Human in the Loop, Safe Reinforcement Learning for Continuous Control

A Thesis submitted by

Chetan Reddy N - ME19B093

Guide: Prof. Nirav Bhatt

IDDD - Data Science

Indian Institute of Technology Madras

June 2024

THESIS CERTIFICATE

This is to certify that the thesis titled **Human in the Loop, Safe Reinforcement Learning for Continuous Control**, submitted by **Chetan Reddy N**, to the Indian Institute of Technology, Madras, for the award of the degree of **Interdisciplinary Dual Degree in Data Science**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Nirav Bhatt

Wadhwani School of Data Science and AI

Department of Biotechnology

Indian Institute of Technology Madras

Chennai – 600036

Place: Chennai

Date: June 14, 2024

Acknowledgments

I am profoundly grateful to my supervisor, **Prof. Nirav Bhatt**, for their unwavering guidance, support, and encouragement throughout the entire process of this thesis. His expertise, patience, and invaluable feedback have been instrumental in shaping this work.

I am grateful to the **Wadhwani School of Data Science and AI** for their inaugural Annual Research Showcase, which provided a wonderful platform to showcase our research work. I am honored to have been **awarded 3rd place for the Best Poster**, along with a cash prize to support our research efforts.

I extend my gratitude to **Arvind Ragghav**, project associate, whose expertise and dedication were instrumental in developing the formulations essential to this research.

I would like to express my appreciation to my family and friends for their support and encouragement throughout. Special thanks to my friends in IDDD-DS. The casual discussions we had about our projects proved invaluable, providing clarity during moments of doubt and contributing significantly to the smooth progression.

Abstract

The ability to interact and explore an environment has enabled Reinforcement Learning agents to generate incredible results in many complex tasks. However, most of these results have been limited to simulated environments. One of the biggest obstacles to moving from simulated environments to real-world applications is safety. RL agents learn new tasks in uncertain environments through extensive exploration and trial and error, which can sometimes violate safety constraints and result in undesirable outcomes. Safe RL encapsulates algorithms that deal with this tradeoff between exploration and safety. In this study, we define a human-in-the-loop framework that ensures safety in both training and deployment.

Our approach involves two steps: the first step is the utilization of expert human input to establish a Safe State Space (SSS) and a corresponding Conservative Safe Policy (CSP). In simple environments, the human directly provides SSS and CSP while in complex environments, safe trajectories generated by the human are used to estimate SSS and CSP. In the second step, we use a modified version of the Deep Deterministic Policy Gradient algorithm augmented with a safety layer (built using SSS and CSP) that is learned by the agent during the training. We focus on continuous control environments which have a wide range of applications in robotics and autonomous systems.

Contents

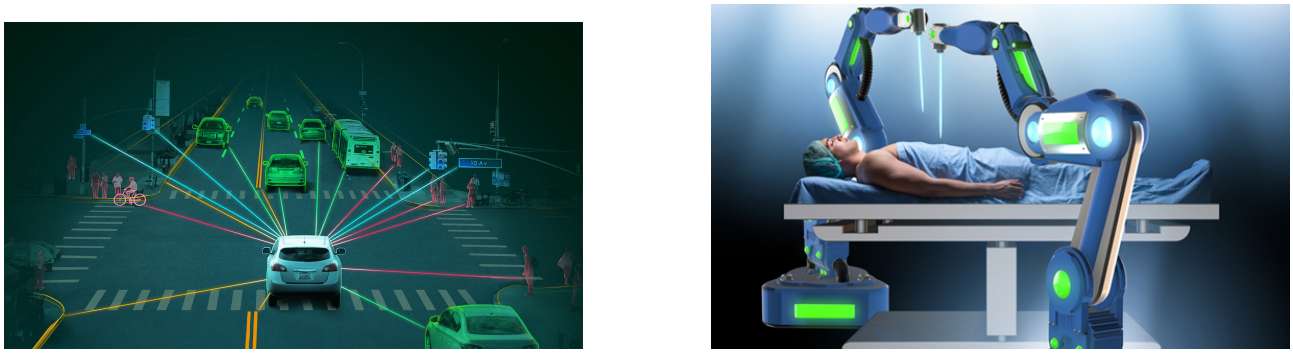
List of Figures

List of Tables

1 Introduction

Industries across the world are adopting Artificial Intelligence and Autonomy at an unprecedented rate. There has been a growing interest in applying learning-based methods, notably Reinforcement Learning (RL), in Autonomous Systems. The ability to interact and explore an environment has enabled Reinforcement Learning agents to generate incredible results in fields like Gaming [1][2]. It has also shown great success in learning control strategies from data and live interactions [3] [4].

In real life applications, apart from learning an optimal policy, the agent needs to strictly comply with safety constraints. For example, when a controller is being developed for a legged robot using RL, we cannot afford to take unsafe actions which can damage the robot. This leads to a tradeoff between exploration in unknown state space and safety. The field of Safe reinforcement learning tackles this challenge.



(a) Autonomous Driving (b) Healthcare Robotics

Figure 1: Applications of Safe RL

As humans entrust autonomous agents with increasingly complex tasks, their involvement in the learning process becomes crucial. This motivated us to develop and formalise a human-in-the-loop safe reinforcement learning framework in which the human specifies the initial safety constraints in the state and action space. This framework acts as a safety net for the RL agent when encountering

highly unfamiliar state spaces. In addition to having safety during training, the RL agent also learns the safety net provided by the human and improves on it during deployment. Our approach particularly focuses on continuous control problems under the assumption that the human can provide a safe state space and an associated safe action space.

The structure of the report is as follows: Section 2 gives a comprehensive study of the existing work in safe reinforcement learning and human-in-the-loop reinforcement learning. Section 3 formulates the problem and introduces some definitions. Our algorithm is described in Section 4. The details of the environment and the results are discussed in Section 5. Section 6 concludes the report and talks about the future work.

2 Literature Review

A comprehensive literature survey was performed at the intersection of safe reinforcement learning and human-in-the-loop reinforcement learning. This is an active research area within the broader field of Safe Artificial Intelligence. Tables 1 and 2 describe a few relevant papers.

2.1 Reinforcement Learning Fundamentals

Reinforcement learning is one of the three paradigms of machine learning, the other two being supervised learning and unsupervised learning. It involves learning to make sequential decisions in an environment to maximize cumulative rewards. It is widely used in various fields, including robotics, finance, supply chain management, gaming and autonomous systems. Fundamentally, reinforcement learning involves an agent interacting with an environment, learning from these interactions, and refining its decision-making process over time.

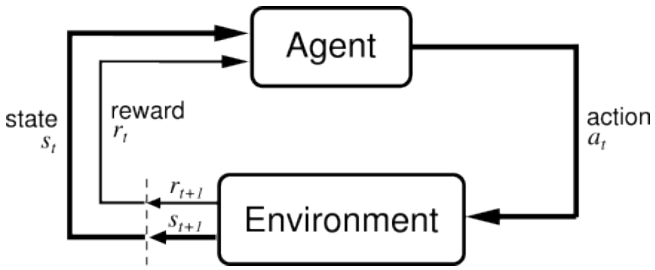


Figure 2: Agent Environment Interaction in RL Frameworks

2.1.1 Markov Decision Processes (MDP)

A Markov Decision Process (MDP) is a mathematical framework for modeling decision-making problems in which an agent interacts with an environment over a series of discrete time steps [5] [6]. It is defined by $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$ consisting of the following components:

- **State Space ($\mathcal{S}$):** The set of all possible states the environment can be in.

- **Action Space ($\mathcal{A}$):** The set of all possible actions the agent can take.
- **Transition Probability Function (P):** The function that specifies the probability of transitioning from one state to another state given an action.
- **Reward Function (R):** The function that specifies the immediate reward the agent receives after taking an action in a particular state.
- **Discount Factor (γ):** A parameter that determines the importance of future rewards relative to immediate rewards.

2.1.2 Constrained Markov Decision Processes (CMDP)

In some scenarios, there may be additional constraints that the agent must satisfy while maximizing the expected cumulative reward [7]. This leads to the formulation of Constrained Markov Decision Processes (CMDP) which is the basis for most Safe Reinforcement Learning Algorithms. In a CMDP, along with the traditional MDP components, we have:

- **Constraint Function (C):** A function that maps states to a cost. Safe states will have zero cost.
- **Constraint Threshold (β):** A threshold value that represents the maximum allowable constraint violation.

The objective function $J(\pi)$ for a CMDP can be formulated as:

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\ \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, \pi\right]$$

$$\text{s.t } \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t C(s_t) \,\middle|\, \pi\right] \leq \beta$$

Here:

- $J(\pi)$ represents the expected cumulative reward under policy π.
- $R(s_t, a_t)$ is the immediate reward obtained after taking action a_t in state s_t.
- $C(s_t)$ is the cost associated with state s_t.
- γ is the discount factor determining the importance of future rewards.
- β is the constraint threshold.

The goal is to find the optimal policy π^* that maximizes $J(\pi)$ while satisfying the constraint. This involves finding the policy that achieves the highest expected cumulative reward while keeping the expected cumulative constraint violation below β.

2.2 Safe Reinforcement Learning

As AI systems become increasingly integrated into various aspects of society, ensuring their safe and reliable operation has become paramount [8]. There has been enough research in Safe Reinforcement Learning over the past decade, with existing work broadly categorized into two main areas: the optimization criteria and the exploration process. [9]

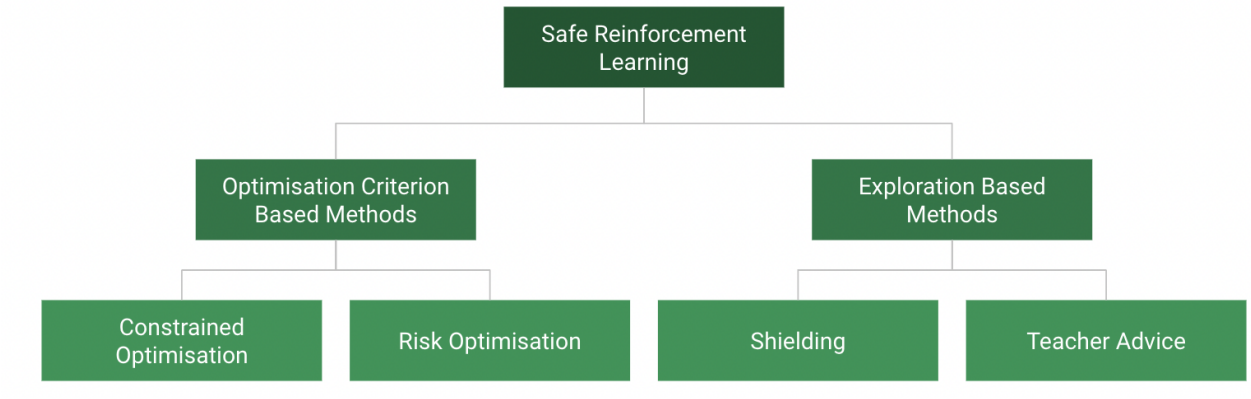


Figure 3: Safe RL Areas

2.2.1 Optimisation Criterion Based Methods

The first area in safe RL involves the integration of safety constraints directly into the optimization process. Constrained policy optimization (CPO) [10, 11] is a policy search algorithm for safe reinforcement learning that guarantees near-constraint satisfaction at each iteration.

Risk optimization in reinforcement learning (RL) [12] often involves managing uncertainty and minimizing the impact of worst-case scenarios. Conditional Value at Risk (CVaR) is a risk measure used in finance and other fields to quantify the potential losses beyond a certain threshold. In RL, CVaR can be employed to address robustness concerns and enhance the stability of learned policies in uncertain environments.

2.2.2 Exploration Based Methods

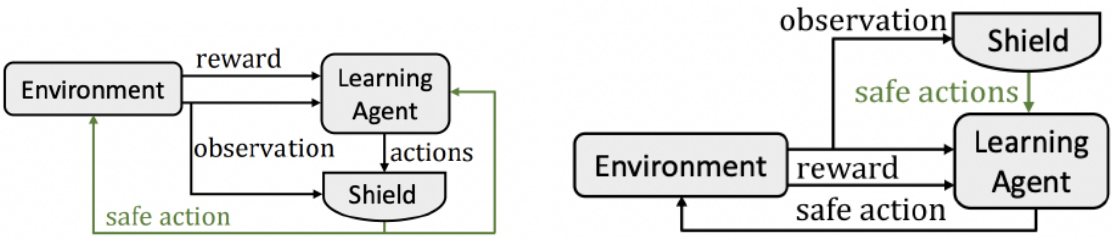
The second approach involves modifying the exploration process to prevent the selection of exploratory actions that may steer the learning system toward undesirable or catastrophic outcomes. The challenge lies in exploring the environment to learn while minimizing the risk of violating safety constraints.

Shielding-based methods [13, 14] impose restrictions on the action space. There are two types of shielding, as shown in Fig.4. An advantage of shielding based methods is that it separates the two objectives, the learning agent can focus on maximizing the reward, while the shield can focus on enforcing safety constraints. Secondly, it enforces safety while speeding up the learning.

External knowledge in the form of 'Teacher Advice' can be incorporated to ensure safety [15, 16, 15]. The teacher can be a human or a simple controller. Both the agent and the teacher can initiate this interaction during the learning process. This falls within the scope of human in the loop reinforcement learning which is discussed in more detail in the next section 2.3.

Paper	Relevance	Evaluation Environments	Safe RL Area
Constrained Policy Optimisation [10]	Safety Constraints obeyed during training	Humanoid Circle and Point Gather	Optimisation Criterion
Towards safe reinforcement learning via constraining conditional value-at-risk [12]	constrained optimization problem by keeping its CVaR under a given threshold	MuJoCo Environments (Half Cheetah)	Optimisation Criterion
Safe reinforcement learning in constrained markov decision processes [11]	Learns safety constraints by understanding the safe region	GP-Safety Gym Mars Surface Exploration	Optimisation Criterion
Safe model-based reinforcement learning [17]	Treats stability as safety	Inverted Pendulum [18]	Optimisation Criterion
End-to-end safe reinforcement learning using CBF [19]	Safety Layer using CBFs [20]	Inverted Pendulum Autonomous Cars	Optimisation Criterion
Safe exploration in continuous action spaces [21]	DDPG with Safety Layer	MuJoCo Domains: Ball and Spaceship	Exploration
A closer look at invalid action masking in policy gradient algorithms [22]	Gradient Changes to learn Safety Layer	Gym-μRTS [23]	Exploration
Conservative safety critics for exploration [24]	Learns a Critic Network to say how safe an action is	2D point navigation, Car Navigation	Exploration
Safe exploration of state and action spaces in reinforcement learning [25]	Robust behaviors for continuous control tasks	Automatic car parking, Pole-balancing, Helicopter Hovering	Exploration
Safe reinforcement learning by imagining the near future [26]	Model based algorithm that can forecast and avoid unsafe states	Safety Gym (Half Cheetah)	Exploration
Safe reinforcement learning using advantage-based intervention [27]	Backup policy and an Intervention based learning	Point Robot, Half Cheetah	Exploration
Safe reinforcement learning via shielding [13]	Safety constraints expressed as temporal logic specifications	Grid World, PacMan, Self Driving Car	Exploration

Table 1: Description of some Safe RL Papers



(a) Post-posed Shielding (b) Preemptive Shielding

Figure 4: Shielding Methods

2.3 Human in the Loop Reinforcement Learning

Human-in-the-loop reinforcement learning (HILRL) integrates human expertise into the RL process, enhancing learning efficiency and interpretability. Humans interact with the RL agent, providing feedback and guidance, influencing its behavior. This interaction occurs at various stages, including policy design, reward specification, and evaluation. It represents a promising approach for real-world applications, where human oversight is crucial for ensuring safety and ethical behavior. HILRL can broadly be from the following areas:

- Preference-Based Methods
- Imitation Learning

2.3.1 Human preference-based RL

Human preference-based RL [15, 28], also known as preference-based or interactive RL, leverages human feedback to guide the learning process. Instead of explicitly defining a reward function, which can be challenging and time-consuming in many real-world applications, human preferences offer a more intuitive and flexible way to specify desirable behaviors. Human preferences can range from simple binary feedback (e.g., "better" or "worse") to more nuanced rankings or comparisons between different actions or trajectories.

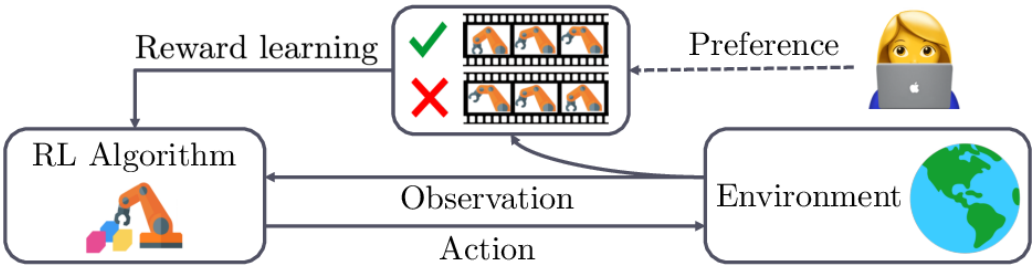


Figure 5: Human Preference Based RL

2.3.2 Imitation Learning

Imitation learning [29, 30, 31, 32, 33], is a prominent approach in reinforcement learning (RL) that leverages expert demonstrations to accelerate the learning process. In imitation learning, the RL agent observes a set of expert demonstrations, typically in the form of state-action pairs, and learns to mimic the behavior of the expert by mapping states to actions. One challenge in imitation learning is the distributional mismatch between the demonstrations provided by the expert and the states encountered during the agent's learning process, which can lead to suboptimal or even unsafe policies. Multiple approaches have been developed to solve this covariance shift problem like inducing adversarial disturbance during training [34] and iteratively collecting human feedback [31].

2.4 Research Questions

There has been some work at the intersection of safe RL and human-in-the-loop RL, but to the best of our knowledge, there is no explicit use of human inputs to drive the safety violations close to zero. This thesis aims to answer the following research questions:

- How can reinforcement learning algorithms adapt and learn from human inputs and safety constraints during continuous control tasks?
- How does the agent learn and improve on the safety inputs provided by the human to ensure close to 100% safety in both training and deployment?

Paper	Relevance	Evaluation Environments	Area
A survey of preference-based reinforcement learning methods [35]	Using expert's preferences instead of a hand-designed numeric reward		Preference Based
Trial without Error: Towards Safe Reinforcement Learning via Human Intervention [15]	Presence of Human/Supervised Imitator during training to ensure 100% safety	Atari Games	Preference Based
Imitate the good and avoid the bad: An incremental approach to safe reinforcement learning [36]	Human input is used to learn both safe and unsafe actions	Safety Gym	Imitation
Deep reinforcement learning from human preferences [37]	Learns a reward function using human preferences	Atari Games	Preference Based
Parenting: Safe reinforcement learning from human input [16]	Humans safely teach the RL agents which upon maturation outperforms	AI Safety Gridworlds [38]	Preference Based
Few-shot preference learning for human-in-the-loop rl [28]	Aims to learn reward functions with as little human feedback as possible	MetaWorld Door-Close and DM Control Reacher environments [39]	Preference Based
Ensembledagger: A bayesian approach to safe imitation learning	Uncertainty estimate is used to determine if the novice agent or expert can take the action	Inverted Pendulum, Half Cheetah	Imitation
Generative adversarial imitation learning	Directly extracts a policy from expert data	Gym Environments (Cartpole, Acrobot, Half Cheetah etc)	Imitation

Table 2: Description of some relevant Human in the Loop RL Papers

3 Problem Formulation

One of the biggest challenges is to avoid unsafe states while performing exploration. On one hand, exploration is required for the agent to learn while on the other hand exploration can lead the agent to visit unsafe states. There are two ways to approach this challenge. One is to allow the agent to visit the unsafe states during exploration but ensure that the number of safety violations is below a threshold as seen in the CMDP formulation. The second approach is to estimate the safe and unsafe states externally by using the system dynamics or human input.

The second method can be used in systems where its critical to maintain safety violations close to zero even during training. In the literature, there are several methods to ensure safety during both training and deployment. However, most of them merely require safety violations to be below a threshold rather than focusing on eliminating them completely. This study aims to achieve this by using human inputs in the form of conservative safe policy and safe state space defined below. It is called "conservative" because the action only ensures safety and is not concerned about the reward.

Consider a Markov Decision Process $(\mathcal{S}, \mathcal{A}, f, \mathcal{R}, \gamma)$ with a state space $\mathcal{S}$, action space $\mathcal{A}$, transition function f s.t $s' = f(s, a)$, reward structure $\mathcal{R}$ and discounting factor γ. A policy is defined as a mapping $\pi : \mathcal{X} \to \mathcal{A}$. Under such a policy, the trajectory of the system starting with initial condition x_0 is denoted by $\varphi(t, \pi, x_0)$.

3.1 Safe State Space (SSS) and Conservative Safe Policy (CSP)

To formalise the human input, we define two sets called the safe state space (SSS or $\mathcal{X}_s$) and the conservative safe policy (CSP). The set $\mathcal{X}_s$ is again composed of two sets called truly safe state space ($\mathcal{X}_{ts}$) and marginally safe state space

($\mathcal{X}_{ms}$). We define the terms below.

Definition 1 (Safe State Space $\mathcal{X}_s$)**.** *It is a subset of the state space $\mathcal{S}$ which is considered safe i.e $C(s) = 0\ \forall s \in \mathcal{X}_s$ where $C(s)$ is the cost function.*

Definition 2 (Truly Safe State Space $\mathcal{X}_{ts}$)**.** *It is a subset of $\mathcal{X}_s$ in which the agent will be safe irrespective of the action taken.*

$$\mathcal{X}_{ts} = \{s \in \mathcal{X}_s \mid s' \in \mathcal{X}_s\ \forall\ a \in \mathcal{A}\ and\ s' = f(s,a)\}$$

Definition 3 (Marginally Safe State Space $\mathcal{X}_{ms}$)**.** *The marginally safe set is region in $\mathcal{X}_s$ that is not in $\mathcal{X}_{ts}$ i.e the region close to the boundary of safety. $\mathcal{X}_{ms} = \mathcal{X}_s - \mathcal{X}_{ts}$. It is the collection of states from which the agent enters the truly safe set $\mathcal{X}_{ts}$, eventually under the CSP. That is, if $x_0 \in \mathcal{X}_{ms} \subseteq \mathbb{R}^n$, then for a time instant $t^* \geq 0$, the agent trajectory under the conservative safe policy lies in the truly safe set eventually, i.e., $\varphi(t, CSP, x_0) \in \mathcal{X}_{ts}, t \geq t^*$.*

Definition 4 (Conservative Safe Policy CSP)**.** *The conservative safe policy satisfies the definition of the marginally safe set. Additionally, it also leaves the safe state space forward invariant i.e*

$$\varphi(t, CSP, x_0) \in \mathcal{X}_{ts},\ \forall\ t \geq t^*(x_0)\ \forall\ x_0 \in \mathcal{X}_{ms} \tag{1}$$

During the agent's exploration of the environment as a part of training, it is possible that it leaves the truly safe set. However, by using the conservative safe policy, it can be pushed back inside. The condition 1 shows that if we use the CSP in the marginally safe state space, the agent always stays in the safe state space $\mathcal{X}_s$ thus ensuring safety throughout.

Assumption 1. *For every state in the marginally safe state space, there exists atleast one action provided by conservative safe policy i.e $CSP(s) \neq \varnothing\ \forall\ s \in \mathcal{X}_{ms}$*

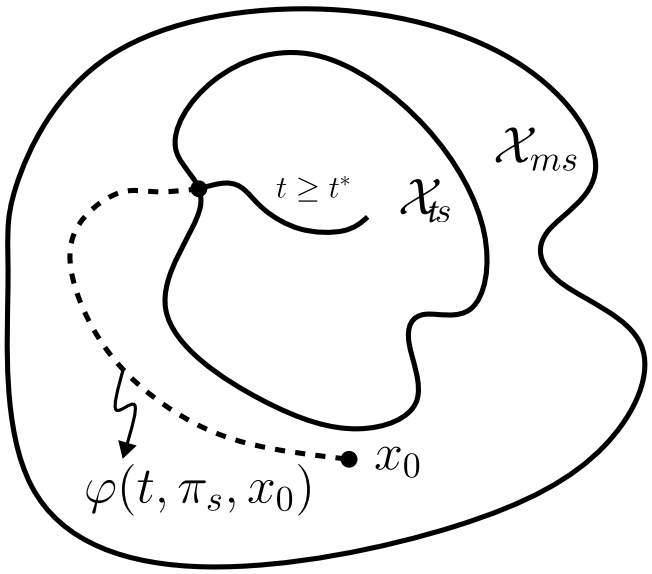


Figure 6: Depiction of the Safe Sets

3.2 Problem Definition

Given $(\mathcal{X}_{ts}, \mathcal{X}_{ms}, CSP)$ complying with assumption 1, we need to find a policy π^*, that maximises the expected reward while ensuring safety. The expected reward is defined as follows.

$$J(\pi) = \mathop{\mathbb{E}}_{x_0 \sim \mathcal{X}_s} \left(\int_0^\infty R(\varphi(t, \pi, x_0))dt \right) \tag{2}$$

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, \pi \right] \tag{3}$$

Equation 2 corresponds to continuous time domain while Equation 3 corresponds to discrete time steps. The problem can be condensed into the following optimisation equation:

$$\begin{aligned} \pi_* = \underset{\pi}{\text{argmax}} \;\; & J(\pi) \\ \text{s.t. } & \varphi(t, \pi, x_0) \in \mathcal{X}_s \;\forall\; t \geq 0 \;\forall\; x_0 \in \mathcal{X}_s \end{aligned}$$

The condition states that the trajectory must be within the safe state space throughout training. Using a vanilla reinforcement learning approach to find

the solutions to the above problem statement would only ensure that the reward is maximised due to which safety violations can occur during training. In many robotics applications where online reinforcement learning is used, ensuring safety during training is also critical. This problem of addressing safety during both the training and testing phase is the central focus of this thesis.

4 Methodology

There are three major aspects to our algorithm:

1. We develop a pipeline that ensures both safe training and deployment. Safe Training is ensured by using the CSP (Conservative Safe Policy).

2. A loss term in the actor policy loss ensures that the safety layer is also simultaneously learnt during the training process

3. The final policy that is obtained not only obeys the safety constraints but is expected to be better than the conservative safe policy with respect to the rewards gained.

4.1 Architecture

The Deep Deterministic Policy Gradient (DDPG) algorithm is a powerful actor-critic method designed for solving continuous action space reinforcement learning problems. It features two neural networks: an actor network for learning deterministic policies mapping states to actions, and a critic network estimating the Q-value of state-action pairs. DDPG uses experience replay and target networks to stabilize training.

We modify the DDPG architecture to incorporate human provided safety formulations. During the training phase, if the state is within $\mathcal{X}_{ts}$, the agent is allowed to explore and take any action according to μ_θ but if the state is in $\mathcal{X}_{ms}$, an action is taken in accordance to the conservative safe policy (definition 4) thus pushing it back to the safe region.

The conservative safe policy will output a safe action region that can be a singleton set or a range. The safe action is found by discretizing the safe action region at state s and choosing the one with the highest Q value given by the critic network(ϕ) as given below:

$$a = \underset{a' \in CSP(s)}{argmax}\ Q_\phi(s, a')$$

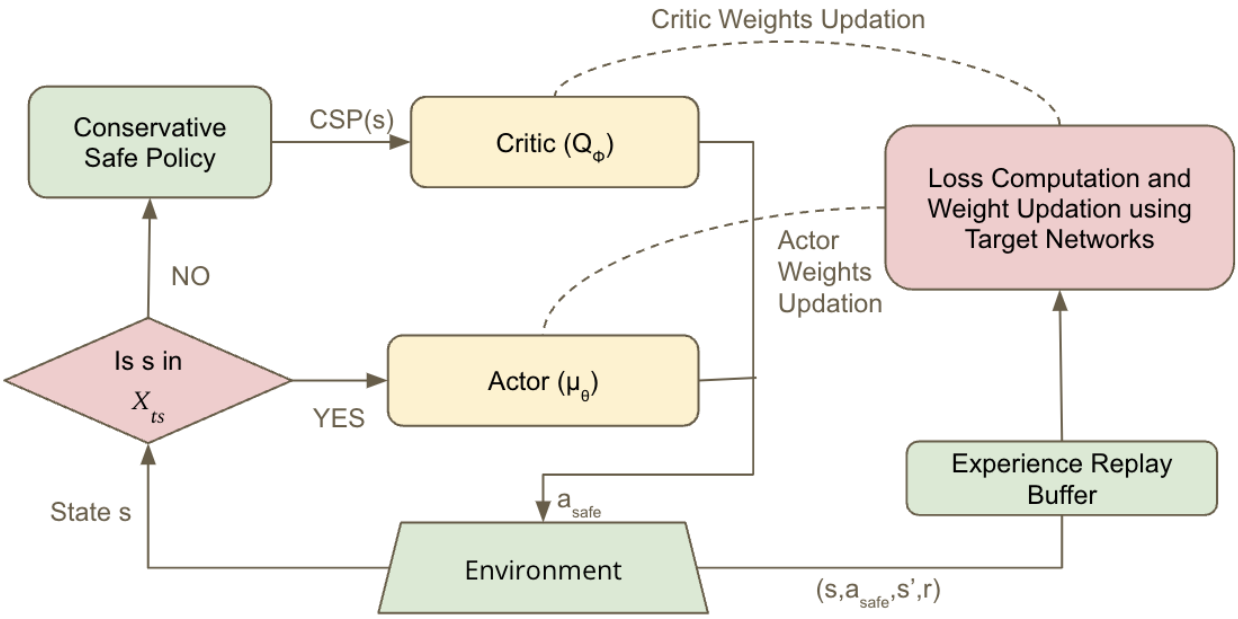


Figure 7: Model Pipeline

4.2 Critic and Actor Losses

The loss function for the critic network takes the form of the mean squared error between the predicted value and the target value. Mathematically, the critic loss $L(\phi)$ can be expressed as:

$$L(\phi) = \frac{1}{|B|} \sum_{(s,a,r,s',d) \in B} \left(Q_\phi(s,a) - y(r,s',d)\right)^2$$

where $\mathcal{B}$ represents the replay buffer containing a batch of experiences, $y(r, s', d) = r + \gamma(1-d)Q_{\phi_{\text{targ}}}(s', \mu_{\theta_{\text{targ}}}(s'))$ is the target value, r is the reward, γ is the discount factor, $\mu_{\theta targ}$ is the target policy, and $Q_{\phi_{\text{targ}}}$ is the target action-value function.

On the other hand, the actor network is trained to maximize the expected return by adjusting the policy parameters. The loss function for the actor

network involves maximizing the expected Q-value for the actions selected by the actor. Mathematically, the actor loss $L(\theta^{\mu})$ can be expressed as:

$$L_{actor}(\theta) = \nabla_{\theta} \frac{1}{|B|} \sum_{s \in B} Q_{\phi}(s, \mu_{\theta}(s))$$

Together, these loss functions guide the training process in DDPG, enabling the actor and critic networks to learn effective policies for continuous control tasks.

4.3 Policy Safe Action Loss $L_s(\theta)$

During testing, we expect the agent to take safe actions even without the safety layer. To ensure this, we added an extra loss term $L_s(\theta)$ in the actor loss. Total actor loss is given by

$$L_s(\theta) = (a - \mu_{\theta}(s))^2$$

$$L(\theta) = L_{actor}(\theta) + L_s(\theta)$$

If the state s is in $\mathcal{X}_s$, $L_s(\theta)$ would be zero but if the state is near the safe boundaries, the $L_s(\theta)$ term would penalise the actor for taking actions far from the conservative safe policy. This would ensure that the safety layer is learnt.

4.4 Risk Coefficient ϵ_r

In order to perform exploration better and avoid too conservative behaviour, we introduce the risk coefficient that defines the probability with which the agent can take any action. It is a hyperparamater incorporated into the π_{safe} function. When $\epsilon_s > 0$, we call it partially safe approach.

Algorithm 1 Deep Deterministic Policy Gradient with Safety Layer

Input: initial policy parameters θ, Q-function parameters ϕ, empty replay buffer $\mathcal{D}$
Set target parameters equal to main parameters $\theta_{\text{targ}} \leftarrow \theta$, $\phi_{\text{targ}} \leftarrow \phi$
repeat
  Observe state s and select action $a = \pi_{safe}(s)$, the function is defined below
  Execute a in the environment
  Observe next state s', reward r, and done signal d to indicate whether s' is terminal
  Store (s, a, r, s', d) in replay buffer $\mathcal{D}$
  If s' is terminal, reset environment state.
  if it's time to update **then**
    for however many updates **do**
      Randomly sample a batch of transitions, $B = \{(s, a, r, s', d)\}$ from $\mathcal{D}$
      Compute targets

$$y(r, s', d) = r + \gamma(1 - d)Q_{\phi_{\text{targ}}}(s', \mu_{\theta_{\text{targ}}}(s'))$$

      Update Q-function by one step of gradient descent using (Critic Update)

$$\nabla_\phi L(\phi) = \nabla_\phi \frac{1}{|B|} \sum_{(s,a,r,s',d) \in B} \left(Q_\phi(s, a) - y(r, s', d)\right)^2$$

      Update policy by one step of gradient ascent using (Actor Update)

$$\nabla_\theta L(\theta) = \nabla_\theta \frac{1}{|B|} \sum_{s \in B} Q_\phi(s, \mu_\theta(s)) + (a - \mu_\theta(s))^2$$

      Update target networks with

$$\phi_{\text{targ}} \leftarrow \rho\phi_{\text{targ}} + (1 - \rho)\phi$$
$$\theta_{\text{targ}} \leftarrow \rho\theta_{\text{targ}} + (1 - \rho)\theta$$

    end for
  end if
until convergence
Function $\pi_{safe}(s)$:
  if $s \in \mathcal{X}_s$:
    $a = \text{clip}(\mu_\theta(s) + \epsilon, a_{Low}, a_{High})$, where $\epsilon \sim \mathcal{N}$
  else:
    r = generateRandomNumber(0,1)
    if $r < \epsilon_r$:
      $a = \text{clip}(\mu_\theta(s) + \epsilon, a_{Low}, a_{High})$, where $\epsilon \sim \mathcal{N}$
    else:
      $a = argmax_{a' \in CSP(s)} Q_\phi(s, a')$
return a

5 Results

Experiments were performed in continous control environments from the Safety-Gymnasium Library [40] and the OpenAI Gym's inverted pendulum environment. The details about the environment and the results are described below.

5.1 Inverted Pendulum

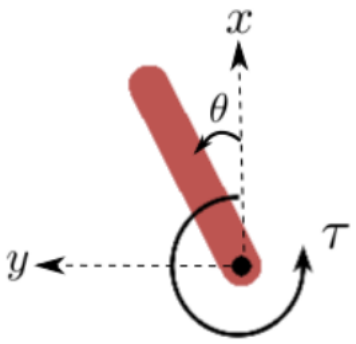


Figure 8: Inverted Pendulum Environment

The inverted pendulum is an ideal testbed for safe reinforcement learning algorithms. It mirrors real-world scenarios like segways and biped robots, where safety is of paramount importance in both training and testing.

5.1.1 SSS and CSP

The truly safe set, marginally safe set, and conservative safe policy according to Definitions 2, 3 and 4 for the inverted pendulum are given below. A simpler heuristic based CSP was proposed earlier in which the torque generated was dependent on angle alone. However, this violated assumption 1. Thus, a CSP dependent on both angle and angular velocity was developed. The precise CSP and SSS are obtained by using the pendulum model equations.

$$f_{\theta_0}(\theta, \dot{\theta}) = \begin{cases} \dot{\theta} - g_{\theta_0}(\theta) \text{ if } \theta \geq 0 \text{ and } \dot{\theta} \geq 0 \\ -\dot{\theta} - g_{\theta_0}(\theta) \text{ if } \theta \geq 0 \text{ and } \dot{\theta} < 0 \\ \dot{\theta} - g_{\theta_0}(-\theta) \text{ if } \theta < 0 \text{ and } \dot{\theta} \geq 0 \\ -\dot{\theta} - g_{\theta_0}(-\theta) \text{ if } \theta < 0 \text{ and } \dot{\theta} < 0 \end{cases} \tag{4}$$

where

$$g_{\theta_0}(\theta) = \sqrt{\frac{6}{ml^2}(T_m(\theta - \theta_0) + \frac{1}{2}mgl(cos(\theta_0) - cos(\theta)))} \tag{5}$$

$$\mathcal{X}_{ts} = \{(\theta, \dot{\theta}) \mid \forall\, f_{\theta_0}(\theta, \dot{\theta}) < 0) \mid \theta_0 = 10^\circ\} \tag{6}$$

$$\mathcal{X}_{s} = \{(\theta, \dot{\theta}) \mid \forall\, f(\theta, \dot{\theta}) < 0) \mid \theta_0 = 20^\circ\} \tag{7}$$

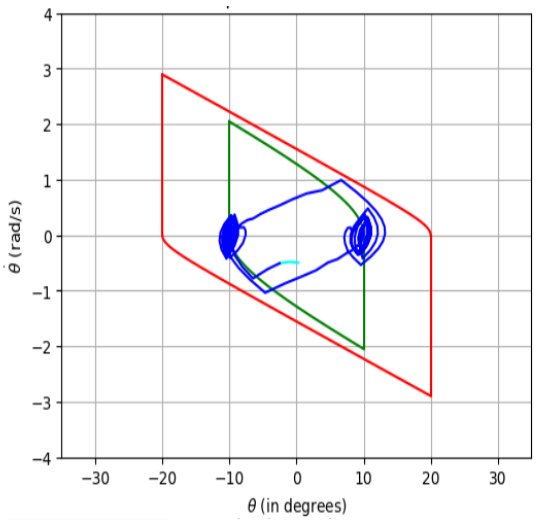
$$\mathcal{X}_{ms} = \mathcal{X}_s - \mathcal{X}_{ts} \tag{8}$$

$$CSP(\theta, \dot{\theta}) = \begin{cases} [-2, -1.7] \text{ if } \theta \geq 0 \text{ and } \dot{\theta} \geq 0 \\ [-0.1, 0.1] \text{ if } \theta \geq 0 \text{ and } \dot{\theta} < 0 \\ [-0.1, 0.1] \text{ if } \theta < 0 \text{ and } \dot{\theta} \geq 0 \\ [1.7, 2] \text{ if } \theta < 0 \text{ and } \dot{\theta} < 0 \end{cases} \tag{9}$$

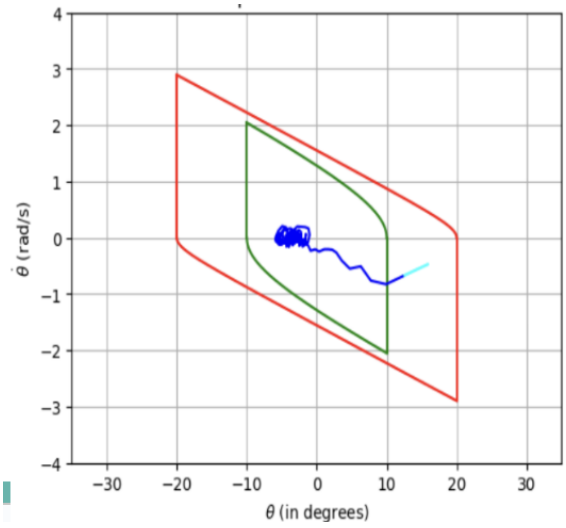
5.1.2 Training Curves and Metrics

The trajectories in the state space are tracked during training and deployment as seen in figure 9. The boundary of $\mathcal{X}_{ts}$ is depicted in green color while the region between the green and red lines is $\mathcal{X}_{ms}$. The light blue part of the trajectory is the start of the trajectory. We observe that during training, the agent tends to leave $\mathcal{X}_{ts}$ but is pushed back using the CSP. The states tracked near the corners show how the untrained agent tends to take random action and leaves the marginally safe limit of 10 degrees, the CSP is immediately activated which pushes it back. This creates oscillations at 10 and -10 degrees as seen in Fig 9a during training.

During Deployment, the agent has learnt to stay within the truly safe region. Even if the initialisation happens in $\mathcal{X}_{ms}$, the agent has learnt to enter $\mathcal{X}_{ts}$.

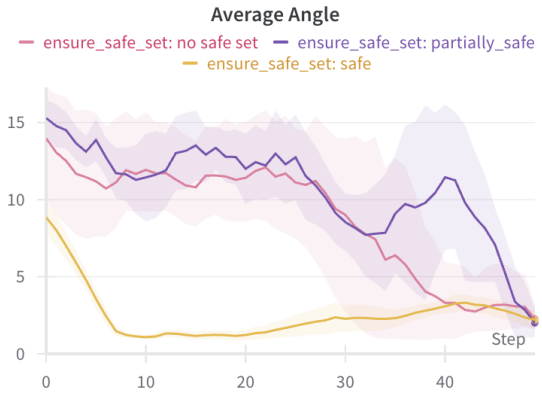


(a) An Episode during Training

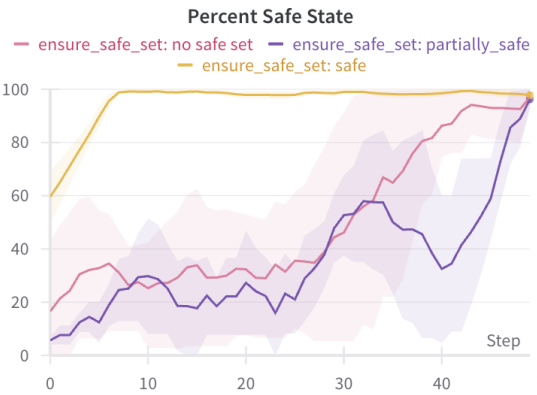


(b) An Episode during Deployment

Figure 9: Trajectories during Training and Deployment



(a) Average Angle (Measure of the Reward)



(b) Safety Percentage

Figure 10: Metrics for Implementations with and without safety layer

From figure 10, we see that our approach i.e SafeDDPG is clearly better with respect to safety and gives a comparable reward value (lower average angle is higher reward). We observe that the partially safe approach (ϵ_r) interestingly yields lesser rewards compared to the safe approach. This can be credited to a good input CSP.

Table 3 shows the metrics for our implementation and compares it with a vanilla DDPG algorithm during both training and deployment. To assess the

Algorithm	% Safe Episodes	% Safe Episodes under Disturbance	Average Angle
DDPG (Train)	74		6.19
DDPG (Test)	96	96	4.86
SafeDDPG (Train)	100		4.13
SafeDDPG (Test)	100	96	2.97

Table 3: Safety Metrics and Performance Comparision

robustness of the algorithm, we introduced adversarial disturbances. When the agent is in $\mathcal{X}_{ts}$, we randomly sampled an action that would push the agent towards the safety boundary and checked if it would recover. We observe that 96% of the episodes are safe even when there is disturbance. We achieve 100% safety during both training and deployment when there is no distrubance.

5.2 Safety Gymnasium Environment

The specific environment used is the 'SafetyPointCircle1'. The Agent needs to circle around the center of the circle area while avoiding going outside the boundaries (yellow walls). Every time the agent leaves the yellow wall, it is considered as a safety violation and a cost is incurred. On the other hand, maximum reward is obtained when the agent moves with maximum velocity along the green circle. This environment shows how rewards and safety can be conflicting unlike the pendulum environment.

5.2.1 SSS and CSP

The state space in the environment is 28 dimensional with 16 values representing Lidar distance from the centre. The other 12 correspond to accelerometer, velocimeter, gyroscope and magnetometer. The values from the lidar data and magnetometer were preprocessed to get polar coordinates with respect to the circle centre and consequently obtain cartesian coordinates of the agent at every instant. The truly safe state space in this environment is defined as a shrunk

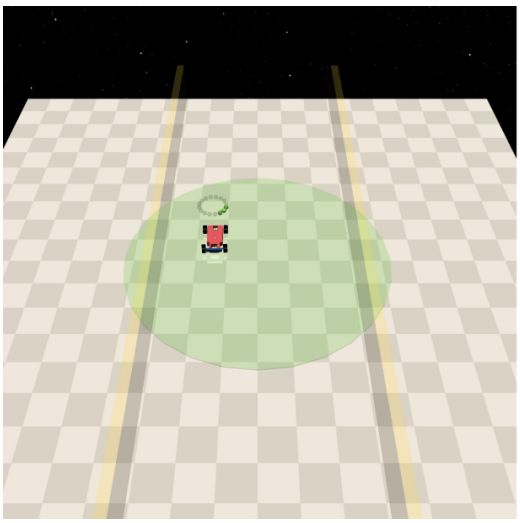


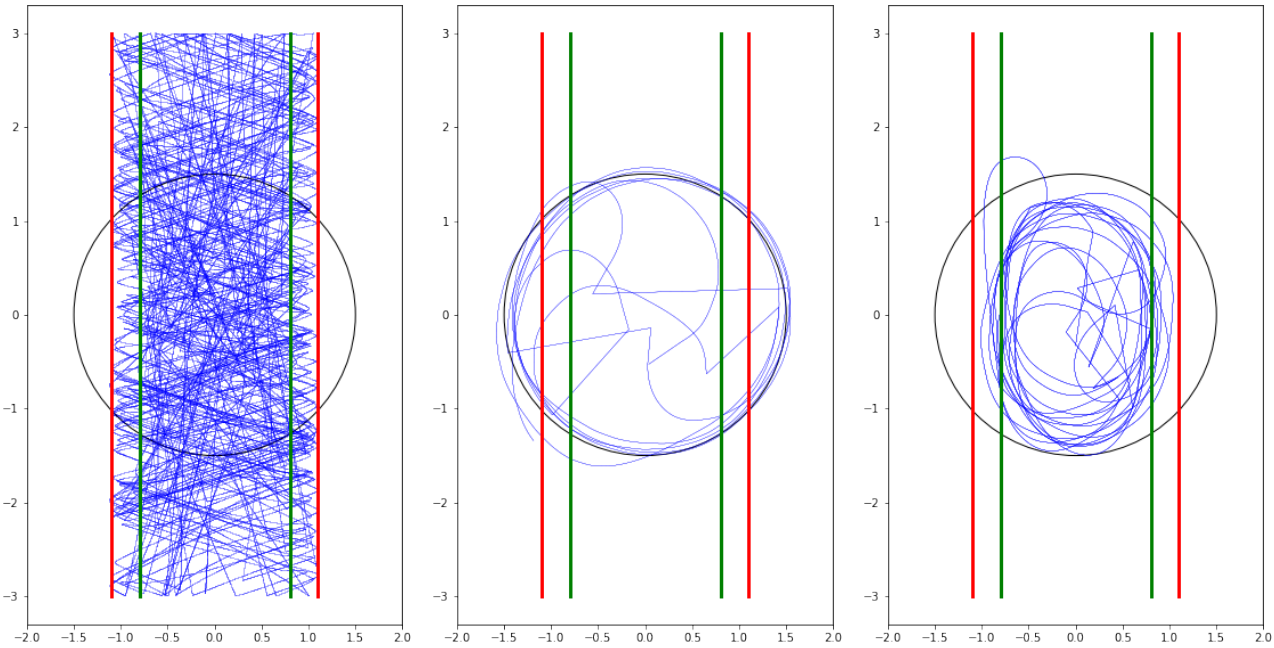
Figure 11: Safety Point Circle Environment

version of the region between the yellow boundary walls.

The CSP can be obtained in two ways - through a heuristic definition or imitation from safe trajectories generated by the human. We used behaviour cloning to learn CSP from safe trajectories. However, it suffered from compounding error. Therefore, the results we observe below are based on a heuristic CSP. After obtaining the cartesian coordinates and orientation of the agent, if it is close to the wall and facing towards it, the CSP generates an action to minimise the linear velocity and generates rotation away from the wall. The CSP implementation on 100 episodes can be seen in 12. We see that that agent is always within $\mathcal{X}_s$ when using the CSP.

5.2.2 Training Curves and Metrics

Pure CSP is when no RL agent is used and the actions are generated only using the CSP. Pure RL is when we use a vanilla DDPG algorithm without any safety intervention. Pure RL (with cost) is the vanilla DDPG with a modified reward (reward = reward - cost). This approach gives negative penalty as a part of the reward. Finally, Safe RL or SafeDDPG is our implementation that uses a CSP and ensures close to 100% safety in both training and deployment.



(a) Pure CSP (b) Pure RL (c) Safe RL (SafeDDPG)

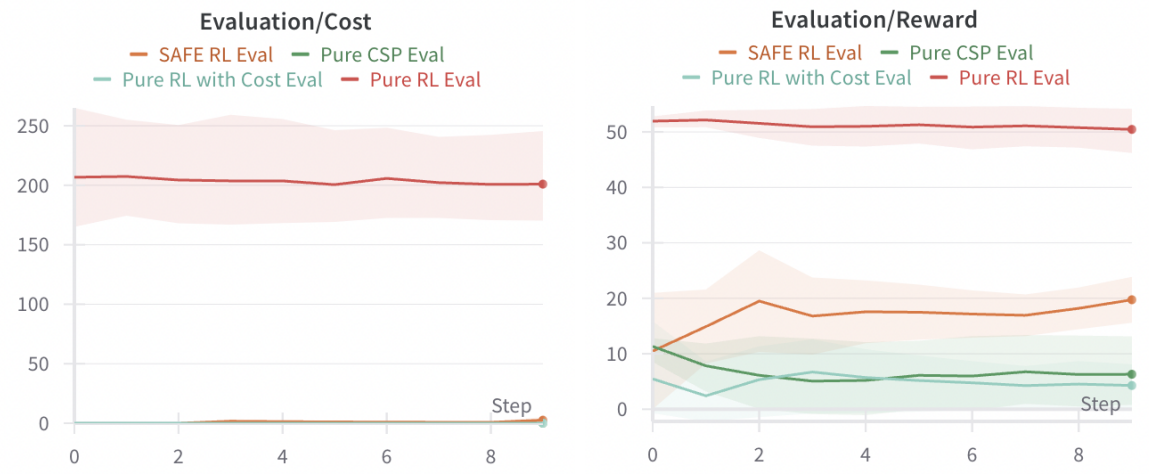
Figure 12: Trajectories with Pure RL, Pure CSP and SafeDDPG

We observe that pure RL learns to gain high rewards but is indifferent to the cost incurred. On the other hand, our implementation ensures almost zero cost throughout training. While Pure RL with modified reward ensures low cost during evaluation, it incurs a high cost during training and its evaluation reward is low.

Figure 13 shows the two extreme approaches with respect to safety and rewards. The CSP (Dark Green) incurs zero safety violations throughout. On the other extreme, a vanilla RL (Red) approach generates the highest reward but compromises completely on the safety. Our approach takes the CSP and improves on it as seen to generate a mean reward of 22.10

Algorithm	**Reward**	**Cost**
Pure RL	47.6	203
Pure RL (Modified Reward)	2.85	10
Pure CSP	4.70	0
Safe RL (SafeDDPG)	**22.10**	**9.5**

Table 4: Reward and Cost during Evaluation/Deployment



(a) Cost Curves during Evaluation (b) Reward Curves during Evaluation

Figure 13: Evaluation Curves for SafeRL and PureRL (Along with comparision with Pure CSP)

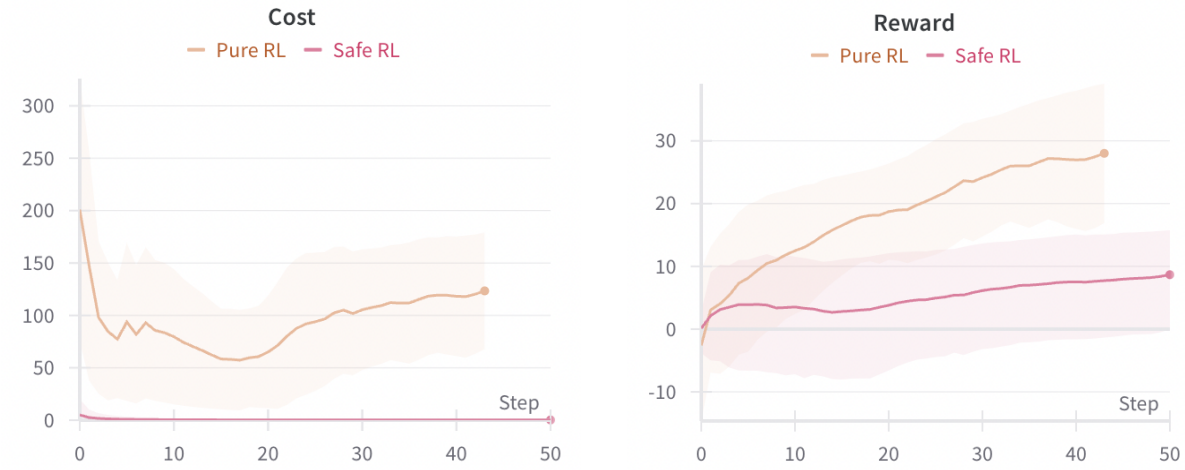


Figure 14: Training Curves for Safe RL and Pure RL

6 Conclusion

In this work, we have developed a framework to formalise expert human input for RL training and deployment. We have attempted to develop a taxonomy for research at the intersection of Safe RL and Human in the loop RL. A modified version of the Deep Deterministic Policy Gradient algorithm was implemented with a safety layer for continuous control. Further, an additional loss term was introduced in the policy gradient term to ensure that the agent learns the safety layer. The algorithm was tested in the inverted pendulum environment

and achieved 100% safety in training. In the safety gymnasium environment, we achieved training with close to zero safety violations and a reward much higher than the value obtained by the CSP.

7 Future Work

7.1 Generating CSP and SSS

The present implementation uses human input and heuristic methods specific to the environment to define the CSP and SSS. It cannot be generalised to complex environments easily. The immediate future work can be about formalising methods to generate CSP and SSS. Imitation learning can be used to learn a CSP from safe expert trajectories. Reachability analysis and model-based approaches can be adopted to accurately determine SSS and marginally safe state space ($\mathcal{X}_{ms}$).

7.2 Use of Options or Macroactions for CSP formulation

"Options" refer to temporally extended actions or sequences of actions that can be treated as single decisions, rather than primitive actions executed at every time step. Sometimes, it is easier to specify a CSP using options instead of specifying an action every timestep. The recovery policy actions under the CSP in the Safety Gymnasium environments can be benefited with Options framework.

7.3 Stochastic CSP

The CSP can be defined as a distribution instead of a deterministic set and stochastic guarantees for safety can be established. For example, the deterministic singleton CSP can be converted into a beta distribution [41] defined in

the action range with the action as the mean and a bayesian approach can be adopted to generate a posterior distribution for the policy after training.

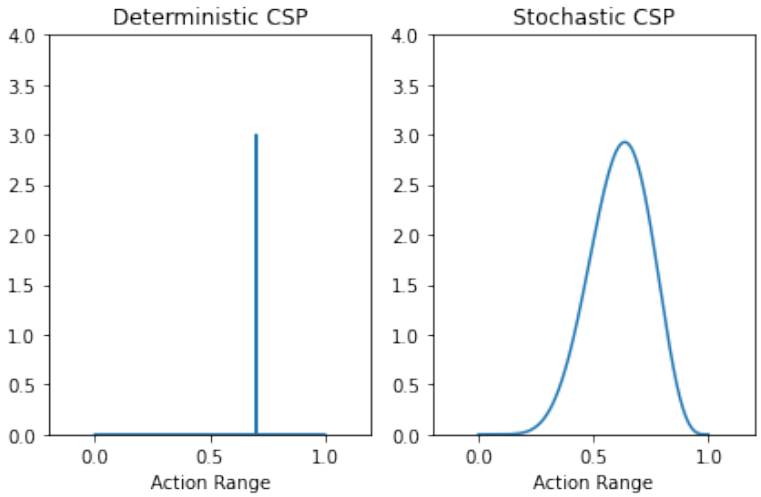


Figure 15: Deterministic to Stochastic CSP

7.4 Implement Other Deep RL Backbone Architectures

The DDPG algorithm was selected in the present implementation for its simple architecture and sample efficiency. However, there were stability issues, overestimation of Q values and the need for extensive hyperparameter tuning. Other algorithms for continuous action space like Soft Actor Critic [42] and Twin Delayed DDPG (TD3) [43] known for their stability can be implemented and compared.

7.5 Complex Environments

The algorithm must of tested on more complex environments. The safety gymnasium has enough benchmarked environments with different complexity levels. Further, ROS + Gazebo implementations can be done. Ultimately, the algorithm can be implemented on real-world mobile robots for navigation tasks.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[3] B. Recht, "A tour of reinforcement learning: The view from continuous control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, pp. 253–279, 2019.

[4] B. Kiumarsi, K. G. Vamvoudakis, H. Modares, and F. L. Lewis, "Optimal and autonomous control using reinforcement learning: A survey," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 6, pp. 2042–2062, 2017.

[5] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[6] A. Agarwal, N. Jiang, S. M. Kakade, and W. Sun, "Reinforcement learning: Theory and algorithms," *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, vol. 32, p. 96, 2019.

[7] E. Altman, *Constrained Markov decision processes*. Routledge, 2021.

[8] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, "Concrete problems in ai safety," *arXiv preprint arXiv:1606.06565*, 2016.

[9] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[10] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International conference on machine learning*, pp. 22–31, PMLR, 2017.

[11] A. Wachi and Y. Sui, "Safe reinforcement learning in constrained markov decision processes," in *International Conference on Machine Learning*, pp. 9797–9806, PMLR, 2020.

[12] C. Ying, X. Zhou, H. Su, D. Yan, N. Chen, and J. Zhu, "Towards safe reinforcement learning via constraining conditional value-at-risk," *arXiv preprint arXiv:2206.04436*, 2022.

[13] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[14] H. Krasowski, J. Thumm, M. Müller, L. Schäfer, X. Wang, and M. Althoff, "Provably safe reinforcement learning: A theoretical and experimental comparison," *arXiv preprint arXiv:2205.06750*, 2022.

[15] W. Saunders, G. Sastry, A. Stuhlmueller, and O. Evans, "Trial without error: Towards safe reinforcement learning via human intervention," *arXiv preprint arXiv:1707.05173*, 2017.

[16] C. Frye and I. Feige, "Parenting: Safe reinforcement learning from human input," *arXiv preprint arXiv:1902.06766*, 2019.

[17] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," *Advances in neural information processing systems*, vol. 30, 2017.

[18] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[19] R. Cheng, G. Orosz, R. M. Murray, and J. 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*, vol. 33, pp. 3387–3395, 2019.

[20] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European control conference (ECC)*, pp. 3420–3431, IEEE, 2019.

[21] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, "Safe exploration in continuous action spaces," *arXiv preprint arXiv:1801.08757*, 2018.

[22] S. Huang and S. Ontañón, "A closer look at invalid action masking in policy gradient algorithms," *arXiv preprint arXiv:2006.14171*, 2020.

[23] S. Huang, S. Ontañón, C. Bamford, and L. Grela, "Gym-μrts: Toward affordable full game real-time strategy games research with deep reinforcement learning," in *2021 IEEE Conference on Games (CoG)*, pp. 1–8, IEEE, 2021.

[24] H. Bharadhwaj, A. Kumar, N. Rhinehart, S. Levine, F. Shkurti, and A. Garg, "Conservative safety critics for exploration," *arXiv preprint arXiv:2010.14497*, 2020.

[25] J. Garcia and F. Fernández, "Safe exploration of state and action spaces in reinforcement learning," *Journal of Artificial Intelligence Research*, vol. 45, pp. 515–564, 2012.

[26] G. Thomas, Y. Luo, and T. Ma, "Safe reinforcement learning by imagining the near future," *Advances in Neural Information Processing Systems*, vol. 34, pp. 13859–13869, 2021.

[27] N. C. Wagener, B. Boots, and C.-A. Cheng, "Safe reinforcement learning using advantage-based intervention," in *International Conference on Machine Learning*, pp. 10630–10640, PMLR, 2021.

[28] D. J. Hejna III and D. Sadigh, "Few-shot preference learning for human-in-the-loop rl," in *Conference on Robot Learning*, pp. 2014–2025, PMLR, 2023.

[29] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, vol. 50, no. 2, pp. 1–35, 2017.

[30] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," *arXiv preprint arXiv:1805.01954*, 2018.

[31] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, "Hg-dagger: Interactive imitation learning with human experts," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8077–8083, IEEE, 2019.

[32] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, 2016.

[33] K. Menda, K. Driggs-Campbell, and M. J. Kochenderfer, "Ensembledagger: A bayesian approach to safe imitation learning," in *2019 IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), pp. 5041–5048, IEEE, 2019.

[34] Y. U. Ciftci, Z. Feng, and S. Bansal, "Safe-gil: Safety guided imitation learning," *arXiv preprint arXiv:2404.05249*, 2024.

[35] C. Wirth, R. Akrour, G. Neumann, and J. Fürnkranz, "A survey of preference-based reinforcement learning methods," *Journal of Machine Learning Research*, vol. 18, no. 136, pp. 1–46, 2017.

[36] H. Hoang, T. Mai, and P. Varakantham, "Imitate the good and avoid the bad: An incremental approach to safe reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 12439–12447, 2024.

[37] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," *Advances in neural information processing systems*, vol. 30, 2017.

[38] J. Leike, M. Martic, V. Krakovna, P. A. Ortega, T. Everitt, A. Lefrancq, L. Orseau, and S. Legg, "Ai safety gridworlds," *arXiv preprint arXiv:1711.09883*, 2017.

[39] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, N. Heess, and Y. Tassa, "dm_control: Software and tasks for continuous control," *Software Impacts*, vol. 6, p. 100022, 2020.

[40] J. Ji, B. Zhang, J. Zhou, X. Pan, W. Huang, R. Sun, Y. Geng, Y. Zhong, J. Dai, and Y. Yang, "Safety gymnasium: A unified safe reinforcement learning benchmark," *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[41] P.-W. Chou, *The beta policy for continuous control reinforcement learning.* PhD thesis, Master's thesis. Pittsburgh: Carnegie Mellon University, 2017.

[42] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, pp. 1861–1870, PMLR, 2018.

[43] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*, pp. 1587–1596, PMLR, 2018.