Learning Safe Behaviours in Goal-Reaching Tasks

Abstract

An important problem in reinforcement learning is designing agents that learn to solve tasks safely in an environment. A common solution is for a human expert to define either a penalty in the reward function or a cost to be minimised when reaching unsafe states. However, this is non-trivial, since too small a penalty may lead to agents that reach unsafe states, while too large a penalty increases the time to convergence. Additionally, the difficulty in designing reward or cost functions can increase with the complexity of the problem. Hence, for a given environment with a set of unsafe states, we are interested in finding the smallest penalty whose optimal policy minimises the probability of reaching unsafe states, irrespective of task rewards. We refer to this specific penalty as the *Minmax penalty*, and show that it can be obtained by taking into account both the controllability and diameter of an environment. We provide a simple practical algorithm for an agent to learn this Minmax penalty while learning the task policy, and demonstrate that this leads to agents that learn safe policies in high-dimensional continuous control environments.

1 Introduction

Reinforcement learning (RL) has in recent years seen great success across a variety of domains, including video games [Shao *et al.*, 2019], robotics applications [Kalashnikov *et al.*, 2018; Kahn *et al.*, 2018] and autonomous driving [Kiran *et al.*, 2021]. In many domains, it is important to ensure that an RL agent completes a task while simultaneously avoiding unsafe or costly behaviour. For example, a robot navigating a real-world must avoid colliding with objects and actors around it, while simultaneously learning to solve the required task. An example of this desired behaviour is illustrated in Figure 1.

Many approaches in RL deal with this problem by allocating arbitrary penalties to unsafe states when hand-crafting a reward function. However, the problem of specifying a reward function for desirable, safe behaviour is notoriously difficult [Amodei *et al.*, 2016]. Indeed, penalties that are too small will result in unsafe behaviour being learned, while penalties that are too large may result in very long learning times [Ray *et al.*, 2019]. Furthermore, these rewards must be specified for each new environment an agent faces, limiting its ability to act autonomously without human guidance. It is therefore impractical to require that a human designer specify penalties to guarantee optimal but safe behaviours.

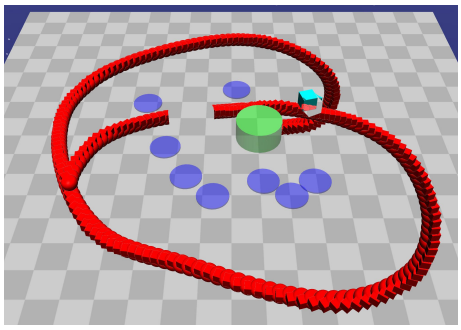


Figure 1: An example safe trajectory where a point mass agent learns to reach a goal location (green cylinder) while avoiding unsafe regions (blue circles).

When safety is an explicit goal, a common approach is to constrain policy learning according to some threshold on cumulative cost [Schulman *et al.*, 2015; Ray *et al.*, 2019; Achiam *et al.*, 2017]. While these approaches have been effective, they require the design of a cost function whose specification can be as difficult as designing a reward function. Additionally, these methods may still result in unacceptably frequent constraint violation in practice.

Rather than attempting to minimise a cost, which is only a proxy for unsafe behaviour, it may be prudent to encapsulate the unsafe behaviour in the reward function, such that the optimal agent behaviour is to minimise the probability of reaching unsafe states. Indeed, this approach is consistent with the *reward hypothesis* [Sutton and Barto, 2018]. which states: " That all of what we mean by goals and purposes can be well thought of as maximization of the expected value of the cumulative sum of a received scalar signal (reward). " Therefore, the question which we examine in this work is to determine the smallest penalty that should be assigned to unsafe states such that this behaviour is observed. Rather than requiring a human expert's input, we show that this penalty can be defined by the *controllability* and *diameter* of an environment, and in practice can be autonomously learned by an agent using its

current value estimates.

More specifically, we make the following contributions: (i) we define the analytical form of the penalty and prove that its use results in learned behaviours that minimise the probability of visiting unsafe states; (ii) we propose an algorithm for learning this penalty online; and (iii) we empirically demonstrate that our method outperforms existing baselines in high-dimensional, continuous control tasks, learning to complete the tasks while minimising unsafe behaviour. Our results demonstrate that, while prior methods often violate safety constraints, our approach results in agents capable of learning to solve tasks while avoiding unsafe states.

2 Background

In the typical RL setting that we consider, the task an agent must learn to complete is modelled by a Markov Decision Process (MDP). An MDP is defined as a tuple $(\mathcal{S}, \mathcal{A}, \rho, r)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $\rho : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is the transition function, and $r \in \mathbb{R}$ is a reward function bounded by $[R_{\text{MIN}}\ R_{\text{MAX}}]$. Our particular focus is on undiscounted MDPs which model stochastic shortest path problems in which an agent must reach some goal states in the set of absorbing states $\mathcal{G} \subseteq \mathcal{S}$ [Bertsekas and Tsitsiklis, 1991], while avoiding unsafe absorbing states $\mathcal{G}^! \in \mathcal{G}$. The set of non-absorbing states $\mathcal{S} \setminus \mathcal{G}$ are referred to as *internal states*.

A *policy* $\pi : \mathcal{S} \mapsto \mathcal{A}$ is a mapping from states to actions. The *value function* $V^\pi(s) = \mathbb{E}[\sum_{t=0}^{\infty} r(s_t, a_t)]$ associated with a policy specifies the expected return under that policy starting from state s. The goal of an agent in these settings is to learn an optimal policy π^* that maximises the value function $V^{\pi^*}(s) = V^*(s) = \max_\pi V^\pi(s)$ for each $s \in \mathcal{S}$. Since tasks are undiscounted, an optimal policy is guaranteed to exist by assuming that the value function of *improper policies* is unbounded below—where *proper policies* are those that are guaranteed to reach an absorbing state [Van Niekerk *et al.*, 2019]. Since there always exist a deterministic optimal policy [Sutton *et al.*, 1998], and all optimal policies are proper, we will focus our attention to the set of all deterministic proper policies Π.

3 Avoiding Unsafe Absorbing States

Given an MDP, our goal here is to determine the smallest penalty (hence largest reward) to give at unsafe states to obtain safe policies. We formally define a safe policy as one that minimises the probability of reaching any unsafe terminal state from any internal state:

Definition 1 (Safe Policy) *Where s_T is the final state of a trajectory, let $P_s^\pi(s_T \in \mathcal{G}^!)$ be the probability of reaching $\mathcal{G}^!$ from s under a proper policy $\pi \in \Pi$. Then π is called safe if*

$$\pi \in \arg\min_{\pi'} P_s^{\pi' \in \Pi}(s_T \in \mathcal{G}^!),$$

Consider for example the simple *chain-walk* MDP in Figure 2. The agent here has two actions a_1, a_2, always receives -1 step reward at internal states, and needs to go to the goal state s_3 but not the unsafe state s_1. Since the transitions per action are stochastic, controlled by $p \in [0\ 1]$, and s_3 is further from

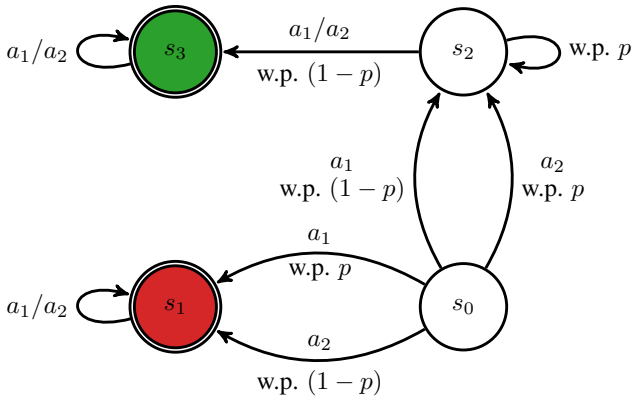


Figure 2: Illustration of a simple chain-walk MDP. The MDP consists of four states $s_0, \ldots, s_3$ and two actions a_1, a_2. s_1, coloured in red, is an unsafe absorbing state, while s_3 (green) is a safe absorbing state. The initial state is s_0, and the diagram denotes the transitions and associated probabilities when executing actions. The absorbing transitions have reward of 0 while all other transitions have a reward of -1.

the start state s_0 than s_1 is, the agent may not always be able to avoid s_1. In fact, for -1 step reward at all internal states, the optimal policy is to always pick a_2 which always reaches s_1. However, for a sufficiently high penalty for going to s_1 (for example -3) and $p = 0$, the optimal policy is to always pick action a_1 which always reaches s_3. Finally, the larger p is, the longer the agent is expected to get stuck in s_2 if it reaches that state and therefore the penalty for s_1 must be larger so that it is not optimal to go to s_1 to avoid spending a long time in s_2. To capture this relationship between the stochasticity of an MDP and the required penalty to obtain safe policies, we introduce the notion of *controllability*, which measures how much of an effect actions have on the transition dynamics.

Definition 2 (Controllability) *Define the degree of controllability of the environment with*

$$C = \min_{s \in \mathcal{S} \setminus \mathcal{G}} \min_{\substack{\pi_1, \pi_2 \in \Pi \\ \Delta P_s(\pi_1, \pi_2) > 0}} \Delta P_s(\pi_1, \pi_2),$$

where $\Delta P_s(\pi_1, \pi_2) \coloneqq P_s^{\pi_1}(s_T \notin \mathcal{G}^!) - P_s^{\pi_2}(s_T \notin \mathcal{G}^!)$.

C measures the degree of controllability of the environment by simply taking the smallest difference between the probability of reaching a safe goal state by following any two policies. For example, by computing C with deterministic policies, we can see that a deterministic task will have $C = 1$ and that C will tend to 0 as the task's stochasticity increases. This can be seen in Figure 3 for the chain-walk MDP with different choices for p.

Since we are interested in controllable MDPs, we assume that there exist at least two policies π_1 and π_2 such that $P_s^{\pi_1}(s_T \notin \mathcal{G}^!) - P_s^{\pi_2}(s_T \notin \mathcal{G}^!) > 0$ for at least one internal state. In the chain-walk MDP, these are the always a_1 and always a_2 policies, which are the only deterministic proper policies.

Clearly, the size of the penalty that needs to be given for unsafe states also depends on the *size* of the MDP. We define

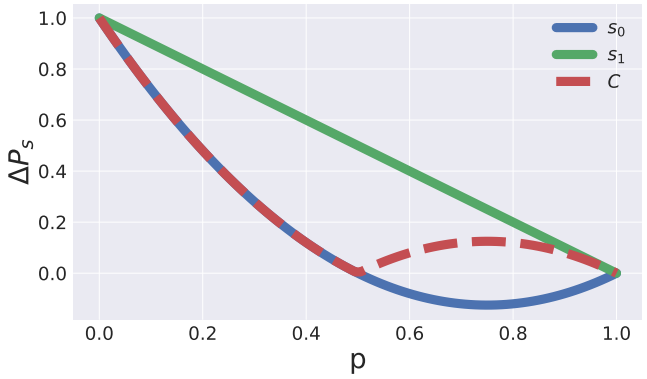


Figure 3: Controllability of the chain-walk MDP for varying p. There are only 2 deterministic proper policies $\pi_1(s) = a_1, \pi_2(s) = a_2$, giving $\Delta P_{s_0}(\pi_1, \pi_2) = (p-1)(p-1) - p(p-1)$ and $\Delta P_{s_2}(\pi_1, \pi_2) = (p-1)$.

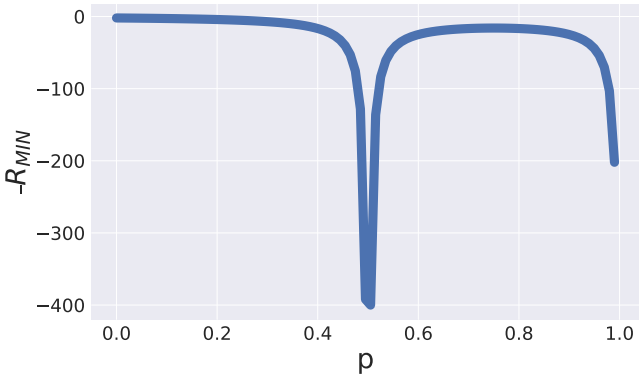


Figure 4: Minmax penalty for the chain-walk MDP for varying p. $p = 0.5$ and $p = 1$ are excluded since their corresponding controllability is 0. They respectively lead to a completely stochastic MDP and one with a zero probability of reaching the safe goal s_3.

this *size* as the *diameter* of the MDP, which is highest number of expected timesteps required to reach an absorbing state from an internal state:

Definition 3 (Diameter) *Define the diameter of an MDP M as*

$$D = \max_{s \in \mathcal{S} \setminus \mathcal{G}} \max_{\pi \in \Pi} \mathbb{E}\left[T(s_T \in \mathcal{G} | \pi)\right],$$

where T is the number of timesteps required to first reach $\mathcal{G}$ from s under a proper policy π.

Given the controllability and diameter of an MDP, we can now obtain the maximum reward which minimises the probability of reaching unsafe states. We can this reward the *minmax penalty* 4, and prove in Theorem 1 that any reward less than or equal to it leads to safe optimal policies.

Definition 4 (Minmax Penalty) *Define the reward at unsafe absorbing states as*

$$\bar{R}_{MIN} = \min\{R_{MIN}, (R_{MIN} - R_{MAX})\frac{D}{C}\}.$$

This choice of $\bar{R}_{\text{MIN}}$ says that since stochastic shortest path tasks require an agent to learn to achieve desired terminal states, if the agent enters an unsafe terminal state, it should receive the largest penalty possible by a proper policy. For example, Figure 4 shows the minmax penalty for varying choices of p.

We now show that $\bar{R}_{\text{MIN}}$ is indeed the smallest penalty (largest reward) that minimises the probability of reaching unsafe terminal states.

Theorem 1 (Safety Bound) *Let π^* be the optimal policy for an MDP whose rewards at unsafe goal states satisfy $R(s,a) \leq \bar{R}_{MIN}$. Then π^* is safe.*

Proof Since π^* is optimal, it is also proper and hence must reach $\mathcal{G}$. Assume there exists another proper policy π such that

$$P_s^{\pi^*}(s_T \notin \mathcal{G}^!) < P_s^{\pi}(s_T \notin \mathcal{G}^!).$$

Let V^{π^*} and V^{π} be the value functions for the respective policies. Then,

$$\bar{V}^{\pi^*}(s) \geq \bar{V}^{\pi}(s)$$

$$\implies \mathbb{E}_s^{\pi^*}\left[\sum_{t=0}^{\infty} \gamma^t \bar{R}(s_t, a_t)\right] \geq \mathbb{E}_s^{\pi}\left[\sum_{t=0}^{\infty} \gamma^t \bar{R}(s_t, a_t)\right]$$

$$\implies \mathbb{E}_s^{\pi^*}\left[G^{T-1} + \bar{R}(s_T, a_T)\right] \geq \mathbb{E}_s^{\pi}\left[G^{T-1} + \bar{R}(s_T, a_T)\right],$$

where $G^{T-1} = \sum_{t=0}^{T-1} \gamma^t \bar{R}(s_t, a_t)$ and T is a random variable denoting the time at which $s_T \in \mathcal{G}$.

$$\implies \mathbb{E}_s^{\pi^*}\left[G^{T-1}\right] + \left(P_s^{\pi^*}(s_T \notin \mathcal{G}^!) R(s_T, a_T) + P_s^{\pi^*}(s_T \in \mathcal{G}^!)\bar{R}_{\text{MIN}}\right) \geq \mathbb{E}_s^{\pi_g}\left[G^{T-1}\right] + \left(P_s^{\pi_g}(s_T \notin \mathcal{G}^!) R(s_T, a_T) + P_s^{\pi_g}(s_T \in \mathcal{G}^!)\bar{R}_{\text{MIN}}\right),$$

using definition of $\bar{R}$, where $a_T = \pi^*(g)$.

$$\implies \mathbb{E}_s^{\pi^*}\left[G^{T-1}\right] + \left(P_s^{\pi^*}(s_T \in \mathcal{G}^!) - P_s^{\pi}(s_T \in \mathcal{G}^!)\right)\bar{R}_{\text{MIN}} \geq \mathbb{E}_s^{\pi_g}\left[G^{T-1}\right] + \left(P_s^{\pi}(s_T \notin \mathcal{G}^!) - P_s^{\pi^*}(s_T \notin \mathcal{G}^!)\right) R(s_T, a_T)$$

$$\implies \mathbb{E}_s^{\pi^*}\left[G^{T-1}\right] + \left(P_s^{\pi^*}(s_T \in \mathcal{G}^!) - P_s^{\pi}(s_T \in \mathcal{G}^!)\right)(R_{\text{MIN}} - R_{\text{MAX}})\frac{D}{C} \geq \mathbb{E}_s^{\pi_g}\left[G^{T-1}\right] + \left(P_s^{\pi}(s_T \notin \mathcal{G}^!) - P_s^{\pi^*}(s_T \notin \mathcal{G}^!)\right) R(s_T, a_T),$$

using definition of $\bar{R}_{\text{MIN}}$.

$$
\begin{aligned}
&\implies \mathbb{E}_s^{\pi^*}\left[G^{T-1}\right] + (R_{\text{MIN}} - R_{\text{MAX}})D \\
&\quad \geq \\
&\quad \mathbb{E}_s^{\pi_g}\left[G^{T-1}\right] \\
&\quad + \left(P_s^{\pi}(s_T \notin \mathcal{G}^!) - P_s^{\pi^*}(s_T \notin \mathcal{G}^!)\right) R(s_T, a_T), \\
&\quad \text{using definition of } C. \\
&\implies \mathbb{E}_s^{\pi^*}\left[G^{T-1}\right] - R_{\text{MAX}}D \\
&\quad \geq \\
&\quad \mathbb{E}_s^{\pi_g}\left[G^{T-1}\right] \\
&\quad + \left(P_s^{\pi}(s_T \notin \mathcal{G}^!) - P_s^{\pi^*}(s_T \notin \mathcal{G}^!)\right) R(s_T, a_T) - R_{\text{MIN}}D \\
&\implies \mathbb{E}_s^{\pi^*}\left[G^{T-1}\right] - R_{\text{MAX}}D \geq 0, \text{ since } \mathbb{E}_s^{\pi_g}\left[G^{T-1}\right] \\
&\quad + \left(P_s^{\pi}(s_T \notin \mathcal{G}^!) - P_s^{\pi^*}(s_T \notin \mathcal{G}^!)\right) R(s_T, a_T) \geq R_{\text{MIN}}D \\
&\implies \mathbb{E}_s^{\pi^*}\left[G^{T-1}\right] \geq R_{\text{MAX}}D.
\end{aligned}
$$

But this is a contradiction since the expected return of following an optimal policy up to a terminal state without the reward for entering the terminal state must be strictly less than receiving R_{MAX} for every step of the longest possible trajectory to $\mathcal{G}$. Hence we must have

$$\pi^* \in \arg\min_{\pi} P_s^{\pi}(s_T \in \mathcal{G}^!).$$

■

Theorem 1 says that for any MDP whose rewards at unsafe goal states are bounded above by $\bar{R}_{\text{MIN}}$, the optimal policy minimises the probability of reaching those states.

4 Extimating the Minmax Penalty

The Minmax penalty can easily be estimated using observations of the reward and an agent's estimate of the value function. This method requires initial estimates of R_{MIN} and R_{MAX}, which in this work are initialised to 0. After every interaction with the environment, having obtained an observation of the reward, the estimate of the Minmax penalty $\bar{R}_{\text{MIN}}$ is updated according to the update rule in Algorithm 1. Whenever an agent encounters an unsafe state, the reward can be replaced by $\bar{R}_{\text{MIN}}$ so as to disincentivise unsafe behaviour.

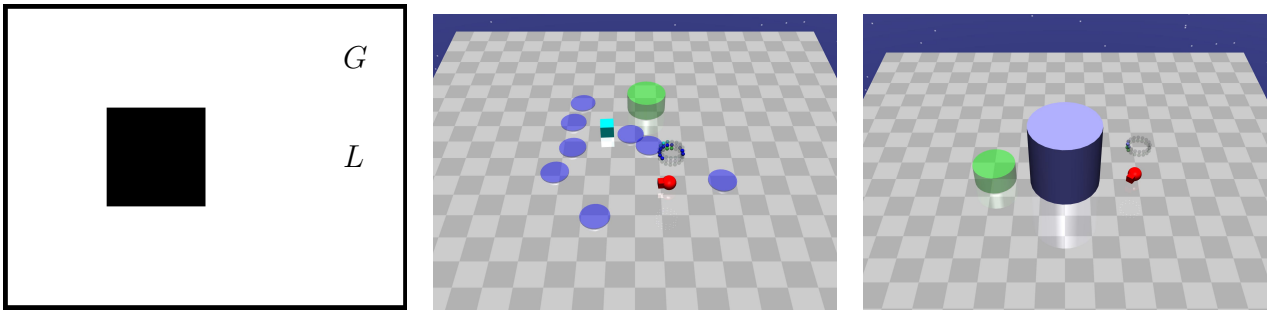
Algorithm 1: Learning the Minmax Penalty

Input : T
Initialise : $R_{\text{MIN}} = 0, R_{\text{MAX}} = 0, V_{\text{MIN}} = R_{\text{MIN}}, V_{\text{MAX}} = R_{\text{MAX}}$
  observe s_0
  initialise V
  for t in T **do**
    select action a_t
    observe s_{t+1}, r_t
    $R_{\text{MIN}} \leftarrow \min(R_{\text{MIN}}, r_t)$
    $R_{\text{MAX}} \leftarrow \max(R_{\text{MAX}}, r_t)$
    $V_{\text{MIN}} \leftarrow \min(V_{\text{MIN}}, R_{\text{MIN}}, V(s_t))$
    $V_{\text{MAX}} \leftarrow \max(V_{\text{MAX}}, R_{\text{MAX}}, V(s_t))$
    $\bar{R}_{\text{MIN}} = \min(R_{\text{MIN}}, V_{\text{MIN}} - V_{\text{MAX}})$
    if s_{t+1} is unsafe **then**
      $r_t \leftarrow \bar{R}_{\text{MIN}}$
    end if
    update $V(s)$ with r_t
  end for

5 Experiments

To demonstrate that an agent using the Minmax penalty for unsafe states is able to learn an optimally safe policy, we evaluate our approach across two domains.

Lava Gridworld The first domain in which we evaluate our approach is a simple tabular environment, the LAVA GRIDWORLD domain, in which an agent must reach a goal location while avoiding a lava location. A wall is also present in the environment and, while not unsafe, must be navigated around. The environment has a *slip probability* of 0.25, so that with probability 0.25 the agent's action is overriden with a random action. The agent receives a positive reward (+1) for reaching the goal, as well as negative rewards (-0.1) at each timestep to incentivise taking the shortest path to the goal.



(a) LAVA GRIDWORLD (b) POINTGOAL1 from Safety Gym (c) Safety Gym Pillar

Figure 5: Experiments domains. (a) An agent must navigate a gridworld to goal G while avoiding the lava, denoted L. The black region represents an impassable wall that must be navigated around. (b) and (c) The agent in red must navigate to the green region, while avoiding the blue hazards.

To test our approach, we modify Q-learning [Watkins, 1989] with ϵ-greedy exploration to use the Minmax penalty whenever the lava state is reached, following the procedure outlined in Section 4. The action-value function is initialised to 0 for all states and actions, $\epsilon = 0.1$ and the learning rate $\alpha = 0.1$. The experiments are run over 10,000 episodes and averaged over 70 runs.

Safety Gym PointGoal1 The second domain in which we evaluate our approach is a modified version of the POINTGOAL1 task from OpenAI's Safety Gym environments [Ray *et al.*, 2019], which represents a complex, high-dimensional, continuous control task. Here a simple robot must navigate to a goal location across a 2D plane while avoiding a number of obstacles. The agent uses *pseudo-lidar* to observe the distance to objects around it, and the action space is continuous over two actuators controlling direction and forward velocity. The goal's location is randomly reset when the agent reaches it, while the locations of the obstacles remain unchanged. The agent is rewarded for reaching the goal, as well as for moving towards it. We modify the environment so that any collision with a hazard results in episode termination.

As a baseline representative of typical RL approaches, we use Trust Region Policy Optimisation (TRPO) [Schulman *et al.*, 2015]. Our approach modifies TRPO (denoted TRPO-Minmax) to use the estimate of the minmax penalty as described in Algorithm 1. To represent constraint-based approaches, we compare against Constrained Policy Optimisation (CPO) [Achiam *et al.*, 2017] and TRPO with Lagrangian constraints (TRPO-Lagrangian) [Ray *et al.*, 2019]. All baselines use the implementations provided by [Ray *et al.*, 2019].

As in [Ray *et al.*, 2019], all approaches use feed-forward MLPs, value networks of size (256,256), and $tanh$ activation functions. Results are averaged over 10 runs with different seeds, where episode lengths are capped at 1000 interactions. The initial penalty for the constrained algorithms is set to 1, and the initial values of R_{MIN} and R_{MAX} for our approach are set to 0.

Safety Gym Pillar The final domain we consider is a custom Safety Gym PILLAR environment, in which the a simple robot must navigate to a goal location around a large hazard. Aside from the locations of the goal and hazard, the environment is identical to the POINTGOAL1 environment. Results are averaged over 5 runs with different seeds.

5.1 Results

Does the Minmax penalty give the best tradeoff between safety and sample efficiency?

To answer this question we examine the average episode length (converged timesteps) and failure rate (converged failure rate) of an agent that has converged to a policy in the Lava Gridworld domain, as well as the number of steps to converge (total timesteps). These metrics are compared against the size of the penalty for entering the lava, ranging from -5 to 0. Normalised results are plotted in Figure 6.

These results clearly show the tradeoff between choices of penalty size, with large penalties resulting in longer convergence times and small penalties resulting in unsafe policies. Furthermore, these results show that the learned estimate of the Minmax penalty provides a good balance between the failure rate and the convergence time.

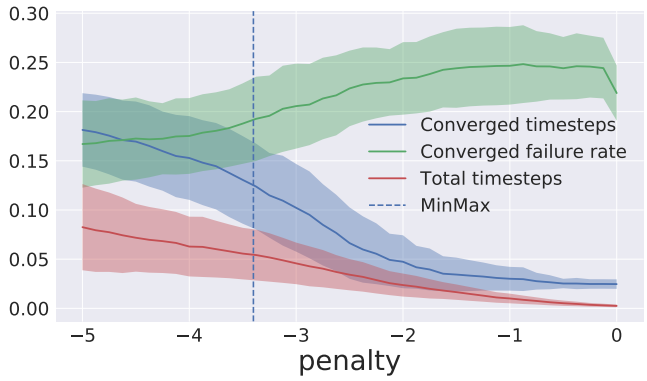


Figure 6: Timesteps to convergence and failure rate for our Q-learning approach and various choices of penalty in the LAVA GRIDWORLD. The learned value of the Minmax penalty is also pictured.

How does stochasticity affect the Minmax penalty?

To answer this question, we compare the performance of our modified Q-learning approach across three values of the slip probability of the LAVA GRIDWORLD. A slip probability of 0 represents a fully deterministic environment, while a slip probability of 0.5 represents a more stochastic environment. Results are plotted in Figure 7.

In the case of the fully deterministic environment, the agent is able to use a relatively small penalty to consistently maximise returns and minimise failure rate. As the stochasticity of the environment increases, a larger penalty is learned to incentivise longer, safer policies. Given the starting position of the agent next to the lava, the failure rate inevitably increases with increased stochasticity.

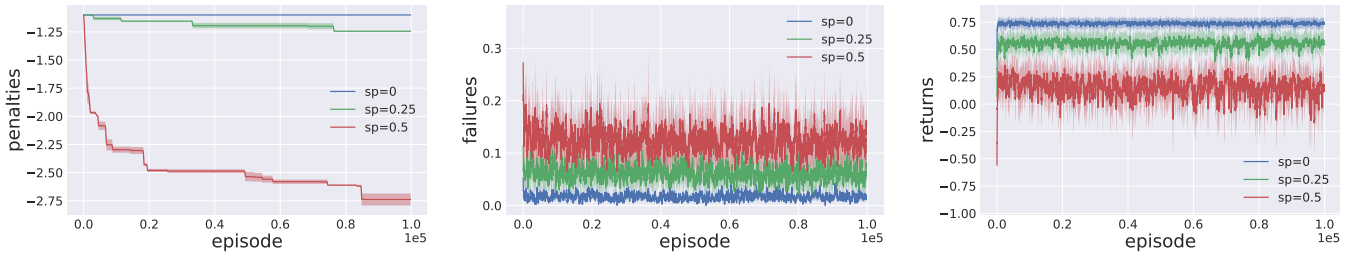
We can therefore conclude that increased stochasticity in an environment, which can increase the risk of entering unsafe states, necessitates learning a larger Minmax penalty in order to incentivise more risk-averse behaviour.

Does the learned Minmax penalty improve safety in high-dimensional domains requiring function approximation?

To answer this question we turn to our major results in the POINTGOAL1 environment, plotted in Figure 8.

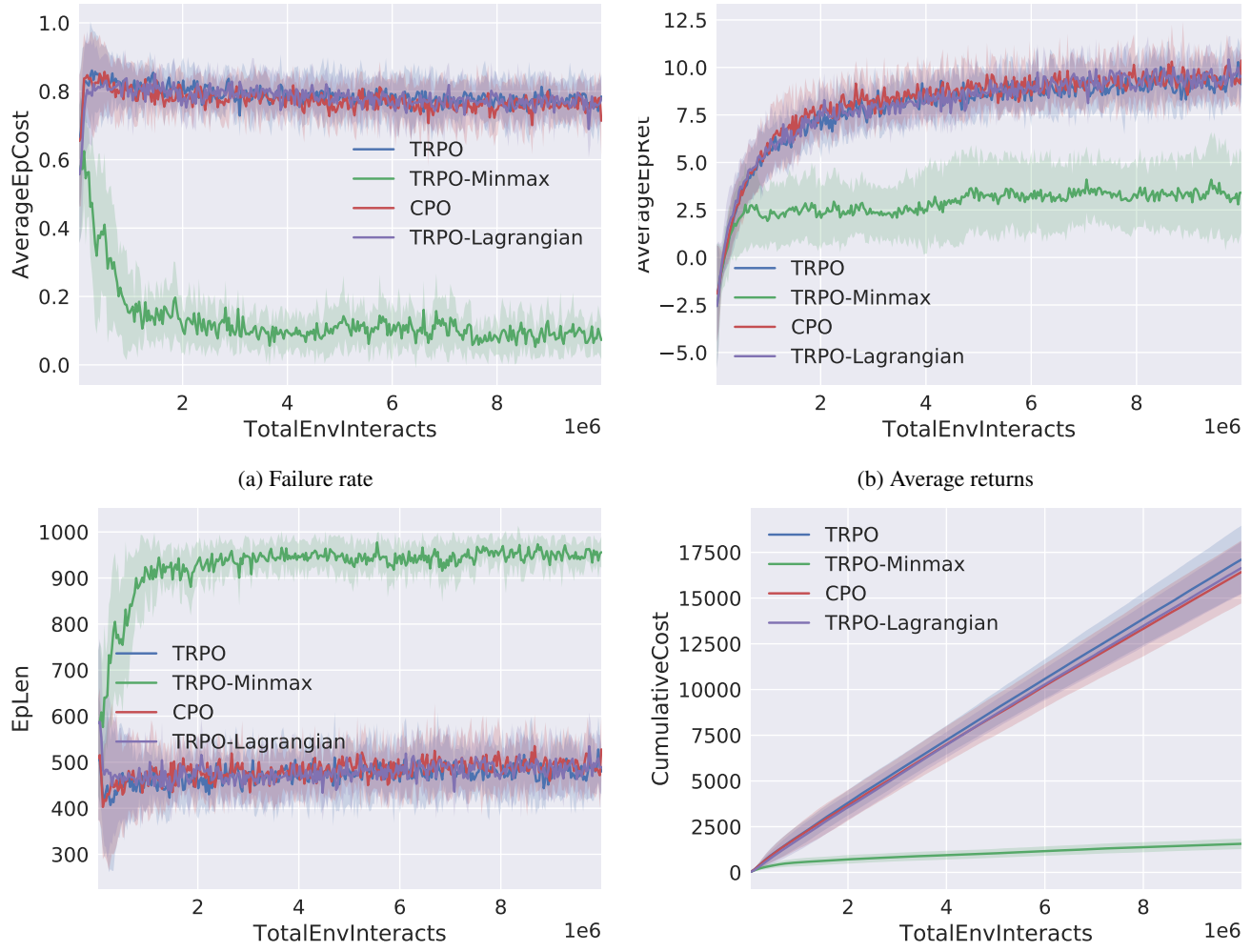
The baselines all achieve similar performance, maximising returns but maintaining a relatively high failure rate. By examining the average episode length (Figure 8d), we can conclude that the baselines have learned risky policies that maximise rewards over short trajectories that are highly likely to result in collisions.

By comparison, TRPO-Minmax uses the learned Minmax penalty to dramatically reduce failure rate (and thus cumulative failures and as a result is able to maintain significantly longer trajectories. The average returns achieved, as well as the trajectories observed (for example, in Figure 1), indicate that the improved safety does not compromise the agent's ability to learn to complete the task, although returns are lower due



(a) Learned penalty (b) Failure rate (c) Average returns

Figure 7: Effect of increase in the slip probability of the LAVA GRIDWORLD on the learned Minmax penalty and corresponding failure rate and returns.



(a) Failure rate (b) Average returns

(c) Average episode length (d) The cumulative cost.

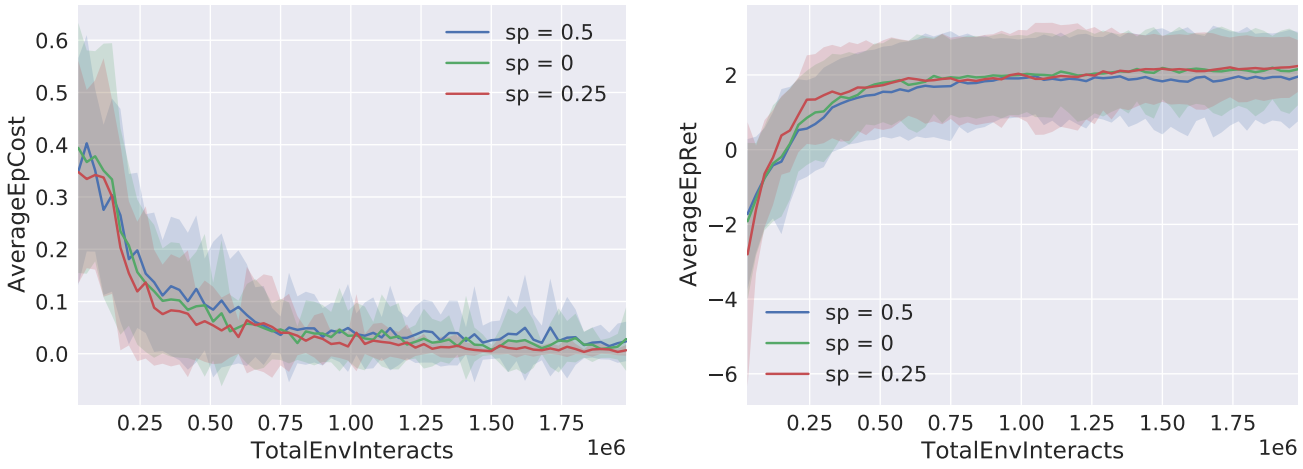
Figure 8: Comparison with baselines in the POINTGOAL1 environment.

to the dense reward function that incentivises moving towards the goal.

We can conclude therefore that our approach is able to produce safe goal-reaching policies even in complex, high-dimensional domains that require function approximation.

Is the learned Minmax penalty robust to noisy continuous dynamics?

To examine the effect of noisy continuous dynamics on the performance of TRPO-Minmax, we compare results of the method in the PILLAR environment with varying levels of



(a) Failure rate

(b) Average returns

Figure 9: Comparisons of TRPO-Minmax with varying slip probabilities in the PILLAR environment.

stochasticity, in a similar vein to the experiments in Section 5.1. Once again, the value of the slip probability denotes the probability of overriding the agent's action with a random action. Results are plotted in Figure 9.

The value of the slip probability appears to have little effect on both the failure rate of and average returns obtained by the agent, indicating that in an environment with a relatively simple hazard configuration such as the PILLAR environment, the learned Minmax penalty is indeed robust to noisy continuous dynamics.

5.2 Related Work

The problem of designing reward functions to produce desired policies in RL settings is well-studied [Singh *et al.*, 2009]. Particular focus has been placed on the practice of *reward-shaping*, in which an initial reward function provided by an MDP is augmented in order to improve the rate at which an agent learns a policy [Ng *et al.*, 1999; Devidze *et al.*, 2021]. These approaches differ from ours in that they seek to find reward functions that improve convergence while not differing in optimality from an initial reward function, while ours seeks to determine the optimal rewards for terminal states in order to disincentivise undesirable behaviours.

Disincentivising or preventing undesirable behaviours is core to the field of safe RL. A popular approach is to define constraints on the behaviour of an agent [Altman, 1999; Achiam *et al.*, 2017; Ray *et al.*, 2019]. Agents in these settings are tasked with limiting the accumulation of costs associated with violating safety constraints while simultaneously maximising reward. Examples of these approaches include the CPO [Achiam *et al.*, 2017] and TRPO-Lagrangian [Ray *et al.*, 2019] algorithms outlined in Section 5.

Other approaches include relying on interventions from a model [Dalal *et al.*, 2018; Wagener *et al.*, 2021] or human [Tennenholtz *et al.*, 2022] to prevent unsafe actions from being executed.

6 Conclusion

In this work, we present the Minmax penalty, which takes into account the diameter and controllability of an environment in order to minimise the probability of encountering unsafe states. We prove that the penalty does indeed minimise this probability, and present a method that uses an agent's value estimates to learn an estimate of the penalty.

Our results in tabular and high-dimensional continuous settings have demonstrated that, by encoding the safe behaviour directly in the reward function via the Minmax penalty, our method is able to solve tasks while prioritising safety, learning safer policies than popular constraint-based approaches.

Our method is easy to incorporate with any off-the-shelf RL algorithms that maintain value estimates, requiring no changes to the algorithms themselves. By autonomously learning the penalty, our method also alleviates the need for a human designer to manually tweak rewards or cost functions to elicit safe behaviour.

While it may be feasible to handcraft reward or cost functions to induce safe behaviour for individual tasks, our ultimate aim is to have general agents capable of operating safely in a variety of environments, and thus we cannot rely on human-crafted reward or cost functions. We see this as a step towards truly autonomous agents capable of independently learning how to safely solve tasks.

References

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

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

[Amodei *et al.*, 2016] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[Bertsekas and Tsitsiklis, 1991] Dimitri P Bertsekas and John N Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3):580–595, 1991.

[Dalal *et al.*, 2018] 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.

[Devidze *et al.*, 2021] Rati Devidze, Goran Radanovic, Parameswaran Kamalaruban, and Adish Singla. Explicable reward design for reinforcement learning agents. *Advances in Neural Information Processing Systems*, 34:20118–20131, 2021.

[Kahn *et al.*, 2018] Gregory Kahn, Adam Villaflor, Bosen Ding, Pieter Abbeel, and Sergey Levine. Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5129–5136. IEEE, 2018.

[Kalashnikov *et al.*, 2018] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[Kiran *et al.*, 2021] B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[Ng *et al.*, 1999] Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287, 1999.

[Ray *et al.*, 2019] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking Safe Exploration in Deep Reinforcement Learning. 2019.

[Schulman *et al.*, 2015] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[Shao *et al.*, 2019] Kun Shao, Zhentao Tang, Yuanheng Zhu, Nannan Li, and Dongbin Zhao. A survey of deep reinforcement learning in video games. *arXiv preprint arXiv:1912.10944*, 2019.

[Singh *et al.*, 2009] Satinder Singh, Richard L Lewis, and Andrew G Barto. Where do rewards come from. In *Proceedings of the annual conference of the cognitive science society*, pages 2601–2606. Cognitive Science Society, 2009.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Sutton *et al.*, 1998] R. Sutton, A. Barto, et al. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.

[Tennenholtz *et al.*, 2022] Guy Tennenholtz, Nadav Merlis, Lior Shani, Shie Mannor, Uri Shalit, Gal Chechik, Assaf Hallak, and Gal Dalal. Reinforcement learning with a terminator. *arXiv preprint arXiv:2205.15376*, 2022.

[Van Niekerk *et al.*, 2019] Benjamin Van Niekerk, Steven James, Adam Earle, and Benjamin Rosman. Composing value functions in reinforcement learning. In *International conference on machine learning*, pages 6401–6409. PMLR, 2019.

[Wagener *et al.*, 2021] 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.

[Watkins, 1989] C. Watkins. *Learning from delayed rewards*. PhD thesis, King's College, Cambridge, 1989.