation Environment

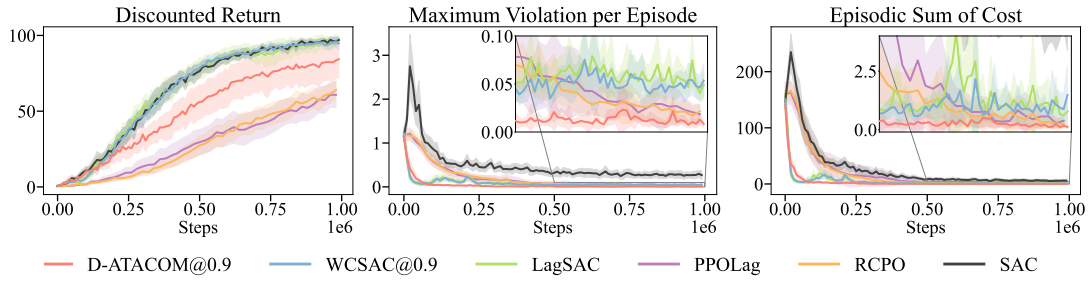


Figure 4.4.: Learning Curves for the Air Hockey Environment

the reward is defined as:

$$r(s) = -\|d_G\| - \text{sigmoid}\left(30(\|d_G\| - 0.2)\right)\frac{\theta_G - \theta}{\pi} - 0.1\|a\|$$

The constraint is the smallest 2d cartesian distance between the TIAGo base and every joint of the Fetch Robot. Additionally, the constraint also prevents the TIAGo from hitting the surrounding walls. Given the TIAGos' position p_T and cartesian position of the ith Fetch joint p_F^i the Fetch cost is $c_F(s) = \max_i(-(\|p_T - p_F^i\| - \omega))$ where ω is a constant that accounts for the width of the robots. The wall cost is defined as $c_W(s) = \max_i(-(d_{\text{wall}}^i - \omega))$ where d_{wall}^i is the distance to the ith wall. The step cost is $c(s) = \max(c_F(s), c_W(s))$.

From the result in Figure 4.3, D-ATACOM clearly outperforms all the state-of-the-art methods both in terms of safety and final task performance. Inspecting the learned policies, it is clear that D-ATACOM is the only approach presenting active collision avoidance behaviors, including driving backward to prevent collision. These collision avoidance behaviors are mostly achieved by the model-based treatment of the constraint function. In this setting, using the constraint gradient and the model of the environment is a strong inductive bias for the method. On top of that, the control system can exploit the physical meaning of the variables, such as other obstacle velocity, allowing it to compensate in advance for the other robot movements. In this task, most approaches behave the same. We argue that in this task, the conflict between the task objective, pushing the robot to disregard the obstacle, and the constraint function, forcing the robot to cause a detour, is problematic for the Lagrangian approaches. Indeed, Lagrangian optimization is trying to balance constraint satisfaction and policy improvement in the update step, possibly causing the algorithm to get stuck in local minima.

4.1.3. 3dof Robot Air Hockey

The objective of this task is to control a 3-DoF arm to score a goal in the robot air hockey task, shown in Figure 4.1c. The episode terminates when the puck enters the goal or hits one of the table's walls.

The state space consists of the robots' joint positions, velocities, puck position, and velocity. The action space is the acceleration setpoint for each robot joint.

The reward for non-absorbing states is the change of distance between the puck and the goal. In absorbing states the reward depends on the distance of the puck to the goal. Given the puck position $[x^t, y^t]^T$ at timestep t and the distance between puck and goal as d^t, we define the reward as:

$$r(s_t) = \begin{cases} 50(d^{t-1} - d^t) & \text{if not absorbing} \\ \rho(1.5 - 5 \cdot \text{clip}(|y^t|, 0, 0.1)) & \text{if puck in goal} \\ \rho(1 - 2 \cdot \text{clip}(|y^t| - 0.1, 0, 0.35)) & \text{if puck on backboard next to goal} \\ \rho(0.3 - 0.3 \cdot \text{clip}(1.43 - |x^t|, 0, 1)) & \text{if puck on sidebars} \\ 0 & \text{otherwise} \end{cases}$$

where ρ is a constant that scales the reward. The constraint prevents the mallet from touching the sides of the table and the robot from violating its joint position and joint velocity limits. The mallet cost is defined as $c_M(s) = \max_i(-d^i_W + \omega)$ where d^i_W is the distance to the ith wall and ω is a constant that accounts for the width of the mallet. Given the joint positions q_i and the joint velocities $\dot{q}_i$ the position cost is $c_P(s) = \max_i(q_i - q_{u,i}, -q_i + q_{l,i})$ and the velocity cost is $c_V(s) = \max(\{\dot{q}_i - \dot{q}_{u,i}, -\dot{q}_i + \dot{q}_{l,i})$. The total cost is $c(s) = \max(c_P(s), c_V(s), c_M(s), 0)$ Since the optimal strategy for hitting the puck toward the goal is achievable within the tables's boundary, high reward performance and low constraint violations are achievable at the same time.

In this task, D-ATACOM is safer than the other baselines at the cost of slower learning performance, as shown in Figure 4.4. The reason for this performance drop is that our approach learns to expand the safe region progressively and improve the performance. The final performance is lower as the robot hits more cautiously at the boundary regions due to strict constraint satisfaction, while other approaches allow the robot to go outside for stronger hitting, as shown in Experiment 4.2.3. A characteristic of this environment is that the constraints do not majorly affect an optimal policy. Therefore, constraint satisfaction is more difficult in the initial phases of learning than in the final one. This allows classical

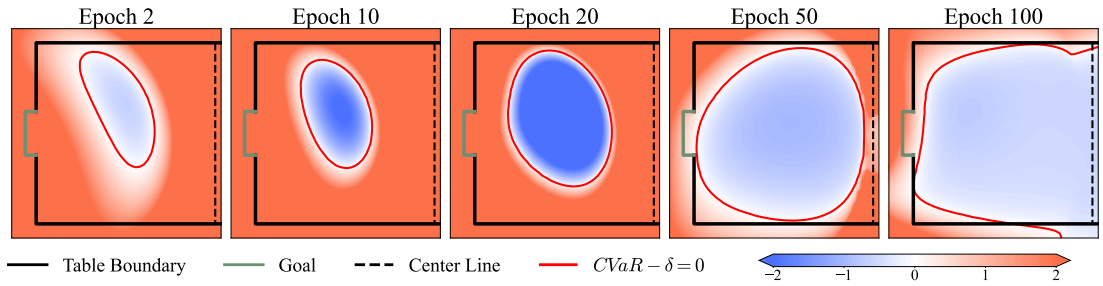


Figure 4.5.: FVF for the Air Hockey Environment. The colored region shows half of the air hockey table on the agent's side. We compute the value of $\mathrm{CVaR} - \delta$ for each end-effector's position on the table with 0 velocity. After a short training with an initial dataset, The feasible region shrinks to a small region on the table at Epoch 2 and then increases progressively reaching considerable coverage at the end of training.

lagrangian methods to be particularly competitive in the task. Figure 4.5 illustrates the learned constraint at different training steps. We can clearly observe that the feasible region expands progressively and reaches a good coverage at the final epoch. Since FVF is a policy-dependent value function, the prediction at the corner regions is poor, as the policy does not reach these states with the hitting behavior.

4.2. Additional Experiments

In this section, we investigate the impact of the cost budget and accepted risk hyperparameters on the performance of D-ATACOM. We also analyze the final performance of the policies in the air hockey task to understand the differences that lead to the performance gap.

4.2.1. Impact of Cost Budget

In this experiment, we will compare the impact of the cost budget parameter on D-ATACOM and WCSAC. We chose the CartPole task for this comparison because both algorithms do not learn a completely safe policy. Figure 4.6 shows the performance of D-ATACOM and WCSAC with different cost budgets. We can observe that the performance of D-ATACOM

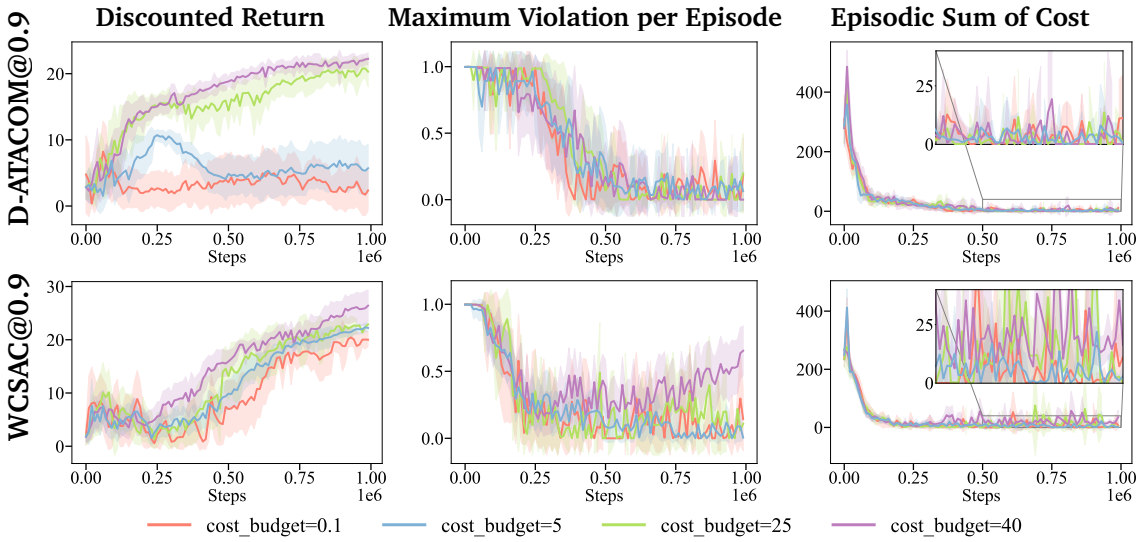


Figure 4.6.: Impact of the cost budget parameter on D-ATACOM and WCSAC performance in the CartPole task

is more sensitive to the cost budget parameter compared to WCSAC. When the policy cannot achieve the given cost budget the performance of D-ATACOM degrades significantly. This performance drop occurs because the delta eventually will converge towards zero, which results in a very conservative policy. The behavior for D-ATACOM with the cost budgets of 0.1 and 5 is balancing to the pole in its initial position because the policy is too conservative to move towards the goal, as this will lead to constraint violations.

On the other hand, WCSAC is more robust w.r.t. the cost budget parameter. An unreasonable cost budget will increase the Lagrange multiplier, giving more weight to the constraint. The difference is that the Lagrange multiplier does not set an explicit limit to the constraint like the delta does in D-ATACOM. Instead, WCSAC gives more weight to the constraint violations in the optimization problem, which has less impact on policy performance. Is worth noting that, depending on the application, one of the two behaviors would be preferable. In safety-critical applications, having an algorithm that strongly enforces the constraint violation, independently of the performance, is preferable. Instead, when partial constraint satisfaction is enough, it may be better to choose a lagrangian-based algorithm.

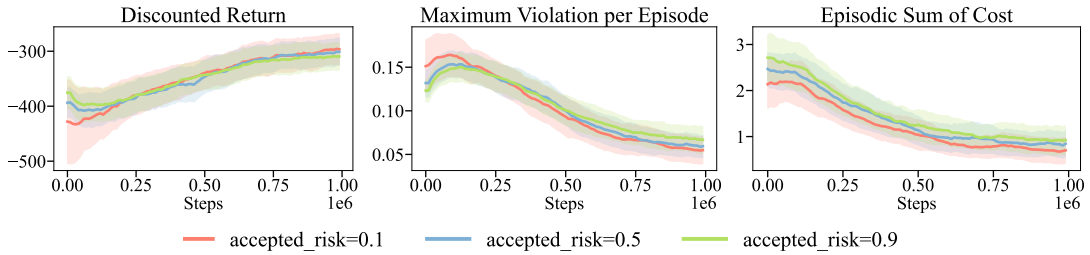


Figure 4.7.: Impact of accepted risk on performance in the Navigation task with a fixed delta. The plots are smoothed via the exponential moving average with 0.9 weight

4.2.2. Experiment with different Accepted Risk

In the distributional setting, the parameter accepted risk determines how much of the tail of the distribution we are willing to violate, i.e., how much risk we want to take. However, this is not the only parameter that influences the safety of a policy. Usually, there is another parameter that is tuned with a given cost budget that also influences how safe the behavior is. For WCSAC this parameter is the Lagrange multiplier beta, and for D-ATACOM it is the learned δ. To show the complete impact of the accepted risk, we fix δ to a constant value such that it cannot compensate for the difference in the accepted risk. Figure 4.7 shows the performance of D-ATACOM with a fixed delta and different levels of accepted risk in the Navigation task. Clearly, a lower accepted risk leads to safer behavior.

The impact of the accepted risk on the safety shrinks for D-ATACOM when the delta is learned. Delta can compensate for a high accepted risk, resulting in the same safe policy as a lower accepted risk would produce. The accepted risk has an impact in this setting toward the beginning of the training when delta is not yet converged. Thus accepted risk determines how risky the exploration at the beginning of the training will be. Figure 4.8 shows the impact of different accepted risk settings on the air hockey task. The lower accepted risk explores slower, thus the discounted return converges slower. The maximum violation and sum of cost are comparable for all accepted risk settings because, in the air hockey task, the constraint does not majorly affect the optimal policy.

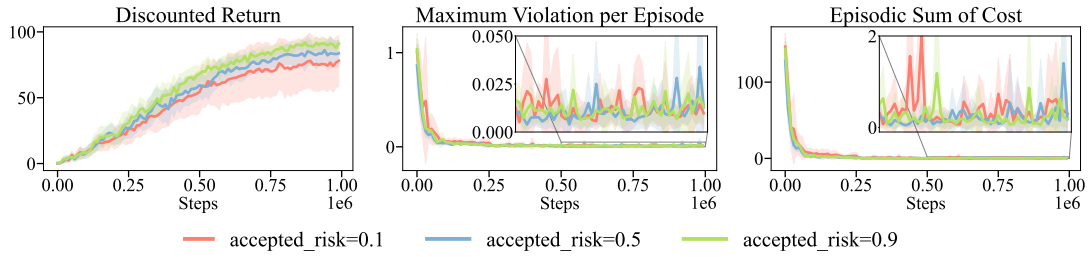


Figure 4.8.: Impact of accepted risk on performance the air hockey task

4.2.3. Analysis of Air Hockey

In the air hockey task, D-ATACOM cannot reach the same discounted return as LagSAC and WCSAC. We investigate the final performance of the policies to understand the differences that lead to the performance gap. As D-ATACOM results in a safer policy, we theorize that performance is lost when the puck is initialized too close to the edge of the table. To test this hypothesis, we evaluate the performance of the final policies with an adjusted region for the initial puck position, that omits these critical positions. Figure 4.9 shows the performance for the original and adjusted regions and the difference between them.

The original region D-ATACOM has significant outliers in the discounted return compared to WCSAC and LagSAC. However, LagSAC and WCSAC have more outliers in the maximum violation and sum of cost. Thus, WCSAC and LagSAC sacrifice safety to gain a stable performance. The safe exploration of D-ATACOM results in the opposite behavior, where the policy will sacrifice performance to ensure safety.

When we evaluate the performance with the adjusted region, we can observe that the discounted return of D-ATACOM increases more compared to WCSAC and LagSAC. Additionally, the decrease in maximum violation and sum of cost is more significant for LagSAC and WCSAC. This result confirms our hypothesis that D-ATACOM does not properly hit the puck when it is too close to the edge of the table because it is not possible to do so safely. WCSAC and LagSAC learn to hit the puck in these critical positions, but this comes at the cost of safety.

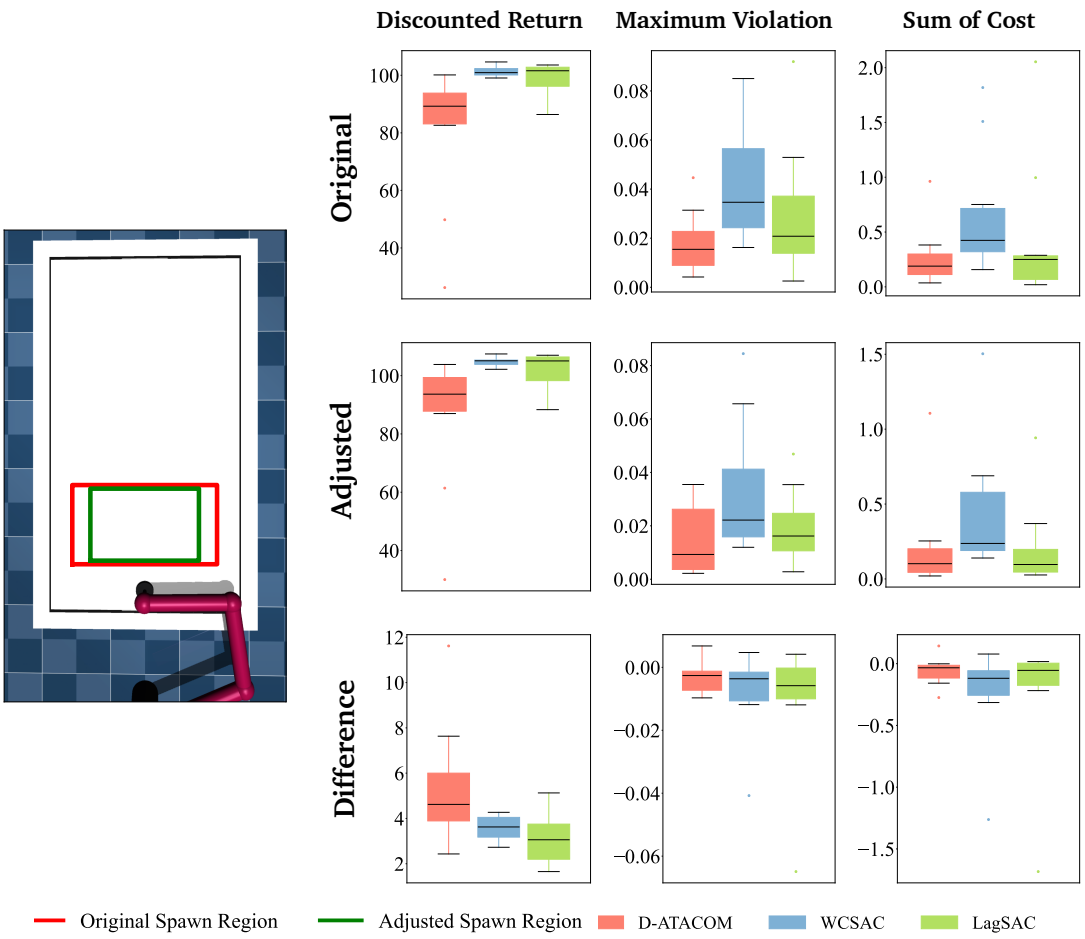


Figure 4.9.: Performance of the final policy from D-ATACOM, WCSAC, and LagSAC in the air hockey task. The performance is evaluated with the original and an adjusted region for the initial puck position.

5. Conclusion

In this thesis, we started to bridge the gap between model-based SafeExp methods and model-free SafeRL approaches. We extended the ATACOM framework to work with learned constraints, by dropping some key assumptions of the original formulation, such as known constraints and local safety. We directly learn the constraint in the form of a distributional FVF from data, which allows us to estimate the total uncertainty of the model. D-ATACOM produces a risk-aware policy by restricting the level of acceptable risk, allowing the agent to explore more cautiously whenever the constraints are uncertain or the policy is close to violating them.

Our results show that our method is competitive with state-of-the-art approaches, outperforming them in terms of safety, keeping an on-par learning speed, and achieving similar or better performance for environments with different characteristics. Also, the method does not require excessive parameter tuning, as it includes automatic tuning rules for the most important hyperparameters. Therefore, our work proves that including prior knowledge in data-driven methods can actually be beneficial for scaling SafeRL approaches. Although all of the experiments are trained from scratch, we believe starting with an offline dataset and pre-training will significantly reduce the initial violations, leading to safe performance.

At the moment D-ATACOM cannot solve complex control tasks, such as the 7-DoF Robot Air Hockey task, without predefined constraints. Exploring such a large space while trying to accurately estimate the complex constraints is too hard for the current implementation. D-ATACOM ends up overestimating the constraints, which leads to overly cautious behavior that does not resemble the desired behavior. Unfortunately, this makes the deployment of our method on complex real-world tasks problematic. Nevertheless, we believe that D-ATACOM provides a solid foundation for future research in this direction. Another limitation of this approach is that it requires knowledge of the robot dynamics. While this is often the case in many applications, for other approaches, it may be problematic to

derive even an approximate model. Lastly, our experiments provided a dense cost function, which is not always available in real-world applications.

In future work, we will further investigate how to integrate known local constraints with long-term safety. This can be easily integrated into the D-ATACOM framework and will allow scaling the ATACOM approach to real-world robotics tasks involving complex long-term constraints and human-robot interaction. Additionally, we will explore the impact of sparse cost functions on the performance of the method. Another interesting direction is to learn the model dynamics from data, which will allow our approach to be applied to a wider range of tasks. It has been shown that ATACOM is still effective with a moderate model mismatch, thus learning the dynamics from data should be feasible. Finally, using a more structured constraint instead of the FVF might lead to better performance. Combining a stepwise constraint (2.2b) with a linear add-on that ensures long-term safety similar to viability constraints might lead to better constraint estimation and gradients which are crucial for ATACOMs performance.

Bibliography

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained Policy Optimization. In *International Conference on Machine Learning (ICML)*, 2017.

[2] Eitan Altman. Constrained Markov Decision Processes with Total Cost Criteria: Lagrangian Approach and Dual Linear Program. *Mathematical methods of operations research*, 48(3):387–417, 1998.

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

[4] Haitham Bou Ammar, Rasul Tutunov, and Eric Eaton. Safe policy search for lifelong reinforcement learning with sublinear regret. In *International Conference on Machine Learning*. PMLR, 2015.

[5] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pages 449–458. PMLR, 2017.

[6] Marc G. Bellemare, Will Dabney, and Mark Rowland. *Distributional Reinforcement Learning*. MIT Press, 2023. `http://www.distributional-rl.org`.

[7] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemyslaw Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Christopher Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique Pondé de Oliveira Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. Dota 2 with large scale deep reinforcement learning. *CoRR*, abs/1912.06680, 2019. URL `http://arxiv.org/abs/1912.06680`.

[8] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai

Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars, 2016.

[9] Vivek Borkar and Rahul Jain. Risk-constrained markov decision processes. *IEEE Transactions on Automatic Control*, 59(9):2574–2579, 2014.

[10] 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.

[11] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A Lyapunov-based Approach to Safe Reinforcement Learning. In *Conference on Neural Information Processing Systems (NIPS)*, 2018.

[12] Yinlam Chow, Ofir Nachum, Aleksandra Faust, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. Lyapunov-based Safe Policy Optimization for Continuous Control. In *Reinforcement Learning for Real Life (RL4RealLife) Workshop in the 36 th International Conference on Machine Learning*, 2019.

[13] Alexander I Cowen-Rivers, Daniel Palenicek, Vincent Moens, Mohammed Amin Abdullah, Aivar Sootla, Jun Wang, and Haitham Bou-Ammar. Samba: Safe model-based & active reinforcement learning. *Machine Learning*, pages 1–31, 2022.

[14] Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pages 1096–1105. PMLR, 2018.

[15] Will Dabney, Mark Rowland, Marc Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32(1), 2018.

[16] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe Exploration in Continuous Action Spaces. *arXiv preprint arXiv:1801.08757*, 2018.

[17] Carlo D'Eramo, Davide Tateo, Andrea Bonarini, Marcello Restelli, and Jan Peters. Mushroomrl: Simplifying reinforcement learning research. *Journal of Machine Learning Research*, 22(131):1–5, 2021. URL `http://jmlr.org/papers/v22/18-056.html`.

[18] Dongsheng Ding, Xiaohan Wei, Zhuoran Yang, Zhaoran Wang, and Mihailo R. Jovanovic. Provably Efficient Safe Exploration via Primal-Dual Policy Optimization. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 130, 2021.

[19] Jingliang Duan, Yang Guan, Shengbo Eben Li, Yangang Ren, Qi Sun, and Bo Cheng. Distributional soft actor-critic: Off-policy reinforcement learning for addressing value estimation errors. *IEEE Transactions on Neural Networks and Learning Systems*, 33 (11):6584–6598, 2022. doi: 10.1109/TNNLS.2021.3082568.

[20] 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, July 2019. ISSN 0018-9286, 1558-2523, 2334-3303. doi: 10.1109/TAC.2018.2876389. URL `https://ieeexplore.ieee.org/document/8493361/`.

[21] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[22] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3389–3396, 2017. doi: 10.1109/ICRA.2017.7989385.

[23] Sehoon Ha, Peng Xu, Zhenyu Tan, Sergey Levine, and Jie Tan. Learning to walk in the real world with minimal human effort. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 1110–1120. PMLR, 16–18 Nov 2021. URL `https://proceedings.mlr.press/v155/ha21c.html`.

[24] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[25] et al. Jiaming Ji, Jiayi Zhou. Omnisafe: An infrastructure for accelerating safe reinforcement learning research. *arXiv preprint arXiv:2305.09304*, 2023.

[26] Dohyeong Kim and Songhwai Oh. Efficient off-policy safe reinforcement learning using trust region conditional value at risk. *IEEE Robotics and Automation Letters*, 7 (3):7644–7651, 2022.

[27] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous Control with Deep Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2016.

[28] Puze Liu, Davide Tateo, Haitham Bou Ammar, and Jan Peters. Robot Reinforcement Learning on the Constraint Manifold. In *Conference on Robot Learning*, pages 1357–1366. PMLR, 2022.

[29] Puze Liu, Kuo Zhang, Davide Tateo, Snehal Jauhri, Zhiyuan Hu, Jan Peters, and Georgia Chalvatzaki. Safe Reinforcement Learning of Dynamic High-Dimensional Robotic Tasks: Navigation, Manipulation, Interaction. In *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2023.

[30] Puze Liu, Haitham Bou-Ammar, Jan Peters, and Davide Tateo. Safe reinforcement learning on the constraint manifold: Theory and applications. *arXiv preprint arXiv:2404.09080*, 2024.

[31] Simin Liu, Changliu Liu, and John Dolan. Safe control under input limits with neural control barrier functions. In *Conference on Robot Learning*, pages 1970–1980. PMLR, 2023.

[32] Yongshuai Liu, Jiaxin Ding, and Xin Liu. Ipo: Interior-point policy optimization under constraints. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 34(04), pages 4940–4947, 2020.

[33] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Kirkeby Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015. URL `https://api.semanticscholar.org/CorpusID:205242740`.

[34] Baiyu Peng, Yao Mu, Jingliang Duan, Yang Guan, Shengbo Eben Li, and Jianyu Chen. Separated proportional-integral lagrangian for chance constrained reinforcement learning. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 193–199. IEEE, 2021.

[35] Samuel Pfrommer, Tanmay Gautam, Alec Zhou, and Somayeh Sojoudi. Safe reinforcement learning with chance-constrained model predictive control. In *Learning for Dynamics and Control Conference*, pages 291–303. PMLR, 2022.

[36] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[37] Harshit Sikchi, Wenxuan Zhou, and David Held. Lyapunov barrier policy optimization. *arXiv preprint arXiv:2103.09230*, 2021.

[38] David Silver, Aja Huang, Christopher J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529:484–503, 2016. URL `http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html`.

[39] Kyle Stachowicz and Sergey Levine. Racer: Epistemic risk-sensitive rl enables fast driving with fewer crashes. *arXiv preprint arXiv:2405.04714*, 2024.

[40] Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive Safety in Reinforcement Learning by PID Lagrangian Methods. In *International Conference on Machine Learning (ICML)*, 2020.

[41] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018. URL `http://incompleteideas.net/book/the-book-2nd.html`.

[42] Daniel CH Tan, Fernando Acero, Robert McCarthy, Dimitrios Kanoulas, and Zhibin Alex Li. Your value function is a control barrier function: Verification of learned policies using control theory. *arXiv preprint arXiv:2306.04026*, 2023.

[43] Yichuan Charlie Tang, Jian Zhang, and Ruslan Salakhutdinov. Worst Cases Policy Gradients. In *Proceedings of the Conference on Robot Learning*, volume 100, pages 1078–1093. PMLR, 2020.

[44] Andrew Taylor, Andrew Singletary, Yisong Yue, and Aaron Ames. Learning for safety-critical control with control barrier functions. In *Learning for Dynamics and Control*, pages 708–717. PMLR, 2020.

[45] Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward Constrained Policy Optimization. In *International Conference on Learning Representations (ICLR)*, 2019.

[46] Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, L. Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander Sasha Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom Le Paine, Caglar Gulcehre, Ziyun Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy P. Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575:350 – 354, 2019. URL `https://api.semanticscholar.org/CorpusID:204972004`.

[47] Nolan C Wagener, Byron Boots, and Ching-An Cheng. Safe reinforcement learning using advantage-based intervention. In *International Conference on Machine Learning*, pages 10630–10640. PMLR, 2021.

[48] Qisong Yang, Thiago D Simão, Simon H Tindemans, and Matthijs TJ Spaan. Wcsac: Worst-case soft actor critic for safety-constrained reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35 (12), pages 10639–10646, 2021.

[49] Qisong Yang, Thiago D Simão, Simon H Tindemans, and Matthijs TJ Spaan. Safety-constrained reinforcement learning with a distributional safety critic. *Machine Learning*, 112(3):859–887, 2023.

[50] Yujie Yang, Zhilong Zheng, and Shengbo Eben Li. Feasible policy iteration. *arXiv preprint arXiv:2304.08845*, 2023.

[51] Chengyang Ying, Xinning Zhou, Hang Su, Dong Yan, Ning Chen, and Jun Zhu. Towards safe reinforcement learning via constraining conditional value-at-risk. In *International Joint Conference on Artificial Intelligence*, 2022.

[52] Ming Yu, Zhuoran Yang, Mladen Kolar, and Zhaoran Wang. Convergent policy optimization for safe reinforcement learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[53] Yinan Zheng, Jianxiong Li, Dongjie Yu, Yujie Yang, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Safe offline reinforcement learning with feasibility-guided diffusion model. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=j5JvZCaDM0`.

A. Hyperparameter Search

In this chapter, we report the parameter tuning for all the baselines in all tasks. In general, we test all the methods with different learning rates, cost budgets and safety parameters to ensure the performance of the baseline is optimal. We report all the hyperparameter configurations we tried and indicate which configuration is used for the main evaluation.

Every algorithm is first evaluated with the learning rates of $1e^{-4}$, $5e^{-4}$ and $1e^{-3}$. To keep the computation reasonable, we use the same learning rate for the actor, the critic, the constraints, and the learning rates for the Lagrangian multiplier that are updated every step. We report the results of these experiments for each task in the following sections.

As a second step, we experimented with different cost budgets to get the best trade-off between safety and performance. Our goal is to get the least constraint violations possible while maintaining reasonable behavior. As we show in Section 4.2.1, setting the cost budget too low can have an impact on the performance with no safety benefit.

Lastly, we tuned the cost-dampening parameters of LagSAC and WCSAC using the same principle we used for the cost budget.

A.1. CartPole

Figure A.1 shows the results of the learning rate tuning for the CartPole task. We can see that RCPO and LagSAC have a learning rate that achieves the best performance. For PPOLag and WCSAC, the differences are more nuanced. Table A.1 shows all the parameters we tried for the Cartpole task. The resulting best parameters used for the main evaluation can be found in Table A.2.

	RCPO	PPOLag	LagSAC	WCSAC	D-ATACOM
Sweeping parameter					
learning rate actor/critic/constraint			$\{1e^{-3}, 5e^{-4}, 1e^{-4}\}$		
cost budget	5	5		{0.1, 5, 25, 40}	
cost dampening	-	-	{1, 10}		-
learning rate lagrangien multipliers	0.035	0.035		$\{1e^{-4}, 5e^{-4}, 1e^{-4}\}$	
accepted risk	-	-	-	{0.1, 0.5, 0.9}	
Default parameter					
epochs	100	100	100	100	100
steps per epoch	20000	20000	10000	10000	10000
steps per fit	20000	20000	1	1	1
episodes per test	-	-	25	25	25
network size			[128 128]		
batch size	128	64	64	64	64
initial replay size	-	-	2000	2000	2000
max replay size	200000	200000	200000	200000	200000
soft update coefficient	-	-	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
warm-up transitions	-	-	2000	2000	2000
target kl	0.01	0.02	-	-	-
update iterations	10	40	-	-	-

Table A.1.: Training Parameters for the CartPole task

	RCPO	PPOLag	LagSAC	WCSAC	D-ATACOM
Sweeping parameter					
learning rate actor/critic/constraint	$5e^{-4}$	$1e^{-4}$	$5e^{-4}$	$5e^{-4}$	$5e^{-4}$
cost budget	5	5	5	5	40
cost dampening	-	-	1	1	-
learning rate lagrangian multipliers	0.035	0.035	$5e^{-4}$	$5e^{-4}$	$5e^{-4}$
accepted risk	-	-	-	0.9	0.9

Table A.2.: Result of hyperparameter tuning for the CartPole task

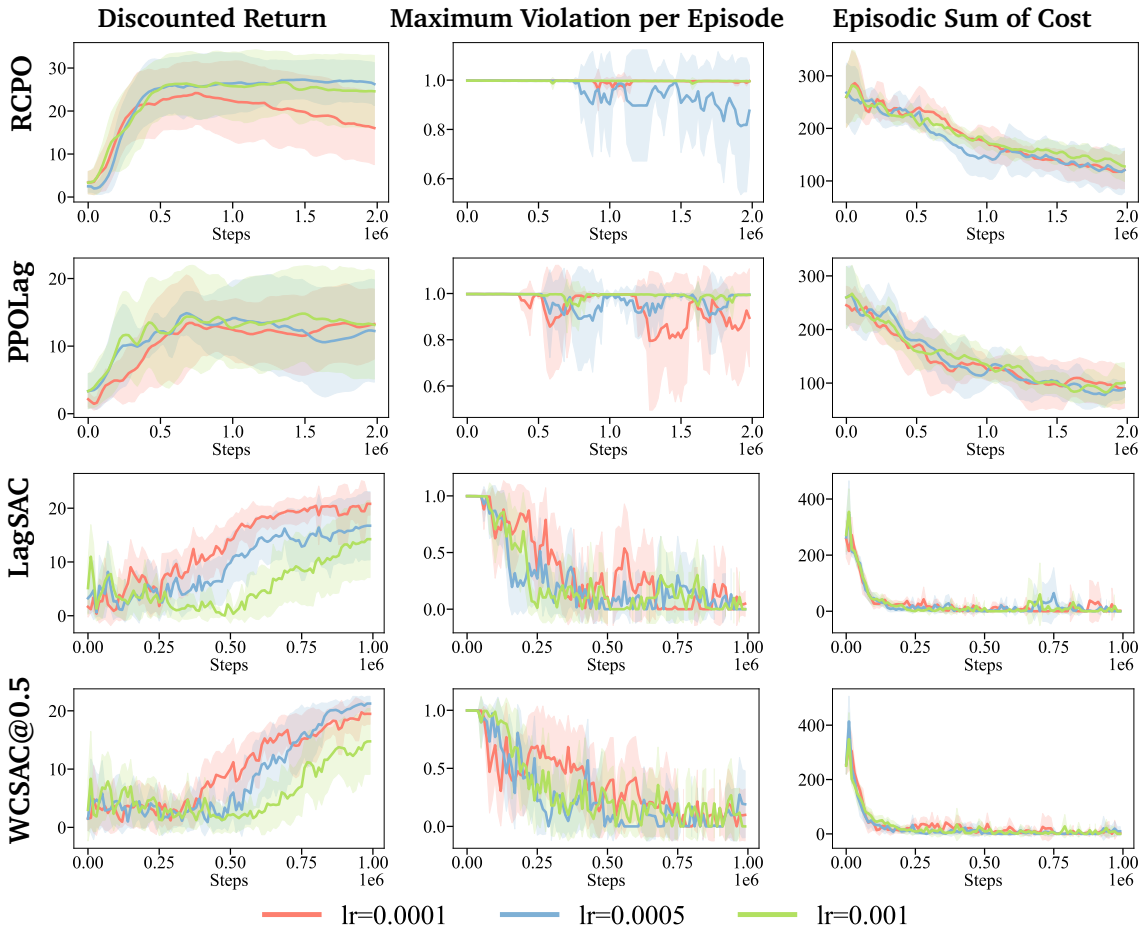


Figure A.1.: Learning rate ablation study for the Cartpole task. For each experiment we run 10 seeds with all learning rates of the algorithm set to the respective value.

A.2. Navigation

Figure A.2 shows the results of the learning rate tuning for the navigation task. We can see WCSAC is the only algorithm where the learning rate has a significant impact on the performance. Table A.3 shows all the parameters we tested for the navigation task. The resulting best parameters used for the main evaluation can be found in Table A.4.

	RCPO	PPOLag	LagSAC	WCSAC	D-ATACOM
Sweeping parameter					
learning rate actor/critic/constraint			$\{1e^{-3}, 5e^{-4}, 1e^{-4}\}$		
cost budget	0	0		{0, 1}	
cost dampening	-	-	{1, 10}		-
learning rate lagrangian multipliers	0.035	0.035		$\{1e^{-4}, 5e^{-4}, 1e^{-4}\}$	
accepted risk	-	-	-	{0.1, 0.5, 0.9}	
Default parameter					
epochs	100	100	100	100	100
steps per epoch	20000	20000	10000	10000	10000
steps per fit	20000	20000	1	1	1
episodes per test	-	-	25	25	25
network size			[128 128]		
batch size	128	64	64	64	64
initial replay size	-	-	2000	2000	2000
max replay size	200000	200000	200000	200000	200000
soft update coefficient	-	-	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
warm-up transitions	-	-	2000	2000	2000
target kl	0.01	0.02	-	-	-
update iterations	10	40	-	-	-

Table A.3.: Training Parameters for the navigation task

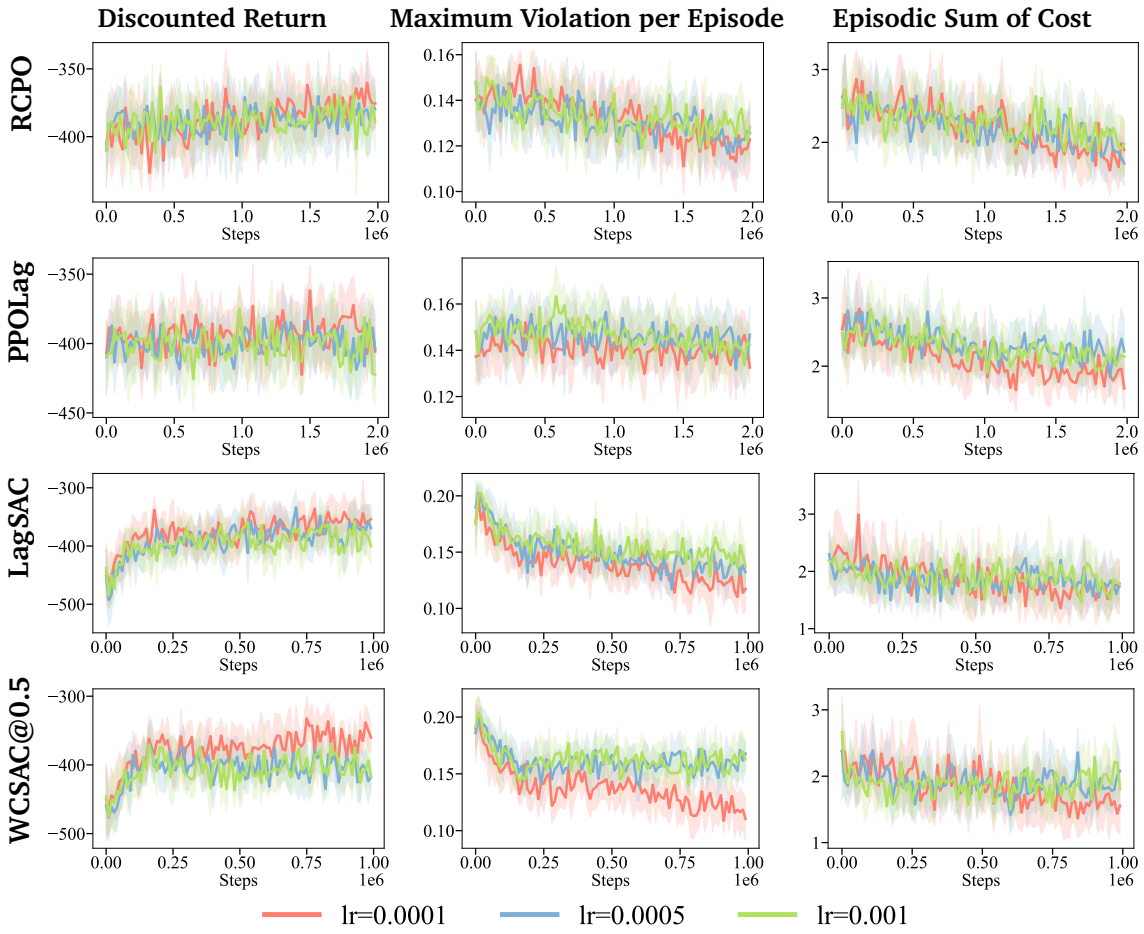


Figure A.2.: Learning rate ablation study for the Navigation task. For each experiment, we run 10 seeds with all learning rates of the algorithm set to the respective value.

	RCPO	PPOLag	LagSAC	WCSAC	D-ATACOM
Sweeping parameter					
learning rate actor/critic/constraint	$1e^{-4}$	$1e^{-4}$	$1e^{-4}$	$1e^{-4}$	$1e^{-4}$
cost budget	0	0	0	0	0
cost dampening	-	-	1	1	-
learning rate lagrangian multipliers	0.035	0.035	$1e^{-4}$	$1e^{-4}$	$1e^{-4}$
accepted risk	-	-	-	0.9	0.5

Table A.4.: Result of hyperparameter tuning for the navigation task

A.3. Air Hockey

Figure A.3 shows the results of the learning rate tuning for the air hockey task. We can see that RCPO and PPOLag learn safer behaviors compared to LagSAC and WCSAC. However, their discounted return is lower, and they need twice as many steps. Table A.5 shows all the parameters we tested for the air hockey task. The resulting parameters used for the main evaluation can be found in Table A.6.

	RCPO	PPOLag	LagSAC	WCSAC	D-ATACOM
Sweeping parameter					
learning rate actor/critic/constraint			$\{1e^{-3}, 5e^{-4}, 1e^{-4}\}$		
cost budget	0	0		{0, 1}	
cost dampening	-	-	{1, 10}		-
learning rate lagrangian multipliers	0.035	0.035		$\{1e^{-4}, 5e^{-4}, 1e^{-4}\}$	
accepted risk	-	-	-	{0.1, 0.5, 0.9}	
Default parameter					
epochs	100	100	100	100	100
steps per epoch	20000	20000	10000	10000	10000
steps per fit	20000	20000	1	1	1
episodes per test	-	-	25	25	25
network size			[128 128]		
batch size	128	64	64	64	64
initial replay size	-	-	2000	2000	2000
max replay size	200000	200000	200000	200000	200000
soft update coefficient	-	-	$1e^{-3}$	$1e^{-3}$	$1e^{-3}$
warm-up transitions	-	-	2000	2000	2000
target kl	0.01	0.02	-	-	-
update iterations	10	40	-	-	-

Table A.5.: Training Parameters for the air hockey task

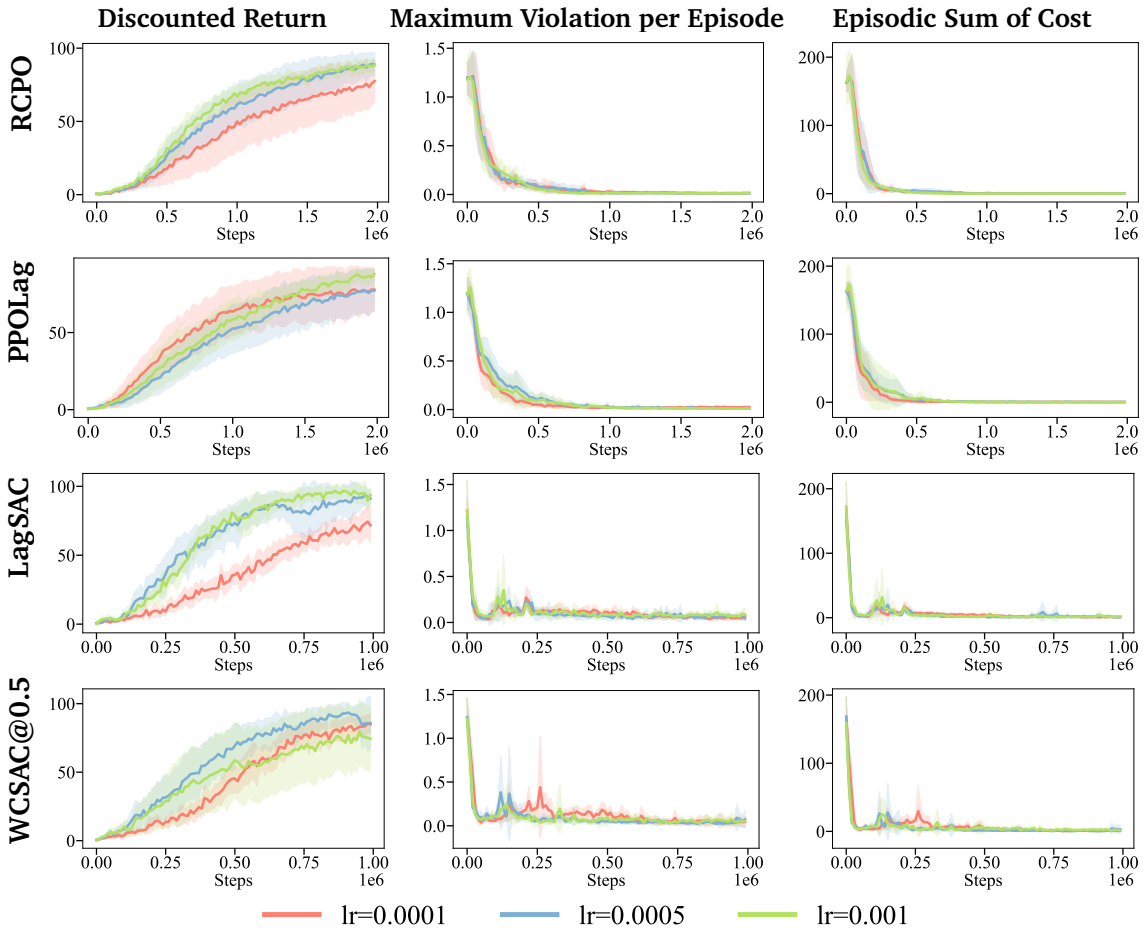


Figure A.3.: Learning rate ablation study for the Air Hockey task. For each experiment, we run 10 seeds with all learning rates of the algorithm set to the respective value.

	RCPO	PPOLag	LagSAC	WCSAC	D-ATACOM
Sweeping parameter					
learning rate actor/critic/constraint	$5e^{-4}$	$1e^{-3}$	$5e^{-4}$	$5e^{-4}$	$5e^{-4}$
cost budget	0	0	0	0	1
cost dampening	-	-	1	1	-
learning rate lagrangian multipliers	0.035	0.035	$5e^{-4}$	$5e^{-4}$	$5e^{-4}$
accepted risk	-	-	-	0.9	0.9

Table A.6.: Result of hyperparameter tuning for the air hockey task