

ProSH: Probabilistic Shielding for Model-free Reinforcement Learning

Edwin Hamel-De le Court*

Imperial College

London, United Kingdom

e.hamel-de-le-court@imperial.ac.uk

Gaspard Ohlmann*

Mulhouse, France

gaspard.ohlmann@outlook.com

Francesco Belardinelli

Imperial College

London, United Kingdom

francesco.belardinelli@imperial.ac.uk

ABSTRACT

Safety is a major concern in reinforcement learning (RL): we aim at developing RL systems that not only perform optimally, but are also safe to deploy by providing formal guarantees about their safety. To this end, we introduce Probabilistic Shielding via Risk Augmentation (ProSH), a model-free algorithm for safe reinforcement learning under cost constraints. ProSH augments the Constrained MDP state space with a risk budget and enforces safety by applying a shield to the agent’s policy distribution using a learned cost critic. The shield ensures that all sampled actions remain safe in expectation. We also show that optimality is preserved when the environment is deterministic. Since ProSH is model-free, safety during training depends on the knowledge we have acquired about the environment. We provide a tight upper-bound on the cost in expectation, depending only on the backup-critic accuracy, that is *always* satisfied during training. Under mild, practically achievable assumptions, ProSH guarantees safety even at training time, as shown in the experiments.

KEYWORDS

Safe Reinforcement Learning, Formal Methods, Shielding, Probabilistic Temporal Logic, Stochastic Systems

* *These authors contributed equally to this work.*

1 INTRODUCTION

A key challenge in AI is designing agents that learn to act optimally in unknown environments [37]. Reinforcement Learning (RL) – particularly when combined with deep neural networks – has made impressive strides in domains such as games [27], robotics [23], and autonomous driving [22]. However, ensuring safety during both training and deployment remains a critical barrier to real-world adoption. This has led to a growing interest in Safe RL, where agents must optimize rewards under constraints, e.g., budgeted costs or formal safety requirements. This paper focuses on safe learning in Constrained Markov Decision Processes (CMDPs), where agents maximize expected discounted reward, while keeping cumulative expected cost below a given threshold. Traditional methods, such as Lagrangian approaches [2, 30, 36], allow to converge toward a safe policy, but do not provide formal guarantees on safety, neither during training nor on the provided policy.

To tackle this problem, we introduce ProSH: a novel probabilistic shielding method that ensures safety also at training time, in model-free RL. Unlike classic shielding methods that restrict unsafe actions based on a model of the environment, ProSH operates on distributions, augments the CMDP with a risk budget, and uses a learned cost critic to guide safe exploration. Crucially, ProSH does

not assume access to a simulator or environment abstraction, and is compatible with continuous environments.

We also show that optimizing over shielded policies in the augmented space is sufficient for constrained optimality in the base CMDP in the deterministic setting. We implement ProSH as an off-policy deep RL algorithm and evaluate it on standard Safe RL benchmarks. Notably, ProSH turns out to be safe during training in the experiments. To summarize, the contributions of the paper are as follows.

- (1) We introduce ProSH, a shielding method for CMDP that assumes no knowledge of the environment’s dynamics.
- (2) We provide formal safety guaranties during training, depending only on the accuracy of the learned cost critic.
- (3) We show that optimizing across shielded policies in the Risk-Augmented MDP suffices for constrained optimality (in the deterministic case).
- (4) We implement ProSH as a DRL algorithm and evaluate its optimality and safety on relevant benchmarks. In particular, ProSH leads to significantly less cost violations in expectation, even in early training.

Due to space restrictions, the proofs of all the results appear in the supplementary material, which also includes the code for the experiments.

Related Work. We refer to [16] for a survey on Safe Reinforcement Learning, while here we focus on the works most closely related to our contribution. Policy-based methods are arguably the most popular approaches to Safe RL. They usually consist of extending known RL algorithms (e.g., PPO [33], TRPO [32], or SAC [17]) with safety constraints, either by using a Lagrangian approach [2, 36], changing the objective function [26], or by modifying the update process [46]. Several of these algorithms have become widely used for benchmarking against newer Safe RL methods, and are implemented in state-of-the-art Safe RL frameworks [2, 20, 21].

Shielding restricts the agent’s actions during training and deployment to ensure safety [4, 12, 19, 44]. Introduced in [4] using LTL safety formulas, shielding has been extended to probabilistic settings in [19], although without formal guaranties. Formal safety under probabilistic constraints has been proved in [24]. More recent work allows shielding without prior models: [35] uses a learned safety critic to restrict actions, while [15] combines shielding with a learned dynamics model. A survey on shielding methods is given in [29]. Compared to prior approaches, we focus on the general case of model-free, continuous environments with probabilistic constraints, while prior approaches typically make further assumptions.

Lagrangian approaches [2, 30, 36] convert CMDPs into unconstrained problems by introducing a dual variable to penalize

cost violations in the reward objective. These methods are effective in both discrete and continuous action spaces and are compatible with policy gradient algorithms (e.g. TRPO [32], PPO [33]). Given enough iterations and tuning, they provide good tradeoffs between cost violations and reward, but cannot guaranty safety at any time, which limits their use in safety-critical scenarios.

State augmentation techniques with parameters representing how far the agent is from being unsafe have also been studied. In [9], the MDP is state-augmented with Lagrange multipliers. In Saute-RL [34], the states are augmented with the remaining safety budget, which is used to reshape the objective. We also use an augmentation approach, but we not only augment states but actions too, allowing agents more freedom in the future probabilistic repartition of the remaining safety budget.

Q-learning is a key model-free RL algorithm for discrete environments [42], known to converge in the tabular case under mild assumptions. However, extending it to function approximation—especially with neural networks—introduces challenges [28]. Stabilization techniques such as double Q-learning [40], dueling networks [41], and distributional methods [7] help address instability. Classic Q-learning is prone to overestimation bias [39, 40]. To mitigate these issues, we adopt a TD3-like method [13]. Throughout, we assume the backup Q -value approximates a fixed point of the Bellman operator—an assumption we discuss in the paper.

Trust-region approaches such as CPO [2], FOCOPS [45], and Safe Policy Iteration [31] are On-policy algorithms that rely on providing local safety bounds that hold within a small neighborhood of the current policy. This is useful to bound constraint violations of the next policy updates if the trust region is small, but the accuracy of the bound depends heavily on the size of the trust region. In addition, these methods do not provide a formal guaranty that the policy update step is safe, even late in the training.

Lyapunov-based methods rely on enforcing constraint satisfaction by projecting updates to satisfy a Lyapunov decrease condition defined relative to a fixed safe baseline policy [10, 11]. The guarantees provided in [10, 11] depend on accurate safety critics, local linearizations around the current policy, and a well-behaved backup policy. We rely on restricting the agent’s actions to enforce safety, but we use a state-augmentation technique rather than a fixed Lyapunov function, which yields explicit, model-free training-time bounds that depend only on the backup critic’s approximation error. Furthermore, our backup policy is learned jointly with the main actor and used as a fallback within the shield, rather than as a fixed reference that defines the feasible set of actions.

2 BACKGROUND

In this section we provide the background on Reinforcement Learning (RL) and the notations that will be used in the rest of the paper. This enables us to define the two key problems that we analyse: the Reinforcement Learning Problem (RLP) [38] and the Constrained RLP (CRLP) [1, 14].

2.1 Key Concepts

A **Markov Decision Processes** (MDP) is a tuple $\mathcal{M} = \langle S, A, P, s_i, r \rangle$, where S is a set of *states*, with s_i as initial state¹; A is a mapping that associates every state $s \in S$ to a nonempty finite set $A(s)$ of *actions*; for $\mathcal{SA} = \{(s, a), a \in A(s)\}$, $P : \mathcal{SA} \rightarrow \mathbb{D}(S)$ is a *transition probability function* that maps every state-action pair $(s, a) \in \mathcal{SA}$ to a probability distribution over S ; and $r : \mathcal{SA} \mapsto \mathbb{R}$ is the *reward function*. An MDP is finite if the sets of states and actions are finite. Finally, a *Constrained MDP* (CMDP) is an MDP additionally equipped with a *cost function* $c : \mathcal{SA} \rightarrow \mathbb{R}_+$.

Paths. A finite (resp. infinite) *path* is a finite (resp. infinite) word $\zeta = s_0 a_0 \dots s_{n-1} a_{n-1} s_n$ (resp. $\zeta = s_0 a_0 \dots s_n a_n \dots$), such that for every $k \leq n$ (resp. for every k), s_{k-} is a state in S , a_{k-1} is an action in $A(s_{k-1})$, and s_k is in the support of $P(s_{k-1}, a_{k-1})$.

Policies. A *policy* π is a mapping that associates every finite path ζ to an action of the probability distribution $\mathbb{D}(A(\text{last}(\zeta)))$. A policy is *memoryless* if $\pi(\zeta)$ only depends on $\text{last}(\zeta)$; it is *deterministic* if for any finite path ζ , $\pi(\zeta)$ is a Dirac distribution; it is *flipping* if for any finite path ζ , $\pi(\zeta)$ is a mixture of two Dirac distributions. Throughout the paper, we let $\sum_{a \in A(s)} \lambda_a a$ denote the probability distribution corresponding to sampling action a with probability λ_a . As an example, if π_1 and π_2 are two deterministic policies on $\mathcal{M}$, the policy defined for any $s \in S$ as $\pi(s) = 0.5\pi_1(s) + 0.5\pi_2(s)$ denotes the flipping policy taking in s the actions $\pi_1(s)$ and $\pi_2(s)$ with equal probability.

For any policy π of $\mathcal{M}$ and state $s \in S$, let $\mathcal{M}_\pi^s$ be the *Markov chain* induced by π in $\mathcal{M}$ starting from state s (c.f. [6]). let $\mathcal{M}_\pi$ denote $\mathcal{M}_\pi^{s_i}$ and P_π the transition function of $\mathcal{M}_\pi$. We denote the usual probability measure induced by the Markov chain $\mathcal{M}_\pi^s$ on Paths($\mathcal{M}$) by $\text{prob}_{\mathcal{M}, \pi}^s$. We refer to [6, 8] for full details.

Discounted expectations. For any random variable $X : \text{Paths}(\mathcal{M}) \mapsto \mathbb{R}$, let $\mathbb{E}_{\mathcal{M}, \pi}^s(X)$ be the expectation of X w.r.t. the probability distribution $\text{prob}_{\mathcal{M}, \pi}^s$, and let $\mathbb{E}_{\mathcal{M}, \pi}(X)$ denote $\mathbb{E}_{\mathcal{M}, \pi}^{s_i}(X)$. Given a path $\zeta = s_0 a_0 \dots s_n a_n \dots$, we denote the *discounted cumulative reward* along ζ as $\mathcal{R}(\zeta) = \sum_{t \in \mathbb{N}} \gamma_r^t r(s_t, a_t)$, and its *discounted cumulative cost* as $\mathcal{C}(\zeta) = \sum_{t \in \mathbb{N}} \gamma_c^t c(s_t, a_t)$, where γ_r and γ_c are the discount factors for reward r and cost c respectively. For any policy π if a CMDP $\mathcal{M}$, we denote the **discounted cumulative reward** of π as

$$\mathcal{R}_{\mathcal{M}}^s(\pi) = \mathbb{E}_{\zeta \sim \pi} \mathcal{R}^s(\zeta) = \mathbb{E}_{(s_t, a_t)_{t \sim \pi, s_0 = s}} \sum_{t \in \mathbb{N}} \gamma_r^t r(s_t, a_t),$$

and the **discounted cumulative cost** of π as

$$\mathcal{C}_{\mathcal{M}}^s(\pi) = \mathbb{E}_{\zeta \sim \pi} \mathcal{C}^s(\zeta) = \mathbb{E}_{(s_t, a_t)_{t \sim \pi, s_0 = s}} \sum_{t \in \mathbb{N}} \gamma_c^t c(s_t, a_t).$$

We let $\mathcal{R}_{\mathcal{M}}(\pi) = \mathcal{R}_{\mathcal{M}}^{s_i}(\pi)$ and $\mathcal{C}_{\mathcal{M}}(\pi) = \mathcal{C}_{\mathcal{M}}^{s_i}(\pi)$, and omit $\mathcal{M}$ when there is no ambiguity.

Q-values are functions $Q : S \times \mathcal{A} \rightarrow \mathbb{R}$ that estimates the expected cost of taking action a in state s , and following some policy thereafter. In our context, we are especially interested in

¹This can be assumed wlog compared to a model with an *initial probability distribution* since it is always possible to add a new initial state to such a model with an action from this initial state whose associated probability distribution is the aforementioned initial probability distribution.

Q -values approximating the *minimal discounted cost*, defined as

$$Q_b^*(s, a) := \min_{\pi: \pi(s)=a} C_M^s(\pi).$$

We denote by Q_b any approximation of Q_b^* , and say it is an ε -approximation if $\|Q_b - Q_b^*\|_\infty \leq \varepsilon$.

2.2 Reinforcement Learning Problems

This work is motivated by two central optimization problems in RL. The first, classic Reinforcement Learning problem (RLP), focuses on optimizing the expected reward [37]. The second, constrained RLP (CRLP), introduces explicit safety requirements in the form of cost constraints [1, 14]. Our method PROSH is designed specifically for CRLP, and aims to provide provably safe probabilistic guarantees even in the absence of a known model.

DEFINITION 1 (RL PROBLEMS). Let $\mathcal{M}$ be an MDP, γ_r the reward discount factor, and Π the set of policies over $\mathcal{M}$.

$$(RLP) : \text{Find policy } \pi^* \in \Pi \text{ such that } \mathcal{R}(\pi^*) = \max_{\pi \in \Pi} \mathcal{R}(\pi)$$

Further, let $\mathcal{M}'$ be a CMDP, γ_r and γ_c the discount factors for rewards and costs respectively, and $d > 0$ be a cost threshold.

Denote $\Pi_{\leq d}$ the set of policies π such that $C(\pi) \leq d$.

$$(CRLP) : \text{Find policy } \pi^* \in \Pi_{\leq d} \text{ such that } \mathcal{R}(\pi^*) = \max_{\pi \in \Pi_{\leq d}} \mathcal{R}(\pi)$$

Notice that Definition 1 is not restricted to memoryless policies, as is usually the case [1, 2, 46]. This is because in the case of two different discount factors for the reward and the cost, optimal memoryless policies for CRLP are not guaranteed to exist [5]. Furthermore, in practical implementations, the cost discount factor is often higher than the reward discount factor since constraint satisfaction is paramount in CRLP. The additional difficulty does not break the optimality of the method we provide in the deterministic case.

Shielding and State-Augmentation challenges: a motivation example. We illustrate the limitations of traditional shielding techniques through the simple CMDP $\mathcal{M}_1$ shown in Fig. 1, with $\gamma_r = \gamma_c = 1$ and cost threshold $d = 0.5$. The goal is to solve the constrained RL problem (CRLP) for $\mathcal{M}_1$:

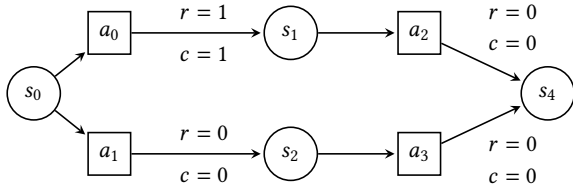


Figure 1: CMDP $\mathcal{M}_1$ with cost threshold $d = 0.5$ and $\gamma_r = \gamma_c = 1$.

The optimal safe policy π^* takes action a_0 and a_1 with equal probability, yielding a cost $C(\pi^*) = 0.5$ and a reward $\mathcal{R}(\pi^*) = 0.5$. This solution cannot be found using hard safety enforcement such as classic shielding [4], which would block a_0 . Nor does a state-augmentation based approach such as [34], as path $s_0 a_0 s_1 a_2 s_4$ has a cumulative cost above the threshold. We claim that *achieving safe optimality requires reasoning over expectations*.

3 PROSH: THEORETICAL FOUNDATIONS

We now present the theoretical backbone of ProSh, a scalable shielding mechanism for CRLP that provides formal safety guarantees. Our approach uses *risk-augmentation*, similarly to [24], whereby each state is paired with a risk budget representing the expected discounted cost that the agent is allowed to incur from that point onward. Unlike traditional shielding approaches, PROSH considers probabilistic expectations that the main actor chooses, and enforces these two structural properties:

- **Approximate consistence.** The combined risk after one step is, in expectancy, approximately equal to the risk of the current step.
- **Approximate realizability.** For every augmented state, there exists a policy whose expected cumulative cost is approximately less than or equal to the current risk.

Our goal is to construct policies that satisfy the safety constraint *throughout training and deployment*, even in the absence of a known model. To this end, we augment the CMDP with a dynamically updated notion of risk.

3.1 Risk-Augmented CMDPs

Hereafter, for any Q -value Q_b of a CMDP $\mathcal{M}$, for any state s and action $a \in A(s)$, we let $Q_b(s)$ denote $\gamma_c \min_{a \in A(s)} (Q_b(s, a))$.

DEFINITION 2 (RISK-AUGMENTED CMDP). For any CMDP $\mathcal{M} = (S, A, s_i, P, R, C, d, \gamma_r, \gamma_c)$ with non-negative costs, maximal cost $c_{\max} = \|c\|_{L^\infty(A, S)}$, and Q -value Q_b , we define $\overline{\mathcal{M}}_d^{Q_b}$ as the MDP with $C_{\max} = \frac{c_{\max}}{(1-\gamma_c)}$, such that

- **States :** $\overline{S} = \{(s, x) \mid s \in S, x \in [-C_{\max}; C_{\max}]\}$, with initial state $\overline{s}_i = (s_i, d)$;
- **Actions :** $\overline{A}(s, x) = \{(a, y) \mid a \in A(s), y \in [-C_{\max}; C_{\max}]\}$;
- **Rewards :** $\overline{r}((s, x), (a, y)) = r(s, a)$;
- **Costs :** $\overline{c}((s, x), (a, y)) = c(s, a)$;
- **Transition Probability Function $\overline{P}$:** for $\overline{s} = (s, x)$, $\overline{s}' = (s', x')$ in $\overline{S}$, and $\overline{a} = (a, y) \in \overline{A}(\overline{s})$, we have

$$\overline{P}(\overline{s}, \overline{a}, \overline{s}') = \begin{cases} 0 & \text{if } x' \neq y - Q_b(s, a) + Q_b(s') \\ P(s, a, s') & \text{if } x' = y - Q_b(s, a) + Q_b(s') \end{cases}$$

The risk-augmented CMDP introduces a new coordinate for states and actions: the *risk*. The policy can then choose any risk for its next actions, and the transition function $\overline{P}$ simply ensures that the corresponding risk is spread coherently to each state in the non-deterministic case. We will introduce later the conditions on the policy to ensure that the second parameter x indeed plays the role of a *running budget*, recording how much cumulative discounted cost the agent is allowed to incur.

We now define a subset of policies on $\overline{\mathcal{M}}$, the *valued policies*.

DEFINITION 3 (VALUED POLICY). A policy $\overline{\pi}$ on the risk-augmented CMDP $\overline{\mathcal{M}}$ is valued if there exists a mapping $y_{\overline{\pi}}: \overline{S} \rightarrow (A(s) \rightarrow \mathbb{R})$ such that, for every augmented state $(s, x) \in \overline{S}$, we can write with the notation $(a, x)_\delta = \delta_{a, x}$, the Dirac Delta distribution choosing the action (a, x) with probability 1,

$$\overline{\pi}(s, x) = \sum_{a \in A(s)} P_{\overline{\pi}}(a \mid s, x) (a, y_{\overline{\pi}}(s, x)(a))_\delta$$

In other words, to any underlying state-risk pair (s, x) , the policy first assigns a risk value $r_{\bar{\pi}}(s, x)(a)$ to every available action $a \in A(s)$.

We call $y_{\bar{\pi}}$ a valuation function, and we let $\bar{\Pi}_{val}$ denote the set of all valued policies.

Asking the agent to attach a risk value to every possible action is not as restrictive as it looks. Indeed, we will show that considering only the set of valued policies is sufficient for optimality.

3.2 Shielded Policies, Safety and Optimality

Among valued policies, we focus on those that keep the risk budget synchronized with the CMDP dynamics. We call those the Q_b -shielded policies. In the rest of the paper, for any Q -value Q_b , we let $\pi_b(Q_b)(s) = \arg \min_{a \in A(s)} Q_b(s, a)$, and when there is no ambiguity, we write only π_b for $\pi_b(Q_b)$. This action is the safest according to the estimation $Q_b(s, a)$ and will be used to decrease the expected cost when necessary.

DEFINITION 4 (Q_b -SHIELDED POLICIES). Let $\mathcal{M}$ be a CMDP, and let $Q_b(\mathcal{M})$ be any Q -valued function defined on $\mathcal{AS}$. Denote $\bar{\mathcal{M}}$ the corresponding augmented CMDP. A policy $\bar{\pi} \in \bar{\Pi}_{val}$ is said to be Q_b -shielded if for any $(s, x) \in \bar{\mathcal{S}}$, if $x \geq Q_b(s)$, there exist λ and $\{y_a\}_{a \in A(s)}$, with $y_a \geq Q_b(s, a)$ for all $a \in A(s)$, such that

$$\begin{cases} \bar{\pi}(s, x) = (1 - \lambda) \sum_a P_{\bar{\pi}}((a, y_a) \mid (s, x))(a, y_a)_{\delta} \\ \quad + \lambda(\pi_b(s), Q_b(s, \pi_b(s)))_{\delta}, \\ x \geq \gamma_c(1 - \lambda) \sum_a P_{\bar{\pi}}((a, y_a) \mid (s, x))y_a + \gamma_c \lambda Q_b(s, \pi_b(s)), \end{cases}$$

and if $x < Q_b(s)$, there exists $z \leq x$ such that

$$\begin{cases} \bar{\pi}(s, x) = (\pi_b(s), z), \\ x \geq \gamma_c z \end{cases}$$

Intuitively, the policies are shielded when they always allow a budget for the actions larger than the estimated minimal cost of taking this action. Moreover, the total budget allowed for the actions is, in expectancy, smaller than the current budget.

We now define the corresponding Shield-Map, which takes any memoryless valued policy $\bar{\pi}$ of $\bar{\mathcal{M}}$ and outputs a Q_b -shielded policy.

DEFINITION 5 (SHIELD-MAP). Let $\bar{\pi}$ be a memoryless valued policy of the augmented MDP $\bar{\mathcal{M}}$. We define the shield-map Ξ such that, for any state (s, x) of $\mathcal{M}$, if we let $\tilde{y}_a = \max(\tilde{y}_a, Q_b(s, a))$, and

$$\tilde{\pi}(s, x) = \sum_{a \in A(s)} P_{\bar{\pi}}(a \mid s, x)(a, \tilde{y}_a)_{\delta}, \quad t = \gamma_c \sum_{a \in A(s)} P_{\bar{\pi}}(a \mid s, x)\tilde{y}_a,$$

we have

$$\Xi(\bar{\pi})(s, x) = \begin{cases} \tilde{\pi}(s, x) & \text{if } t \leq x \\ (\pi_b(s), x/\gamma_c)_{\delta} & \text{if } x < Q_b(s) \\ (1 - \lambda)\tilde{\pi}(s, x)_{\delta} + \lambda(\pi_b(s), Q_b(s, \pi_b(s)))_{\delta} & \text{otherwise,} \end{cases}$$

where

$$\lambda = \frac{\gamma_c t - x}{\gamma_c t - \gamma_c Q_b(s, \pi_b(s))}.$$

Intuitively, the shield map blends the original policy with the estimated safest action π_b so that the resulting policy always satisfies the Q_b -shielded condition. When the policy already respects this safety constraint, no modification is applied. If estimation errors

lead to a state where any distribution would exceed the available budget, the shield falls back to the safest estimated action π_b and allocates the entire budget to it.

Because the constraint in the definition mirrors the Bellman equation for discounted costs, the risk x is indeed an approximate upper bound of the *expected future cost* of the shielded policy that starts from the augmented state (s, x) . The next theorem makes this intuition precise.

THEOREM 1 (SAFETY BOUNDS FOR Q_b -SHIELDS). Let $\mathcal{M}$ be a CMDP with cost discount factor γ_c , and let Q_b^* be its optimal state-action cost function. Assume that Q_b is a Q -value, and let $\Delta_b = \|Q_b - Q_b^*\|_{L^\infty(\mathcal{SA})}$.

For any policy $\bar{\pi} \in \bar{\Pi}_{val}$ that is Q_b -shielded, the expected discounted cost from the augmented state (s_0, x_0) satisfies

$$C^{(s_0, x_0)}(\bar{\pi}) \leq x_0 + \frac{2\Delta_b}{1 - \gamma_c}, \quad \text{whenever } x_0 \geq \frac{Q_b(s_0)}{\gamma_c}.$$

Theorem 1 shows that an ε -accurate Q_b is enough to keep any Q_b -shielded policy inside the budget, up to a small additive slack. Moreover, the policy we consider in the augmented CMDP corresponds to policies in the original CMDP that have the same cost and reward, as stated in the next results.

DEFINITION 6 (PROJECTION ONTO THE BASE CMDP). For any Q_b -shielded policy $\bar{\pi} \in \bar{\Pi}_{val}$ of $\bar{\mathcal{M}}$, we define the backward projection $T(\bar{\pi})$ as the memoryful policy tracking a single scalar variable m as

- In state s and with current memory $m - x$, sample $(a, x') \sim \bar{\pi}(s, x)$ and execute a_t in $\mathcal{M}$.
- As state s' is reached in $\mathcal{M}$, update the memory $m \leftarrow x' - Q_b(s, a) + Q_b(s')$.

We can now prove the following preservation result.

THEOREM 2 (PRESERVATION). The cost and reward of $T(\bar{\pi})$ in Def. 6 satisfy

$$C_{\mathcal{M}}(T(\bar{\pi})) = C_{\bar{\mathcal{M}}}(\bar{\pi}), \quad \mathcal{R}_{\mathcal{M}}(T(\bar{\pi})) = \mathcal{R}_{\bar{\mathcal{M}}}(\bar{\pi}).$$

Theorem 2 states that a Q_b -shielded policy keeps its cost and reward when mapped back to the original CMDP. This allows us to transfer our safety results from $\bar{\mathcal{M}}$ to $\mathcal{M}$. It remains to show that some Q_b -shielded policy attains the constrained optimum. The next theorem precisely answers this question. Recall that flipping policies associate to every state a mixture of two Dirac distributions.

THEOREM 3 (OPTIMALITY OF THE SHIELDED POLICIES). Let $\mathcal{M}$ be a deterministic CMDP with safety threshold d and initial state s_i , Q_b be a Q -value, $\Delta_b = \|Q_b - Q_b^*\|_{L^\infty(\mathcal{SA})}$.

Further, let Π the set of all policies of $\mathcal{M}$, $\bar{\Pi}_{sh}$ be the set of shielded policies of $\bar{\mathcal{M}}$, and $\bar{\Pi}^f$ be the set of valued flipping policies.

The, we have the following for $\mathcal{E} = \frac{2\Delta_b\gamma_c}{1 - \gamma_c}$:

$$\begin{aligned} \max_{\pi \in \Pi, C(\pi) \leq d} \mathcal{R}(\pi) &\leq \max_{\bar{\pi} \in \bar{\Pi}^f, x_0 \leq \gamma_c d + \mathcal{E}} \mathcal{R}^{(s_i, x_0)}(\bar{\pi}), \\ \max_{\bar{\pi} \in \bar{\Pi}^f, x_0 \leq \gamma_c d + \mathcal{E}} \mathcal{R}^{(s_i, x_0)}(\bar{\pi}) &\leq \max_{\pi \in \bar{\Pi}_{sh}, x_0 \leq \gamma_c d + \mathcal{E}} \mathcal{R}^{(s_i, x_0)}(\bar{\pi}), \end{aligned}$$

We now summarize the results obtained in this section, which represent the theoretical backbone of ProSh.

From constrained to unconstrained optimisation. Let Π_{val} the set of valued policies (potentially unsafe) of the augmented MDP, and $\Pi_{\Xi} = \Xi(\Pi_{val})$, then for $\Delta_b = \|Q_b - Q_b^*\|_{L^\infty(\mathcal{S}\mathcal{A})}$,

- The set of shielded policies Π_{Ξ} , starting with a risk up to $x_0 = d + 2\Delta_b$ is enough to reach optimality, i.e.,

$$\max_{\bar{\pi} \in \Pi, x_0 \leq d + 2\Delta_b} \mathcal{R}^{(s_i, x_0)}(\Xi(\bar{\pi})) \geq \max_{\pi \in \Pi} \mathcal{R}(\pi).$$

- Any shielded policy $\Xi(\bar{\pi}) \in \Pi_{\Xi}$ starting with a risk up to $x_0 = d + 2\Delta_b$ has a cost satisfying

$$C^{(s_i, x_0)}(\Xi(\bar{\pi})) \leq d + 2\Delta_b + \frac{\Delta_b}{1 - \gamma_c}.$$

Hence, we can optimize among all policies of the augmented MDP and apply the shield. Alternatively, one can choose a dynamic $x_0 < d$ to ensure safety during training. This can be combined with any approach ensuring $\Delta_b \rightarrow 0$, or making it sufficiently small.

4 SAFE RL THROUGH PROBABILISTIC SHIELDING

In this section, we show how the theoretical foundations in Section 3 are used to derive learning algorithms for CRLP. We first present a general approach and derive the corresponding safety guarantees. We then propose our implementation of the ProSH algorithm.

4.1 Learning Algorithms

In this section we present a minimal version of the training loop for the main actor in Algorithm 1. Then, in Sec. 4.2 we provide a complete implementation. ProSH behaves like "classical" shielding [4], but the shield acts on distribution rather than actions. The shield Ξ uses the values provided by the critic Q_b and combines it with the backup action to obtain a safe distribution.

Algorithm 1 ProSH: Main Actor only

```

1: Input: cost budget  $d$ , margin  $\delta$ , discount  $\gamma_c$ 
2: Initialise main actor  $\bar{\pi}_r^\psi$ , backup critic  $Q_b^\theta$ 
3: for each episode do
4:    $s \leftarrow \bar{s}_i, x \leftarrow d - \delta$ 
5:   while not done do
6:     Shield:  $\mu_{\text{safe}} \leftarrow \Xi_{Q_b}(\bar{\pi}_r^\psi)(s, x)$ 
7:     Sample  $(a, y) \sim \mu_{\text{safe}}$ 
8:     Execute  $(a, y)$ , observe  $(s', x')$ , reward  $r$ , cost  $c$ 
9:     Store transition  $((s, x), (a, y), r, c, (s', r'))$ 
10:     $s \leftarrow s', r \leftarrow r'$ 
11:   end while
12:   Update:
13:     • Update  $\pi_\theta$  using shielded distributions from memory
14: return projected policy  $\bar{\pi}_r^\psi$  (via Thm. 2)
```

The shield precedes sampling, so the exploration is budget-safe. The choice of the RL update for θ is open (e.g., PG, PPO, A2C). Additionally, the backup critic Q_b can be learned, either in parallel, off-policy, or even pre-computed.

The previous results in Sec. 3 allows us to derive the following key safety guarantees for Algorithm 1.

THEOREM 4 (SAFETY AND NEAR-OPTIMALITY). (i) *At any step of the algorithm, the output policy $\bar{\pi}$ satisfies*

$$C_{\mathcal{M}}(\bar{\pi}) \leq r_0 + \frac{2\gamma_c \Delta_b}{1 - \gamma_c}, \quad \Delta_c = \|Q_b - Q_b^*\|_\infty$$

(ii) *The algorithm is asymptotically optimal in deterministic environments as $\Delta_b \rightarrow 0$. More precisely, let $\mathcal{M}$ be a constrained deterministic environment modeled by a CMDP, and for any $\eta > 0$ and Δ_b small enough, there exists $\bar{\pi} \in \Xi_{sh}$ with cost at most $\gamma_c d + \mathcal{E}$ for $\mathcal{E} = \frac{2\Delta_b \gamma_c}{1 - \gamma_c}$, such that*

$$\mathcal{R}(\bar{\pi}) \geq \max_{\pi \in \Pi} \mathcal{R}(\pi) - \eta$$

Since our guarantees involve the difference Δ_b and the assumption $\Delta_b \rightarrow 0$, we discuss how restrictive this assumption is.

REMARK 1 (ON ASSUMPTION $\Delta_b \rightarrow 0$). *We present several cases in which $\Delta_b \rightarrow 0$.*

- *Tabular update:* With exact Q -learning and sufficient exploration, the classical result of [42] gives $\Delta_b \xrightarrow{a.s.} 0$.
- *Neural networks:* Recent results [3, 43] show that over-parameterised deep Q -learning converges to the Bellman fixed point when the optimiser drives training error to zero and the network remains within a neighbourhood of its initialization.
- *Batch (fitted) Q -evaluation:* In the agnostic setting, fitted Q -iteration with minimax regression loss enjoys finite-sample $O(1/\sqrt{n})$ guarantees [25], so Δ_b vanishes as the data set grows.

Hence, in all cases above, the safety slack $\kappa(\Delta_b) = \gamma_c \Delta_b / (1 - \gamma_c)$ converges to zero, ProSH is safe up to a controlled vanishing coefficient, and approaches true constrained optimality.

4.2 ProSH-TD3

In this section we introduce ProSH-TD3, an implementation of ProSH based on TD3 [13], for continuous spaces. Hereafter we describe the four key components of ProSH-TD3: the network architecture, the implementation of the shield, the actor/critic pairs training, and the exploration strategy.

Networks Architecture. The backup actor-critic pair follows the TD3 architecture. The backup critic has two output heads, $Q_b^{\theta_1}$ and $Q_b^{\theta_2}$, and takes as input a state-actor pair of the original environment $\mathcal{M}$. The backup actor π_b^ϕ is deterministic, and takes as input a state in $\mathcal{M}$ and outputs an action. The main actor-critic pair also follows the TD3 architecture. The main critic also has two heads, $\bar{Q}_r^{\xi_1}$ and $\bar{Q}_r^{\xi_2}$, but takes as input a state-action pair of the augmented environment $\bar{\mathcal{M}}$. The main actor $\bar{\pi}_r^\psi$ takes as input a state in the augmented environment and outputs a flipping policy in the augmented environment. Furthermore, we also maintain *target* networks $Q_b^{\theta_1^{arg}}$ and $Q_b^{\theta_2^{arg}}$, etc, which are Polyak averages of the original networks, that we use for computing training targets, in order to stabilize training.

Shield Implementation. We implement the shield using the Shield-map in Def. 5 as a fully differentiable layer after the output of the main actor. More precisely, if $\bar{s} = (s, x)$ and $\bar{\pi}_r^\psi(\bar{s}) = \rho(a_1, y'_1) + (1 - \rho)(a_2, y'_2)$, we implement Ξ as

$$\Xi(\bar{\pi}^\psi)(\bar{s}) = (1 - \lambda)\rho(a_1, y'_1) + (1 - \lambda)(1 - \rho)(a_2, y'_2) + \lambda \bar{a}_3$$

where $y'_1 = \min(y_1, Q_b^{\theta_1}(s, a_1))$, $y'_2 = \min(y_2, Q_b^{\theta_1}(s, a_2))$, $\bar{a}_3 = (\pi_r^\psi(\bar{s}), Q_b^{\theta_1}(s, \pi_b^\phi(s)))$, and

$$\lambda = \text{clip} \left(\frac{\rho y'_1 + (1 - \rho) y'_2 - x}{\text{relu}(\rho y'_1 + (1 - \rho) y'_2 - Q_b^{\theta_1}(s, \pi_b(s))) + \eta}, 0, 1 \right)$$

where η is chosen sufficiently small ($\sim 10^{-8}$).

Actor-critic Pairs Training. We train the actor-critic pairs with batch sampling from a replay buffer in which we store every transition made in the augmented CMDP. For the backup critic, we use the training target $y_b(c, s')$, proposed in [18], equal to

$$c + \gamma_c \left[\beta \min_{j \in \{1,2\}} Q_b^{\theta_j^{arg}}(s', \pi_b^{\psi^{arg}}(s') + \epsilon) + \frac{1 - \beta}{2} \sum_{j \in \{1,2\}} Q_b^{\theta_j^{arg}}(s', \pi_b^{\psi^{arg}}(s') + \epsilon) \right]$$

where ϵ is a small gaussian noise and β is a hyperparameter. Compared to the standard TD3 training target, this target helps combat the underestimation bias of TD3, improving the backup critic accuracy. The backup actor and the main critic are updated with the standard TD3 training losses. The main actor is updated taking into account the shield:

$$\mathcal{L}(\psi) = -\mathbb{E}_{\bar{s} \sim \mathcal{D}} \left[\sum_{k \in \{1,2,3\}} \lambda_k \bar{Q}_r^{\xi_1}(\bar{s}, \bar{a}_k) \right], \text{ where}$$

$$\Xi(\pi_r^\psi)(\bar{s}) = \sum_{k \in \{1,2,3\}} \lambda_k \bar{a}_k.$$

Note that the gradients can flow through λ_k and $\bar{a}_k$ in the main actor training loss as they are computed in a fully differentiable way.

Exploration. For a better estimation of the backup critic, we alternate between *main* episodes and *hybrid* episodes. In the *main* episodes, we sample from the (shielded) main actor. In the *hybrid* episodes, we explore using the shielded main actor until a certain step, after which we explore using the backup actor. All the actions of the backup actor are converted to risk-augmented actions by associating the risk of the current observed state. The corresponding backup actor is always Q_b -shielded by construction, so the sampling is safe during training.

5 EXPERIMENTAL EVALUATION

5.1 Experimental setup

We implement ProSH-TD3 using the Stable-Baselines 3 implementation of TD3². We evaluate our implementation of ProSH-TD3 on six environments of the Safety Gymnasium benchmark suite: SafetyHalfCheetahVelocity, SafetyHopperVelocity, SafetyPointCircle, SafetyPointGoal, SafetyCarCircle, and SafetyCarGoal are from the Navigation Suite. More details on the environments are provided in the comments on the experimental results. We benchmark against diverse Constrained RL algorithms, two of them being Off-Policy

Algorithms (TD3-Lagrangian and PID-Lagrangian), and three being On-Policy Algorithms (CPO, FOCOPS, PPO-Saute). Some algorithms are safer, such as PPO-Saute, while others impose less restrictions and seek higher rewards, such as TD3-Lagrangian. We included them despite not always being directly comparable – PPO-Saute being in most cases the only algorithm that is safe during training – to present the overall performances of ProSH against states of the art algorithms, both in terms of safety and rewards. On each environment, the algorithms are run using three different seeds, and the mean and standard deviation are computed. Finally, the algorithms were implemented in the Omnisafe infrastructure [21] and use the hyperparameters provided by Omnisafe for each environment.

5.2 Experimental Results

Overall Performance. Across all environments, ProSH maintains a high level of safety, with only occasional low-magnitude violations. PPO-Saute also achieves consistent safety, although this often coincides with reduced performance and lower rewards. The Off-Policy algorithms TD3-Lagrangian and PID-TD3 tend to exhibit unsafe behavior in most settings, while the On-Policy methods FOCOPS and CPO show less stable safety profiles, with costs that fluctuate even after extended training. Taken together, the results suggest that ProSH provides the best trade-off between safety and performance in the evaluated environments, especially when ensuring safety both at training and deployment time is a necessity.

The Velocity Suite. In SafetyHopperVelocity and SafetyHalfCheetahVelocity, the agent has to move as quickly as possible, while adhering to a velocity constraint.

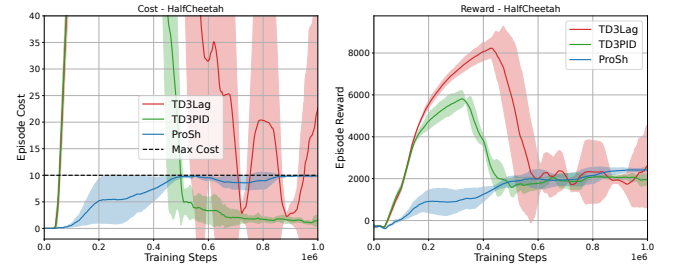


Figure 2: Comparison with Off-Policy Algorithms on Half Cheetah. Cost threshold $d = 10$.

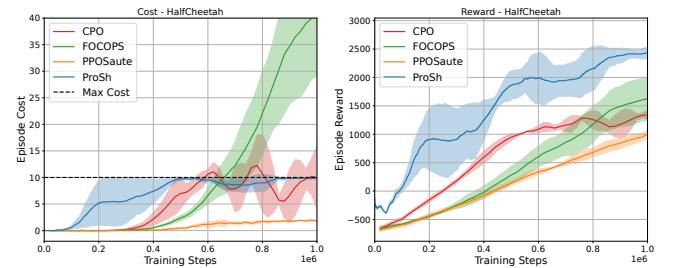


Figure 3: Comparison with On-Policy Algorithms on Half Cheetah. Cost threshold $d = 10$.

²<https://github.com/DLR-RM/stable-baselines3>

Experimental Results: Safety Half Cheetah. PROSH achieves absolute safety and stays under the cost constraint at all times. PID-TD3 displays comparable performance. TD3-Lag fails to be safe and does not achieve higher reward. PROSH displays higher reward than all the on-policy algorithms (CPO, FOCOPS, PPOsaute). Finally, FOCOPS fails to be safe and CPO violates the constraint after several steps.

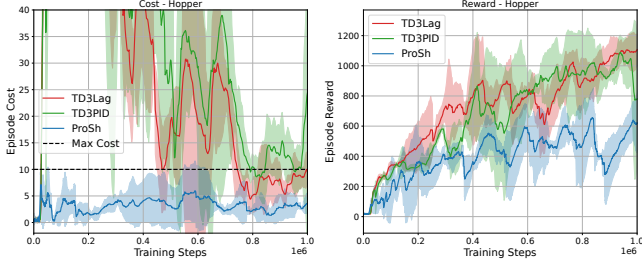


Figure 4: Comparison with Off-Policy Algorithms on Hopper. Cost threshold $d = 10$.

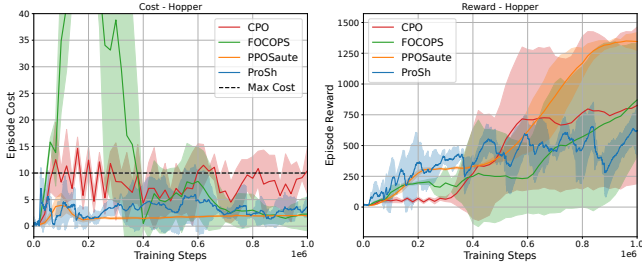


Figure 5: Comparison with On-Policy Algorithms on Hopper. Cost threshold $d = 10$.

Experimental Results: Safety Hopper. PROSH achieves absolute safety and stays under the cost constraint at all times. The off-policy algorithms (TD3-Lag, PID-TD3) fail to be safe. FOCOPS and CPO also fail to be safe (including the standard deviation), while having comparable reward to PROSH. PPO-Saute is safe and has the highest reward on this environment.

The Navigation Suite. We consider four environments: SafetyCarCircle, SafetyCarGoal, SafetyPointCircle, and SafetyPointGoal. The agent controls either a simple robot (Point), which is constrained to the 2D plane, with one actuator for turning and another for moving forward or backward; or a slightly more complicated robot (Car), which has two independently-driven parallel wheels and a free-rolling rear wheel that require coordination to properly navigate. Its goal is either to circle around the center of a circular area while avoiding going outside the boundaries (Circle), or to navigate to the Goal's location while circumventing Hazards (Goal).

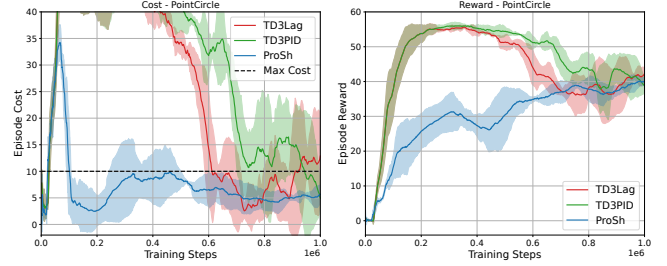


Figure 6: Comparison with Off-Policy Algorithms on Point Circle. Cost threshold $d = 10$.

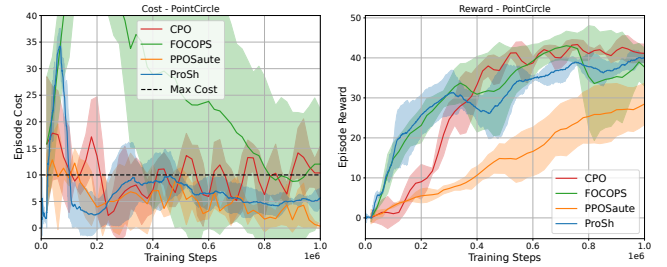


Figure 7: Comparison with On-Policy Algorithms on Point Circle. Cost threshold $d = 10$.

Experimental Results: Safety Point Circle. PROSH achieves absolute safety and stays under the cost constraint at all times after a few steps. The off-policy algorithms (TD3-Lag, PID-TD3) fail to be safe and do not achieve higher rewards. FOCOPS and CPO also fail to be safe, while having comparable reward to PROSH. PPO-Saute is the only other safe algorithm, but it has lower performance.

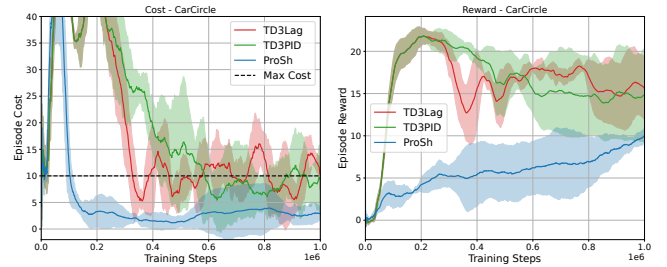


Figure 8: Comparison with Off-Policy Algorithms on Car Circle. Cost threshold $d = 10$.

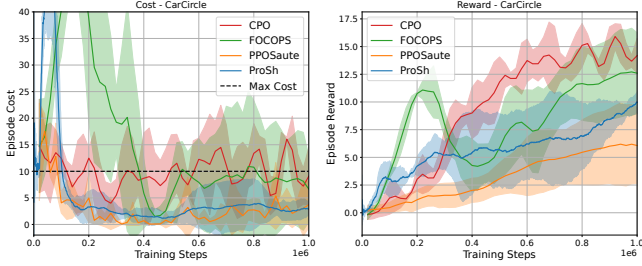


Figure 9: Comparison with On-Policy Algorithms on Car Circle. Cost threshold $d = 10$.

Experimental Results: Safety Car Circle. PROSH achieves absolute safety and stays under the cost constraint at all times after a few steps. The off-policy algorithms (TD3-Lag, PID-TD3) are not safe. FOCOPS and CPO are also not safe and have very slightly greater rewards compared to ProSh. PPO-Saute is the only other safe algorithm, but has lower performance.

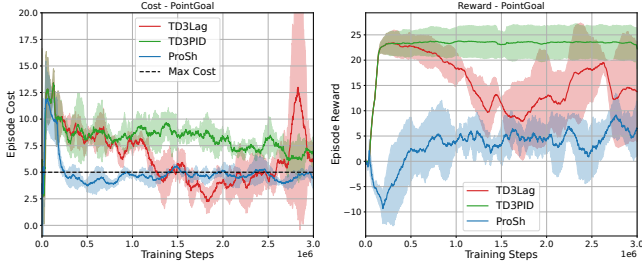


Figure 10: Comparison with Off-Policy Algorithms on Point Goal. Cost threshold $d = 5$.

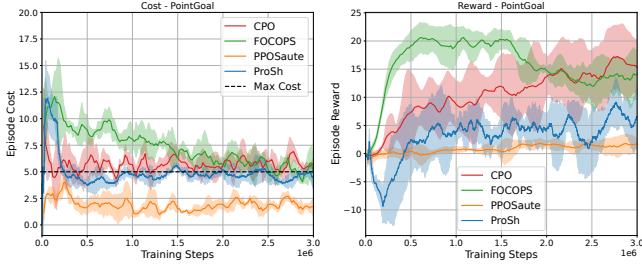


Figure 11: Comparison with On-Policy Algorithms on Point Goal. Cost threshold $d = 5$.

Experimental Results: Safety Point Goal. Every algorithm is unsafe except PPO-Saute, that only achieves a tiny reward. PROSH is much safer than the other Algorithms, with a cost that evolves smoothly below the limit on average. Off-policy algorithms largely exceed the cost threshold, while on-policy algorithms stay closer to the threshold but violate the constraint even after several steps. In this environment, the higher safety of PROSH comes at the cost of diminished reward compared to CPO and FOCOPS.

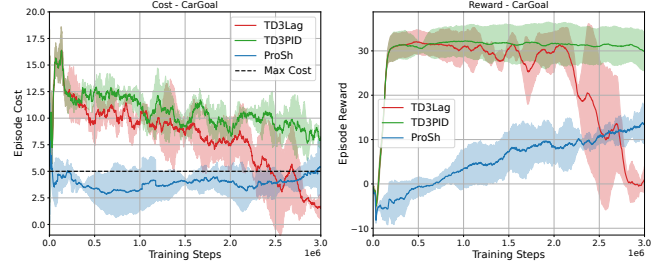


Figure 12: Comparison with Off-Policy Algorithms on Car Goal. Cost threshold $d = 5$.

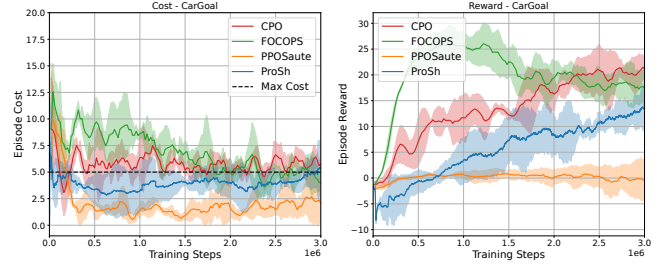


Figure 13: Comparison with On-Policy Algorithms on Car Goal. Cost threshold $d = 5$.

Experimental Results: Safety Car Goal. PPO-Saute is always safe but only achieves a tiny reward. Among the others, PROSH is the safest algorithm, violating only the cost limit if the standard deviation is taken into account. The two off-policy algorithms TD3-Lag and TD3-PID are completely unsafe, while the two on-policy algorithms CPO and FOCOPS are unsafe even after many steps. Their rewards is slightly higher than PROSH, which showcases the best tradeoff between safety and reward.

6 CONCLUSION

We introduced PROSH, a model-free algorithm for safe reinforcement learning based on probabilistic shielding in a risk-augmented state space. Our method enforces strict safety during training, with formal guarantees depending only on the approximation quality of the backup critic Q_b . While this assumption is practically achievable in various settings, future work could leverage a dynamic risk margin that adapts to the critic’s accuracy over time, ensuring safety even in early training stages.

Our theoretical results focus on deterministic environments. To address the stochastic case, a *risk-worthiness* function could be learned to guide risk allocation among successor states.

Finally, while our approach prioritized theoretical guarantees over fine-tuned optimization, the experimental results already demonstrate the soundness and promise of the core shielding mechanism. We expect performance to further improve with more refined implementations.

Acknowledgments. The research described in this paper was partially supported by the EPSRC (grant number EP/X015823/1) and by the Moro-Barry family.

REFERENCES

- [1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*. PMLR, 22–31.
- [2] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2019. Benchmarking Safe Exploration in Deep Reinforcement Learning. *arXiv preprint arXiv:1805.05800* (2019).
- [3] Alekh Agarwal, Sham Kakade, Nan Jiang, and Wen Sun. 2020. Flambe: Structural Complexity and Representation Learning of Low Rank MDPs. In *Advances in Neural Information Processing Systems (NeurIPS)*.
- [4] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. 2017. Safe Reinforcement Learning via Shielding. *CoRR abs/1708.08611* (2017). arXiv:1708.08611 <http://arxiv.org/abs/1708.08611>
- [5] Eitan Altman. 1999. Constrained Markov Decision Processes. <https://api.semanticscholar.org/CorpusID:14906227>
- [6] Christel Baier and Joost-Pieter Katoen. 2008. *Principles of model checking*. MIT press.
- [7] Marc G. Bellemare, Will Dabney, and Rémi Munos. 2017. A Distributional Perspective on Reinforcement Learning. In *International Conference on Machine Learning (ICML)*.
- [8] Dimitri P. Bertsekas and Steven E. Shreve. 2007. *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific.
- [9] Miguel Calvo-Fullana, Santiago Paternain, Luiz F. O. Chamon, and Alejandro Ribeiro. 2021. State Augmented Constrained Reinforcement Learning: Overcoming the Limitations of Learning with Rewards. *CoRR abs/2102.11941* (2021). arXiv:2102.11941 <https://arxiv.org/abs/2102.11941>
- [10] Yinlam Chow, Ofir Nachum, Edgar Dueñez-Guzman, and Mohammad Ghavamzadeh. 2018. A lyapunov-based approach to safe reinforcement learning. *Advances in neural information processing systems* 31 (2018).
- [11] Yinlam Chow, Ofir Nachum, Aleksandra Faust, Mohammad Ghavamzadeh, and Edgar A. Duéñez-Guzmán. 2019. Lyapunov-based Safe Policy Optimization for Continuous Control. *CoRR abs/1901.10031* (2019). arXiv:1901.10031 <http://arxiv.org/abs/1901.10031>
- [12] Ingy Elsayed-Aly, Suda Bharadwaj, Christopher Amato, Rüdiger Ehlers, Ufuk Topcu, and Lu Feng. 2021. Safe Multi-Agent Reinforcement Learning via Shielding. In *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, Frank Dignum, Alessio Lomuscio, Ulle Endriss, and Ann Nowé (Eds.). ACM, 483–491. <https://doi.org/10.5555/3463952.3464013>
- [13] Scott Fujimoto, Herke Van Hoof, and David Meger. 2018. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning (ICML)*. PMLR, 1587–1596.
- [14] Javier Garcia and Fernando Fernández. 2015. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research* 16, 1 (2015), 1437–1480.
- [15] Alexander W. Goodall and Francesco Belardinelli. 2024. Leveraging Approximate Model-based Shielding for Probabilistic Safety Guarantees in Continuous Environments. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2024, Auckland, New Zealand, May 6-10, 2024*, Mehdi Dastani, Jaime Simão Sichman, Natasha Alechina, and Virginia Dignum (Eds.). International Foundation for Autonomous Agents and Multiagent Systems / ACM, 2291–2293. <https://doi.org/10.5555/3635637.3663137>
- [16] Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, and Alois C. Knoll. 2022. A Review of Safe Reinforcement Learning: Methods, Theory and Applications. *CoRR abs/2205.10330* (2022). <https://doi.org/10.48550/ARXIV.2205.10330> arXiv:2205.10330
- [17] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *CoRR abs/1801.01290* (2018). arXiv:1801.01290 <http://arxiv.org/abs/1801.01290>
- [18] Qiang He and Xinwen Hou. 2020. WD3: Taming the Estimation Bias in Deep Reinforcement Learning. In *32nd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2020, Baltimore, MD, USA, November 9-11, 2020*. IEEE, 391–398. <https://doi.org/10.1109/ICTAI50040.2020.00068>
- [19] Nils Jansen, Bettina Könighofer, Sebastian Junges, Alex Serban, and Roderick Bloem. 2020. Safe Reinforcement Learning Using Probabilistic Shields (Invited Paper). In *31st International Conference on Concurrency Theory, CONCUR 2020, September 1-4, 2020, Vienna, Austria (Virtual Conference) (LIPIcs, Vol. 171)*, Igor Konnov and Laura Kovács (Eds.). Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 3:1–3:16. <https://doi.org/10.4230/LIPICS.CONCUR.2020.3>
- [20] Jiaming Ji, Borong Zhang, Jiayi Zhou, Xuehai Pan, Weidong Huang, Ruiyang Sun, Yiran Geng, Yifan Zhong, Juntao Dai, and Yaodong Yang. 2023. Safety-Gymnasium: A Unified Safe Reinforcement Learning Benchmark. arXiv:2310.12567 [cs.AI] <https://arxiv.org/abs/2310.12567>
- [21] Jiaming Ji, Jiayi Zhou, Borong Zhang, Juntao Dai, Xuehai Pan, Ruiyang Sun, Weidong Huang, Yiran Geng, Mickel Liu, and Yaodong Yang. 2023. OmniSafe: An Infrastructure for Accelerating Safe Reinforcement Learning Research.

- arXiv:2305.09304 [cs.LG] <https://arxiv.org/abs/2305.09304>
- [22] Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. 2019. Learning to Drive in a Day. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*. IEEE, 8248–8254. <https://doi.org/10.1109/ICRA.2019.8793742>
 - [23] Jens Kober, J. Bagnell, and Jan Peters. 2013. Reinforcement Learning in Robotics: A Survey. *The International Journal of Robotics Research* 32 (09 2013), 1238–1274. <https://doi.org/10.1177/0278364913495721>
 - [24] Edwin Hamel-De le Court, Francesco Belardinelli, and Alexander W. Goodall. 2025. Probabilistic Shielding for Safe Reinforcement Learning. In *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, Toby Walsh, Julie Shah, and Zico Kolter (Eds.). AAAI Press, 16091–16099. <https://doi.org/10.1609/AAAI.V39I15.33767>
 - [25] Xingtu Liu. 2024. Information-Theoretic Generalization Bounds for Batch Reinforcement Learning. *Entropy* 26, 11 (2024), 995.
 - [26] Yongshuai Liu, Jiaxin Ding, and Xin Liu. 2020. IPO: Interior-Point Policy Optimization under Constraints. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press, 4940–4947. <https://doi.org/10.1609/AAAI.V34I04.5932>
 - [27] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing Atari with Deep Reinforcement Learning. *CoRR abs/1312.5602* (2013). arXiv:1312.5602 <http://arxiv.org/abs/1312.5602>
 - [28] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.
 - [29] Haritz Odriozola-Olalde, Maider Zamalloa, and Nestor Arana-Arexolaieba. 2023. Shielded Reinforcement Learning: A review of reactive methods for safe learning. In *2023 IEEE/SICE International Symposium on System Integration (SII)*. 1–8. <https://doi.org/10.1109/SII55687.2023.10039301>
 - [30] Alex Ray, Joshua Achiam, and Dario Amodei. 2019. Benchmarking Safe Exploration in Deep Reinforcement Learning. In *arXiv preprint arXiv:1910.01708*.
 - [31] Harsh Satija, Philip Amortila, and Joelle Pineau. 2020. Constrained markov decision processes via backward value functions. In *International Conference on Machine Learning*. PMLR, 8502–8511.
 - [32] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. 2015. Trust Region Policy Optimization. *CoRR abs/1502.05477* (2015). arXiv:1502.05477 <http://arxiv.org/abs/1502.05477>
 - [33] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *CoRR abs/1707.06347* (2017). arXiv:1707.06347 <http://arxiv.org/abs/1707.06347>
 - [34] Aivar Sootla, Alexander I. Cowen-Rivers, Taher Jafferjee, Ziyang Wang, David Henry Mguni, Jun Wang, and Haitham Ammar. 2022. Saute RL: Almost Surely Safe Reinforcement Learning Using State Augmentation. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA (Proceedings of Machine Learning Research, Vol. 162)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato (Eds.). PMLR, 20423–20443. <https://proceedings.mlr.press/v162/sootla22a.html>
 - [35] Krishnan Srinivasan, Benjamin Eysenbach, Sehoon Ha, Jie Tan, and Chelsea Finn. 2020. Learning to be Safe: Deep RL with a Safety Critic. *CoRR abs/2010.14603* (2020). arXiv:2010.14603 <https://arxiv.org/abs/2010.14603>
 - [36] Adam Stooke, Joshua Achiam, and Pieter Abbeel. 2020. Responsive Safety in Reinforcement Learning by PID Lagrangian Methods. In *ICML*.
 - [37] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement Learning: An Introduction* (2nd ed.). MIT press, Cambridge, MA.
 - [38] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction*. A Bradford Book, Cambridge, MA, USA.
 - [39] Sebastian Thrun and Anton Schwartz. 1993. Issues in using function approximation for reinforcement learning. In *Proceedings of the 1993 Connectionist Models Summer School*. Lawrence Erlbaum.
 - [40] Hado Van Hasselt. 2010. Double Q-learning. In *Advances in neural information processing systems*. 2613–2621.
 - [41] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado P. van Hasselt, Marc G. Bellemare, and Nando de Freitas. 2016. Dueling Network Architectures for Deep Reinforcement Learning. In *International Conference on Machine Learning (ICML)*.
 - [42] Christopher J. C. H. Watkins and Peter Dayan. 1992. Q-Learning. *Machine Learning* 8, 3–4 (1992), 279–292.
 - [43] Zhihan Xie, Qi Cai, Ethan Zhou, Alexander Risteski, and Alexander G. Gray. 2021. Bellman Error is a Good Proxy for Value Error. In *International Conference on Machine Learning (ICML)*.
 - [44] Wen-Chi Yang, Giuseppe Marra, Gavin Rens, and Luc De Raedt. 2023. Safe Reinforcement Learning via Probabilistic Logic Shields. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*. ijcai.org, 5739–5749. <https://doi.org/10.24963/IJCAI.2023/637>
 - [45] Yiming Zhang, Quan Vuong, and Keith Ross. 2020. First order constrained optimization in policy space. *Advances in Neural Information Processing Systems* 33 (2020), 15338–15349.
 - [46] Yiming Zhang, Quan Vuong, and Keith W. Ross. 2020. First Order Optimization in Policy Space for Constrained Deep Reinforcement Learning. *CoRR abs/2002.06506* (2020). arXiv:2002.06506 <https://arxiv.org/abs/2002.06506>

A TECHNICAL APPENDICES AND SUPPLEMENTARY MATERIAL

A.1 ProSH-TD3 implementation

We provide additional details on the implementation of ProSH-TD3 and the theoretical guarantees that ensure its safety during training. First, we describe the full training procedure, including the update rules for both the main and backup actor-critic pairs. We then present the exact loss functions used to train each component of the algorithm. Finally, we prove that ProSH-TD3 sampling remains safe even in the presence of exploration noise and hybrid training episodes.

A.1.1 Pseudo-code and training. The complete algorithm is outlined in Algorithm 2, including the training of the main actor and the backup actor, using normal episode where the main actor is used for sampling, and hybrid episode where the backup actor is used for sampling after several steps of the main actor.

Algorithm 2 PROSH-TD3

```

1: Input: cost budget  $d$ , margin  $\delta$ , discount factors  $\gamma, \gamma_c$ , hybrid_delay, policy_delay
2: Initialize actor-critic pairs  $Q_b^{\theta_i}, \pi_b^\phi, \bar{Q}_r^{\xi_i}, \bar{\pi}_r^\psi$ ,  $ep\_number \leftarrow 0, L \leftarrow []$ 
3: for each episode do
4:    $s \leftarrow s_{init}, r \leftarrow d - \delta, step\_ind \leftarrow 0$ 
5:   if  $ep\_number \bmod hybrid\_delay = 0$  then
6:      $switch\_step \leftarrow \text{rand}(0, \text{mean}(L))$ 
7:   else
8:      $switch\_step \leftarrow +\infty$ 
9:   end if
10:  while not done do
11:    if  $step\_ind \leq switch\_step$  then
12:      Sample  $\bar{a} \sim \Xi(\bar{\pi}_r^\psi)(s)$  and add noise to  $\bar{a}$ 
13:      Execute  $\bar{a}$ , observe  $\bar{s}'$ , reward  $r$ , cost  $c$ 
14:    else
15:       $(s, x) \leftarrow \bar{s}$ 
16:      Sample  $a \sim \pi_b^\phi$  and add noise to  $a$ 
17:      Execute  $\bar{a} = (a, x)$ , observe  $\bar{s}'$ , reward  $r$ , cost  $c$ 
18:    end if
19:    store  $(\bar{s}, \bar{a}, r, c, \bar{s}')$  in  $\mathcal{D}$ 
20:     $\bar{s} \leftarrow \bar{s}', step\_ind \leftarrow step\_ind + 1$ 
21:    Sample from  $\mathcal{D}$  and update critic networks by minimizing losses  $\mathcal{L}(\theta_i), \mathcal{L}(\xi_i)$ 
22:    if  $ep\_number \bmod policy\_delay = 0$  then
23:      update actor networks by minimizing losses  $\mathcal{L}(\phi), \mathcal{L}(\psi)$ 
24:    end if
25:  end while
26:  append  $step\_ind$  to  $L$ ,  $ep\_number \leftarrow ep\_number + 1$ 
27: end for
28: return projected policy  $\tilde{\pi}$  (via Thm. 2)

```

Actor-critic pairs training. We train the actor-critic pairs with batch sampling from a replay buffer in which we store every transition made in the augmented MDP. For the backup critic, we use the training target $y_b(c, s')$, proposed in [18], equal to

$$c + \gamma_c \left[\beta \min_{j \in \{1,2\}} Q_b^{\theta_j^{targ}}(s', \pi_b^{\psi^{targ}}(s') + \epsilon) + \frac{1 - \beta}{2} \sum_{j \in \{1,2\}} Q_b^{\theta_j^{targ}}(s', \pi_b^{\psi^{targ}}(s') + \epsilon) \right],$$

where ϵ is a small gaussian noise and β is a hyperparameter. Compared to the standard TD3 training target, this target helps combat the underestimation bias of TD3 [18].

Both of the backup critic heads are learned by regressing to this target:

$$\mathcal{L}(\theta_i) = \mathbb{E}_{((s,x),(a,y),r,c(s',x')) \sim \mathcal{D}} \left[\left(Q_b^{\theta_i}(s, a) - y_b(c, s') \right)^2 \right].$$

The backup actor is updated with the standard TD3 actor loss:

$$\mathcal{L}(\phi) = -\mathbb{E}_{(s,x) \sim \mathcal{D}} \left[Q_b^{\theta_1}(s, \pi^\phi(s)) \right].$$

The main critic is updated with a loss adapted from TD3, but taking into account that the shielded main actor outputs three actions with their respective probabilities:

$$\mathcal{L}(\xi_i) = \mathbb{E}_{(\bar{s}, \bar{a}, r, c, \bar{s}') \sim \mathcal{D}} \left[\left(\bar{Q}_r^{\xi_i}(\bar{s}, \bar{a}) - y_r(r, \bar{s}') \right)^2 \right], \text{ where}$$

$$y_r(r, \bar{s}') = r + \gamma \sum_{k \in \{1,2,3\}} \lambda_k \min_{j \in \{1,2\}} \bar{Q}_r^{j, \text{arg}}(\bar{s}', \bar{a}_k + \epsilon) \text{ with } \Xi(\bar{\pi}_r^\psi)(\bar{s}') = \sum_{k \in \{1,2,3\}} \lambda_k \bar{a}_k.$$

Similarly, the main actor is updated taking into account the shield:

$$\mathcal{L}(\psi) = -\mathbb{E}_{\bar{s} \sim \mathcal{D}} \left[\sum_{k \in \{1,2,3\}} \lambda_k \bar{Q}_r^{\xi_1}(\bar{s}, \bar{a}_k) \right] \text{ where } \Xi(\bar{\pi}_r^\psi)(\bar{s}) = \sum_{k \in \{1,2,3\}} \lambda_k \bar{a}_k.$$

Note that the gradients can flow through the λ_k and the $\bar{a}_k$ in the main actor training loss as they are computed in a fully differentiable way.

A.1.2 Safe sampling guarantees. The algorithm 2 includes noise for the exploration as well as exploration rounds to train the backup actor and the backup critic. We will present Theorem 5 stating that the sampling is still safe and provide an upper-bound for the cost.

THEOREM 5 (SAFETY OF THE SHIELDING WITH NOISE). *We consider a MDP $\mathcal{M}$ and any two policies $\bar{\pi}_1$ and $\bar{\pi}_2$, and $\xi \in [0, 1]$ a small number. Then, with $\bar{\pi}$ the policy defined as*

$$\bar{\pi}(s) = (1 - \xi)\bar{\pi}_1(s) + \xi\bar{\pi}_2(s)$$

for all $s \in \mathcal{S}$, the discounted cost of the policy $\bar{\pi}$ satisfies

$$C(\bar{\pi}) \leq C(\bar{\pi}_1) + \frac{\xi c_{\max}}{1 - \gamma_c} \frac{1}{1 - (1 - \xi)\gamma_c}.$$

We now consider the effective policy used during sampling, and not only the policy that the algorithm outputs. We aim to show that it also satisfies the safety guarantees. In practice, PROSH alternates between two modes: standard training episodes using the main actor, and hybrid episodes where the backup actor is used after several steps of the main actor. This alternation is used to improve the training of the backup critic Q_b . In both cases, a small noise is added.

For every augmented state (s, x) , both the main actor policy and the hybrid policy, without noise, can be written as

$$\bar{\pi}(s, x) = \lambda(s, x)\bar{\pi}_1 + (1 - \lambda(s, x))\bar{\pi}_b,$$

where $\bar{\pi}_1$ is the main actor and $\bar{\pi}_b$. Hence, the policy $\bar{\pi}$ is also Q_b -shielded, and its cost satisfies the estimate

$$C(\bar{\pi}) \leq x_0 + \frac{2\Delta_b}{1 - \gamma_c}.$$

We now consider the final policy used for sampling with noise, and define

$$\bar{\pi}_0 = (1 - \xi)\bar{\pi} + \xi\bar{\pi}_{\text{noise}}.$$

Using theorem 5, we have the upperbounds:

$$C(\bar{\pi}_0) \leq C(\bar{\pi}) + \frac{1}{1 - \gamma_c} \frac{\xi c_{\max}}{1 - (1 - \xi)\gamma_c} \leq x_0 + \frac{2\Delta_b}{1 - \gamma_c} + \frac{1}{1 - \gamma_c} \frac{\xi c_{\max}}{1 - (1 - \xi)\gamma_c}.$$

So the policy used for sampling is also safe, and the term induced by the noise can be removed by choosing

$$x_0 \leq d - \frac{1}{1 - \gamma_c} \frac{\xi c_{\max}}{1 - (1 - \xi)\gamma_c}.$$

A.2 Velocity and Navigation environments

We evaluate our method on environments from the Safe Velocity and Safe Navigation suites of Safety-Gymnasium. Specifically, we use HalfCheetah, and Hopper (velocity tasks), and CarGoal, CarCircle, PointGoal, and PointCircle (navigation tasks).

Velocity Tasks. These tasks focus on moving forward quickly while remaining within a safe speed limit. The reward is proportional to forward progress, and the cost is binary: it equals 1 if the agent’s velocity exceeds 50% of its training-time maximum, and 0 otherwise.

- HalfCheetah: A 2D robot with 9 links and 8 joints learns to run forward via torque control.
- Hopper: A one-legged robot learns to hop forward by coordinating joint torques.

Navigation Tasks. These tasks require agents to reach goals or follow paths while avoiding unsafe regions. The reward is based on proximity to the goal or path adherence, and the cost equals 1 upon contact with obstacles or hazardous zones.

- CarGoal vs. PointGoal: Both tasks require reaching a fixed target location.
 - CarGoal uses a wheeled robot with non-holonomic constraints, simulating a simplified autonomous vehicle.
 - PointGoal uses a free-moving point mass, which can change direction instantaneously.
- CarCircle vs. PointCircle: The goal is to follow a predefined circular trajectory.
 - CarCircle again uses the wheeled car agent, requiring continuous steering control.
 - PointCircle uses the point mass agent, allowing for simpler motion planning but still subject to the same cost constraints.

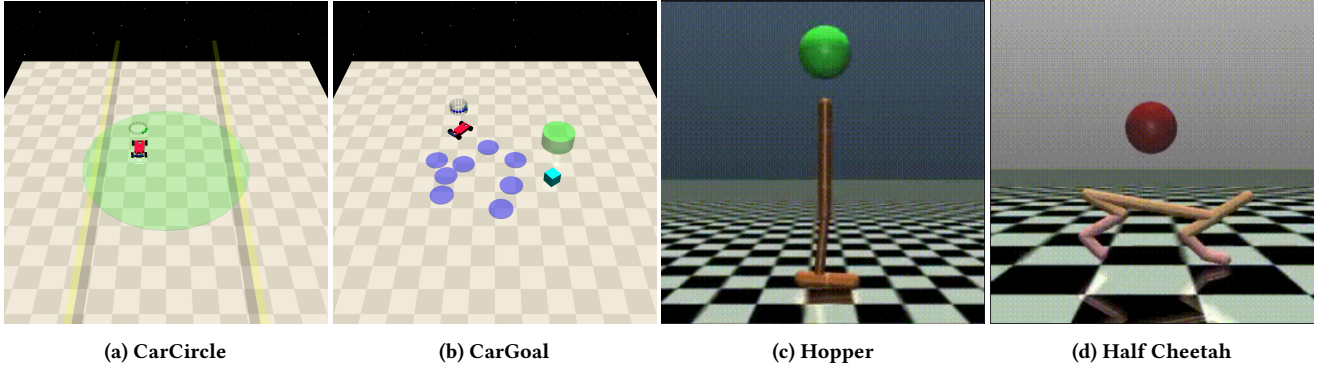


Figure 14: On the left, two environments of the Safe Navigation suite from Safety-Gymnasium, CarCircle and CarGoal. In CarCircle, the agent needs to circle around the center of the circle area while avoiding going outside the boundaries. In CarGoal, the agent needs to navigate to the Goal’s location while circumventing Hazards. In both case we choose $Level = 1$. On the right, two environments of the Safe Velocity suite from Safety-Gymnasium, SafetyHopper and SafetyHalfCheetah. In both cases the agent has to move as quickly as possible while adhering to a velocity constraint.

A.3 Experimental Setup

Hyperparameters. For PPO-Saute and TD3-Lag, we set $\gamma_c = 0.995$, and we set $\gamma_{Saute} = 0.995$ for PPO-Saute. For the other hyperparameters for TD3-Lag and PPO-Saute, we use for each environments the default hyperparameters provided by Omnisafe [21]. As for our algorithm, the set of hyperparameters used for every environment is given in Table 1. The more difficult the environment, the more the learning rate of the actors are decreased and the learning rate of the critics increased, as is standard for modern TD3 implementations.

In Table 1, β is the hyperparameter in the backup critic target.

Experimental Setup. All experiments were conducted on a PBS-managed high-performance computing cluster running Red Hat Enterprise Linux 8.5 (Ootpa). Each compute node is equipped with 2× AMD EPYC 7742 CPUs (128 cores total), 1 TB of RAM, and 8× NVIDIA Quadro RTX 6000 GPUs. We launched each experiment seed as an independent PBS job, allocating 1 CPU core, 12GB of RAM, and 1 GPU per job. A typical training run (1 million environment steps) took approximately 4 hours.

For PPO-Saute, CPO, FOCOPS, TD3-PID, and TD3-Lag, we used Python 3.10.0, Omnisafe version 0.5.0, Torch version 2.7.0 and Cuda 11.8. For ProSH-TD3, we used Python 3.10.0, Stable baselines3 version 2.6.0, Torch version 2.7.0 and Cuda 11.8.

Hyperparameter	HalfCheetah	PointCircle	Hopper	CarCircle	PointGoal	CarGoal
Main actor LR	1e-4	1e-4	5e-4	5e-5	5e-5	5e-6
Backup actor LR	1e-4	1e-4	5e-4	5e-5	5e-6	5e-5
Main critic LR	1e-4	1e-4	1e-3	1e-3	1e-3	1e-4
Backup critic LR	1e-4	1e-4	1e-3	3e-4	3e-4	3e-4
Reward Discount factor	0.99	0.99	0.99	0.99	0.99	0.99
Cost Discount factor	0.995	0.995	0.995	0.995	0.995	0.995
policy_delay	2	2	2	2	2	2
hybrid_delay	2	2	2	2	3	3
β	0.75	0.75	0.75	0.75	0.75	0.75
net_arch	[400,300]	[400,300]	[400,300]	[400,300]	[400,300]	[400,300]
Exploration Policy	$\mathcal{N}(0, 0.2)$	$\mathcal{N}(0, 0.2)$	$\mathcal{N}(0, 0.2)$	$\mathcal{N}(0, 0.2)$	$\mathcal{N}(0, 0.2)$	$\mathcal{N}(0, 0.2)$

Table 1: Hyperparameters for ProSH-TD3

A.4 Theoretical Analysis

In this section, we provide all the proofs for the theorem we have stated throughout the article.

A.4.1 Proof of Theorem 1. We recall Theorem 1:

Theorem 1. (Safety bound for Q_b -shields.)

Let $\mathcal{M}$ be a CMDP with cost discount factor γ_c , and let Q_b^* be its optimal state-action cost function. Assume that Q_b is a Q -value, and let $\Delta_b = \|Q_b - Q_b^*\|_{L^\infty(S, \mathcal{A})}$. For any policy $\bar{\pi} \in \bar{\Pi}_{\text{val}}$ that is Q_b -shielded, the expected discounted cost from the augmented state (s_0, x_0) obeys

$$C^{(s_0, x_0)}(\bar{\pi}) \leq \frac{x_0}{\gamma_c} + \frac{2\Delta_b}{1 - \gamma_c}, \quad \text{whenever } x_0 \geq Q_b(s_0).$$

In order to prove theorem 1, we will first prove the following lemma.

LEMMA 1. Let $C_n(s, x) = \gamma_c D_{n-1}(s, x)$, where D_n is the expected discounted cost of the n first steps of the policy $\bar{\pi}$. Then,

$$\begin{cases} C_n(s, x) \leq x + f_n(\Delta_b), & \text{when } x \geq Q_b(s), \\ C_n(s, x) \leq Q_b(s) + g_n(\Delta_b), & \text{when } x < Q_b(s) \end{cases}$$

as long as the sequences $f_n(\Delta_b)$ and $g_n(\Delta_b)$ satisfy $f_1(\Delta_b) \geq \gamma_c \Delta_b$, $g_1(\Delta_b) \geq \gamma_c \Delta_b$ and for every $n \geq 2$,

$$\begin{cases} f_n(\Delta_b) \geq 2\gamma_c \Delta_b + \gamma_c f_{n-1}, \\ g_n(\Delta_b) \geq \max(2\gamma_c \Delta_b + \gamma_c g_{n-1}(\Delta_b), 3\gamma_c \Delta_b + \gamma_c f_{n-1}(\Delta_b)). \end{cases}$$

PROOF. For $\bar{\pi}$ a Q_b -shielded policy of the augmented CMDP, we define the sequence $C_n(s, x)$ as (we omit the dependency on $\bar{\pi}$)

$$\begin{aligned} C_n(s, x) = & (1 - \lambda) \sum_{a \in A(s)} P_{\bar{\pi}}((a, y_a)|(s, x)) \sum_{s'} P(s', a) \gamma_c (c(s, a) + C_{n-1}(s', x')) \\ & + \lambda \sum_{s'} P(s'|\pi_b(s)) \gamma_c (c(s, \pi_b(s)) + C_{n-1}(s', Q_b(s'))), \quad C_0(s, x) = 0, \end{aligned}$$

where $x' = y_a - Q_b(s, a) + Q_b(s')$. Note that $C_n(s, x) = \gamma_c D_{n-1}(s, x)$, where $D_{n-1}(s, x)$ is the average discounted expected cost of the policy $\bar{\pi}$. We used the fact that $P((s', x')|(a, y_a)) = P(s'|a)$ when $x' = y_a - Q_b(s, a) + Q_b(s')$. Finally, throughout this proof, when there is no ambiguity, we will denote $P_a = P((a, y_a)|(s, x))$.

We will show by induction that

$$\begin{cases} C_n(s, x) \leq x + f_n(\Delta_b), & x \geq Q_b(s), \\ C_n(s, x) \leq Q_b(s) + g_n(\Delta_b), & x < Q_b(s). \end{cases}$$

Base case, $n = 1$, $x \geq Q_b(s)$:

In this case, we have by definition that

$$\bar{\pi}(s, x) = (1 - \lambda) \sum_{a \in A(s)} P_a(a, y_a) + \lambda(\pi_b(s), Q_b(s, \pi_b(s))),$$

where

$$x \geq \gamma_c (1 - \lambda) \sum_a P_a y_a + \gamma_c \lambda Q_b(s, \pi_b(s)).$$

The cost function $C_1(s, x)$ satisfies by definition

$$C_1(s, x) = (1 - \lambda) \sum_a P_a \sum_{s'} P(s'|a) \gamma_c (c(s, a) + C_0(s', x')) \\ + \lambda \gamma_c \sum_{s'} P(s'|\pi_b(s)) (c(s, \pi_b(s)) + C_0(s', y')),$$

so, using $C_0 = 0$,

$$C_1(s, x) = (1 - \lambda) \sum_a P_a \sum_{s'} P(s'|a) \gamma_c c(s, a) + \lambda \gamma_c \sum_{s'} P(s'|\pi_b(s)) c(s, \pi_b(s)).$$

Now, any policy taking action a has a cost at least equal to $\sum_{s'} P(s'|a) c(s, a)$, so $Q_b^*(s, a) \geq \sum_{s'} P(s'|a) c(s, a)$. Similarly, $Q_b^*(s, \pi_b(s)) \geq \sum_{s'} P(s'|\pi_b(s)) c(s, \pi_b(s))$. Using $\|Q_b^* - Q_b\|_\infty \leq \Delta_b$ gives

$$\sum_{s'} P(s', a) c(s, a) \leq Q_b(s, a) + \Delta_b, \quad \sum_{s'} P(s'|\pi_b(s)) c(s, \pi_b(s)) \leq Q_b(s, \pi_b(s)) + \Delta_b.$$

Hence, we get

$$C_1(s, x) \leq (1 - \lambda) \gamma_c \sum_a P_a Q_b(s, a) + \lambda \gamma_c Q_b(s, \pi_b(s)) + \gamma_c \Delta_b.$$

Since for any $a \in A(s)$, $y_a \geq Q_b(s, a)$, we obtain

$$C_1(s, x) \leq (1 - \lambda) \gamma_c \sum_a P_a y_a + \lambda \gamma_c Q_b(s, \pi_b(s)) + \gamma_c \Delta_b.$$

Using the weight condition, we finally obtain

$$C_1(s, x) \leq x + \gamma_c \Delta_b.$$

Base case, $n = 1$, $x < Q_b(s)$:

In this situation, we have by the definition of a Q_b -shielded policy that

$$\bar{\pi}(s, x) = (\pi_b(s), z), \quad \gamma_c z \leq x.$$

In this case, the cost function C_1 satisfies

$$C_1(s, x) = \sum_{s'} P(s'|\pi_b(s)) \gamma_c (C_0(s', Q_b(s')) + c(s, \pi_b(s))) = \sum_{s'} P(s'|\pi_b(s)) \gamma_c c(s, \pi_b(s)).$$

Any policy taking s' with probability 1 would have a cost of at least $\sum_{s'} P(s'|\pi_b(s)) c(s, \pi_b(s))$, which means

$$Q_b^*(s, \pi_b(s)) \geq \sum_{s'} P(s'|\pi_b(s)) c(s, \pi_b(s)),$$

so

$$Q_b(s, \pi_b(s)) \geq \sum_{s'} P(s'|\pi_b(s)) c(s, \pi_b(s)) - \Delta_b.$$

Consequently,

$$C_1(s, x) \leq \gamma_c (Q_b(s, \pi_b(s)) + \Delta_b) = Q_b(s) + \gamma_c \Delta_b.$$

Induction step, $x \geq Q_b(s)$:

With the same notation, $\bar{\pi}$ now satisfies

$$\bar{\pi}(s, x) = (1 - \lambda) \sum_a P_a(a, y_a) + \lambda (\pi_b(s), Q_b(s, \pi_b(s))),$$

where

$$x \geq \gamma_c (1 - \lambda) \sum_a P_a y_a + \gamma_c \lambda Q_b(s, \pi_b(s)).$$

By definition, the cost functional C_n satisfies

$$C_n(s, x) = (1 - \lambda) \sum_a P_a \sum_{s'} P(s'|a) \gamma_c (c(s, a) + C_{n-1}(s', x')) \\ + \lambda \sum_{s'} P(s'|\pi_b(s)) \gamma_c (c(s, \pi_b(s)) + C_{n-1}(s', Q_b(s'))),$$

where $x' = y_a - Q_b(s, a) + Q_b(s')$. Since for all $a \in A(s)$, $y_a \geq Q_b(s, a)$, then $x' \geq Q_b(s')$, so we can apply the induction hypothesis and find

$$C_{n-1}(s', x') \leq x' + f_{n-1}(\Delta_b), \quad C_{n-1}(s', Q_b(s')) \leq Q_b(s') + f_{n-1}(\Delta_b).$$

Hence, $C_n(s, x)$ satisfies

$$C_n(s, x) \leq (1 - \lambda) A + \lambda B + \gamma_c f_{n-1}(\Delta_b),$$

where

$$A = \sum_a P_a \sum_{s'} P(s', a) \gamma_c(c(s, a) + x'),$$

$$B = \sum_{s'} P(s' | \pi_b(s)) \gamma_c(c(s, \pi_b(s)) + Q_b(s')).$$

Now, we have

$$Q_b^*(s, a) = c(s, a) + \sum_{s'} P(s' | a) Q_b^*(s') \geq c(s, a) + \sum_{s'} P(s' | a) (Q_b(s') - \Delta_b),$$

so $Q_b(s, a) \geq c(s, a) + \sum_a P_a(s' | a) - 2\Delta_b$. Using additionally $x' = y_a - Q_b(s, a) + Q_b(s')$, we obtain

$$\begin{aligned} A &\leq \sum_a P_a \sum_{s'} P(s' | a) \gamma_c(c(s, a) + y_a - Q_b(s, a) + Q_b(s')) \\ &\leq \sum_a P_a \sum_{s'} P(s' | a) \gamma_c Q_b(s') - \sum_a P_a \sum_{s'} P(s' | a) \gamma_c Q_b(s') \\ &\quad + \sum_a P_a \sum_{s'} P(s', a) \gamma_c y_a + 2\gamma_c \Delta_b \\ &\leq \sum_a \gamma_c P_a y_a + 2\gamma_c \Delta_b. \end{aligned}$$

We reason in a similar way for the second term B . We have

$$Q_b^*(s, \pi_b(s)) \geq c(s, \pi_b(s)) + \sum_{s'} P(s' | \pi_b(s)) (Q_b(s') - \Delta_b),$$

so $Q_b(s, \pi_b(s)) \geq c(s, \pi_b(s)) + \sum_{s'} P(s' | \pi_b(s)) Q_b(s') - 2\Delta_b$. Using this inequality gives for B

$$B \leq \sum_{s'} P(s' | \pi_b(s)) \gamma_c(c(s, \pi_b(s)) + Q_b(s')) \leq \gamma_c Q_b(s, \pi_b(s)) + 2\gamma_c \Delta_b.$$

Finally, we get

$$C_n(s, x) \leq (1 - \lambda) \sum_a \gamma_c P_a y_a + (1 - \lambda) 2\gamma_c \Delta_b + \lambda \gamma_c Q_b(s, \pi_b(s)) + 2\lambda \gamma_c \Delta_b + \gamma_c f_{n-1}(\Delta_b).$$

Using the shielding condition on the weights that $\bar{\pi}$ satisfies, we finally obtain

$$C_n(s, x) \leq x + 2\gamma_c \Delta_b + \gamma_c f_{n-1}(\Delta_b) = x + f_n(\Delta_b).$$

Induction step, $x < Q_b(s)$: In this case, $\bar{\pi}$ is given by

$$\bar{\pi}(s, x) = (\pi_b(s), z), \quad \gamma_c z \leq x.$$

This means that the cost functional satisfies

$$C_n(s, x) = \sum_{s'} P(s' | \pi_b(s)) \gamma_c(c(s, \pi_b(s)) + C_{n-1}(s', x')), \quad x' = z - Q_b(s, \pi_b(s)) + Q_b(s').$$

Since $z \leq x/\gamma_c$, and $x < Q_b(s)$, we obtain $z < Q_b(s)/\gamma_c$. As a result,

$$x' < Q_b(s)/\gamma_c - Q_b(s, \pi_b(s)) + Q_b(s').$$

Now, $Q_b(s, \pi_b(s)) \geq Q_b^*(s, \pi_b(s)) - \Delta_b$, and $Q_b^*(s) \leq \gamma_c Q_b^*(s, \pi_b(s))$ as it is defined as the minimum over the actions, so

$$Q_b(s, \pi_b(s)) \geq Q_b^*(s)/\gamma_c - \Delta_b.$$

Overall,

$$x' < Q_b(s') + \Delta_b.$$

Two situations are possible here. Either $x' < Q_b(s')$, or $x' \geq Q_b(s')$. We start with $x' < Q_b(s')$.

In this case, we obtain

$$C_{n-1}(s', x') \leq Q_b(s') + f_{n-1}(\Delta_b),$$

so we get for $C_n(s, x)$

$$C_n(s, x) \leq \sum_{s'} P(s' | \pi_b(s)) \gamma_c(c(s, \pi_b(s)) + Q_b(s')) + \gamma_c f_{n-1}(\Delta_b).$$

Now, $Q_b(s') \leq Q_b^*(s') + \Delta_b$, and $\sum_{s'} P(s' | \pi_b(s)) \gamma_c(c(s, \pi_b(s)) + Q_b^*(s')) = \gamma_c Q_b^*(s, \pi_b(s))$, so

$$C_n(s, x) \leq \gamma_c Q_b^*(s, \pi_b(s)) + \gamma_c \Delta_b + \gamma_c f_{n-1}(\Delta_b).$$

Finally, using $Q_b^*(s, \pi_b(s)) \leq Q_b(s, \pi_b(s)) + \Delta_b$,

$$C_n(s, x) \leq \gamma_c Q_b(s, \pi_b(s)) + 2\gamma_c + \gamma_c f_{n-1}(\Delta_b) \leq Q_b(s) + 2\gamma_c + \gamma_c f_{n-1}(\Delta_b) \leq Q_b(s) + f_n(\Delta_b).$$

Now, in the other situation, we have $x' \geq Q_b(s')$. But we also have $x' \leq Q_b(s') + \Delta_b$. Hence, we can write

$$C_{n-1}(s', x') \leq x' + f_{n-1}(\Delta_b) \leq Q_b(s') + \Delta_b + f_{n-1}(\Delta_b).$$

So

$$\begin{aligned} C_n(s, x) &\leq \sum_{s'} P(s'|\pi_b(s)) \gamma_c(c(s, \pi_b(s)) + x' + f_{n-1}(\Delta_b)) \\ &\leq \sum_{s'} P(s'|\pi_b(s)) \gamma_c(c(s, \pi_b(s)) + Q_b(s') + \Delta_b + f_{n-1}(\Delta_b)) \\ &\leq \sum_{s'} P(s'|\pi_b(s)) \gamma_c(c(s, \pi_b(s)) + Q_b^*(s') + 2\Delta_b + f_{n-1}(\Delta_b)) \\ &\leq \gamma_c Q_b^*(s, \pi_b(s)) + 2\gamma_c \Delta_b + \gamma_c f_{n-1}(\Delta_b) \\ &\leq \gamma_c Q_b(s, \pi_b(s)) + 3\gamma_c \Delta_b + \gamma_c f_{n-1}(\Delta_b) = Q_b(s) + 3\gamma_c \Delta_b + \gamma_c f_{n-1}(\Delta_b). \end{aligned}$$

□

We can now go on with the proof of theorem 1.

PROOF. Assume the assumptions of the theorem are satisfied. We use lemma 1 and obtain

$$\begin{cases} C_n(s, x) \leq x + f_n(\Delta_b), & \text{when } x \geq Q_b(s), \\ C_n(s, x) \leq Q_b(s) + g_n(\Delta_b), & \text{when } x < Q_b(s) \end{cases}$$

as long as the sequences $f_n(\Delta_b)$ and $g_n(\Delta_b)$ satisfy $f_1(\Delta_b) \geq \gamma_c \Delta_b$, $g_1(\Delta_b) \geq \gamma_c \Delta_b$ and for every $n \geq 2$,

$$\begin{cases} f_n(\Delta_b) \geq 2\gamma_c \Delta_b + \gamma_c f_{n-1}, \\ g_n(\Delta_b) \geq \max(2\gamma_c \Delta_b + \gamma_c g_{n-1}(\Delta_b), 3\gamma_c \Delta_b + \gamma_c f_{n-1}(\Delta_b)). \end{cases}$$

We can choose $f_n(\Delta_b) = \sum_{k=1}^n 2\Delta_b \gamma_c^k$ and $g_n(\Delta_b) = \sum_{k=1}^n 3\Delta_b \gamma_c^k$, and we obtain for every n

$$\begin{cases} C_n(s, x) \leq x + \sum_{k=1}^n 2\Delta_b \gamma_c^k, & \text{when } x \geq Q_b(s), \\ C_n(s, x) \leq Q_b(s) + \sum_{k=1}^n 3\Delta_b \gamma_c^k, & \text{when } x < Q_b(s) \end{cases}$$

Subsequently, we obtain for the finite expected discounted cost

$$\begin{cases} D_{n-1}(s, x) \leq \frac{x}{\gamma_c} + \sum_{k=1}^n 2\Delta_b \gamma_c^{k-1} \leq \frac{x}{\gamma_c} + 2\Delta_b \frac{1}{1-\gamma_c}, & \text{when } x \geq Q_b(s), \\ D_{n-1}(s, x) \leq \frac{Q_b(s)}{\gamma_c} + \sum_{k=1}^n 3\Delta_b \gamma_c^{k-1} \leq \frac{Q_b(s)}{\gamma_c} + 3\Delta_b \frac{1}{1-\gamma_c}, & \text{when } x < Q_b(s). \end{cases}$$

Since D_n is Cauchy, we can just take the limit and obtain

$$\begin{cases} C(s, x) \leq \frac{x}{\gamma_c} + \frac{2\Delta_b}{1-\gamma_c}, & \text{when } x \geq Q_b(s), \\ C(s, x) \leq \frac{Q_b(s)}{\gamma_c} + \frac{3\Delta_b}{1-\gamma_c}, & \text{when } x < Q_b(s). \end{cases}$$

□

A.4.2 Proof of theorem 3. We recall Theorem 3.

Theorem 3. (Optimality of the shielded policies) *Let $\mathcal{M}$ be a deterministic CMDP with safety threshold d and initial state s_i , Q_b be a Q -value, $\Delta_b = \|Q_b - Q_b^*\|_{L^\infty(\mathcal{S}, \mathcal{A})}$. Then, if we let Π the set of all policies of $\mathcal{M}$, $\bar{\Pi}_{sh}$ be the set of shielded policies of $\bar{\mathcal{M}}$, and $\bar{\Pi}^f$ be the set of valued flipping policies, we have*

$$\max_{\pi \in \Pi, C(\pi) \leq d} \mathcal{R}(\pi) \leq \max_{\bar{\pi} \in \bar{\Pi}^f, x_0 \leq \gamma_c d + \mathcal{E}} \mathcal{R}^{(s_i, x_0)}(\bar{\pi}) \leq \max_{\pi \in \bar{\Pi}_{sh}, x_0 \leq \gamma_c d + \mathcal{E}} \mathcal{R}^{(s_i, x_0)}(\bar{\pi}),$$

for $\mathcal{E} = \frac{2\Delta_b \gamma_c}{1-\gamma_c}$.

Before proving Theorem 3, we first need to introduce the following lemma.

LEMMA 2. For any deterministic MDP $\mathcal{M}$ and safety threshold d , there exists a Q_b^* -shielded flipping memoryless policy $\bar{\pi}^*$ of $\bar{\mathcal{M}}^{Q_b^*}$ such that

$$\mathcal{R}(\bar{\pi}^*) = \max_{\pi \in \Pi, C(\pi) \leq d} \mathcal{R}(\pi),$$

and starting at initial state (s_0, d) . Subsequently, $\bar{\pi}^*$ also satisfies

$$C(\bar{\pi}^*) \leq d.$$

PROOF. Let $\mathcal{M} = (S, A, s_i, P, R, C, d)$ be a deterministic CMDP with positive costs, and let C_{max} be equal to $\frac{C_{max}}{1-\gamma_c}$. For any state (s, x) of $\mathcal{M}$, and any mapping $y : A(s) \mapsto [0; C_{max}]$ such that $y(a) \geq Q_b^*(s, a)$, we let $D(s, x, y)$ be the set of points $z \in \mathbb{R}^{A(s)}$ such that $x_a \geq 0$ for all $a \in A(s)$,

$$\sum_{a \in A(s)} z_a y(a) \leq \frac{x}{\gamma_c}, \quad (1)$$

and

$$\sum_{a \in A(s)} z_a = 1. \quad (2)$$

For any s, x, y , $D(s, x, y)$ is the intersection between the convex polytope whose extreme points are all points $z^a \in \mathbb{R}^{A(s)}$ such that for any $a, a' \in A(s)$, $z^a_{a'} = 1$ if $a = a'$ and 0 otherwise, and the (convex) half-space defined by Equation (1). Thus, for any s, x, y , $D(s, x, y)$ is a convex polytope, and its extreme points are of the form $\lambda z^a + (1 - \lambda) z^{a'}$ where $a, a' \in A(s)$. Let $V(s, x, y)$ be the extreme points of $D(s, x, y)$. We let $V_b^*(s) = \min_{a \in A(s)} Q_b^*(s, a)$, and we define $\hat{\mathcal{M}}$ to be the MDP with

- States : $\hat{S} = \{(s, x) \mid s \in S, x \in [V_b^*(s); C_{max}]\}$, with initial state $\bar{s}_i = (s_i, d)$;
- Actions : $\hat{A}(s, x)$ equal to the set of all (y, v) where $y : A(s) \mapsto [0; C_{max}]$ and $y(a) \geq Q_b^*(s, a)$ for any $a \in A(s)$ and $v \in V(s, x, y)$;
- Rewards : $\hat{r}((s, x), (a, y)) = r(s, a)$;
- Transition Probability Function $\hat{P}$: for $\hat{s} = (s, x)$, $\hat{s}' = (s', x')$ in $\hat{S}$, and $\hat{a} = (y, v) \in \hat{A}(\hat{s})$, we have

$$\hat{P}(\hat{s}, \hat{v}, \hat{s}') = \begin{cases} 0 & \text{if } x' \neq y - Q_b^*(s, a) + Q_b^*(s') \text{ for all } a \in A(s) \\ \sum_{a \mid x' = y - Q_b^*(s, a) + Q_b^*(s')} v_a P(s, a, s') & \text{otherwise} \end{cases}.$$

First, notice that the MDP $\hat{\mathcal{M}}$ satisfies the hypothesis of the lower-continuous model (Definition 8.7 of [8]), and thus admits an optimal memoryless policy $\hat{\pi}^*$ (Corollary 9.17.2 of [8]). Second, any memoryfull stochastic policy π of $\mathcal{M}$ such that $C(\pi) \leq d$ can be transformed into a memoryfull policy $\hat{\pi}$ of $\hat{\mathcal{M}}$ with the same cost and reward by letting $P_{\hat{\pi}}(s, a) = \sum_{v \in V(s, x, y)} \lambda_v(y, v)$, where

$$y(a) = \mathbb{E}_{s_0, a_0, \dots \sim \pi, (s_0, a_0) = (s, a)} \sum_t \gamma_c^t c(s_t, a_t),$$

and where the λ_v are such that $\sum_{v \in V(s, x, y)} \lambda_v v = P_\pi$, $\lambda_v \in [0; 1]$ and $\sum_{v \in V(s, x, y)} \lambda_v = 1$. Conversely, similarly to Theorem 2, any memoryless deterministic policy of $\hat{\mathcal{M}}$ can be backwards projected to a memoryfull stochastic policy of $\mathcal{M}$ with the same cost and reward. Furthermore, by definition of $\hat{\mathcal{M}}$, any policy $\hat{\pi}$ of can be seen as a Q_b^* -shielded policy of $\bar{\mathcal{M}}^{Q_b^*}$, and we thus have by Theorem 1 that $C(\hat{\pi}) \leq d$. The lemma follows. $\square$

We now go on and provide a proof of Theorem 3.

Using Lemma 2, we have the existence of a Q_b^* policy $\bar{\pi}^*$ of the Risk-Augmented MDP $\bar{\mathcal{M}}^{Q_b^*}$, satisfying

$$\begin{cases} \mathcal{R}(\bar{\pi}^*) = \max_{\pi \in \Pi, C(\pi) \leq d} \mathcal{R}(\pi), \\ x_0 = d, \\ C(\bar{\pi}^*) \leq d. \end{cases}$$

Moreover, $\bar{\pi}^*$ is Q_b^* -shielded, valued, and flipping, so we can write for any $(s, x) \in \bar{S}$ such that for $x \geq Q_b^*(s)$,

$$\bar{\pi}^*(s, x) = P_1(a_1, y_1) + P_2(a_2, y_2),$$

where the risks satisfy

$$\begin{cases} x = \gamma_c(P_1 y_1 + P_2 y_2), \\ y_1 \geq Q_b^*(s, a_1), y_2 \geq Q_b^*(s, a_2). \end{cases}$$

Note that $\bar{\pi}^*$ is not defined yet on the set $\{(s, x), x < Q_b^*(s)\}$. We show now that it does not matter, since this set can not be reached from the initial state.

Define $D = \{(s, x), x \geq Q_b^*(s)\}$. Call $\bar{S}_{\bar{\pi}^*, n}$ the set of states reached following $\bar{\pi}^*$ in n steps with a non-zero probability. We have, $\bar{S}_{\bar{\pi}^*, 0} = \{(s_0, d)\} \subseteq D$. By induction, assume now that $\bar{S}_{\bar{\pi}^*, n} \subseteq D$. For any $(s, x) \in D$,

$$\bar{\pi}^*(s, x) = P_1(a_1, y_1) + P_2(a_2, y_2), \quad y_j \geq Q_b^*(s, a_j).$$

Hence, the state $\bar{\pi}^*$ can reach in one more step, using the definition of the augmented MDP, have risks satisfying

$$x_j = y_j - Q_b^*(s, a_j) + Q_b^*(s_j) \geq Q_b^*(s_j),$$

which means $\bar{\mathcal{S}}_{\bar{\pi}^*, n+1} \subseteq D$.

The challenge here is that $\bar{\pi}^*$ is not a Q_b -shielded policy, as $Q_b \neq Q_b^*$. We will now construct a policy $\bar{\pi}$, that will be Q_b -shielded, have a slightly increased cost, and yield the same reward.

Definition of $\bar{\pi}$.

The initial state of $\bar{\pi}$ is $(s_0, d + \mathcal{E})$.

We first define $\bar{\pi}$ on the set $\tilde{D} = \{(s, x), x \geq Q_b^*(s) + \mathcal{E}\}$. We show later that it is enough. For $(s, x) \in \tilde{D}$, we define

$$\bar{\pi}(s, x) = P_1(a_1, z_1) + P_2(a_2, z_2), \quad z_j = y_j + \mathcal{E} + Q_b(s, a_j) - Q_b(s_j) - c(s, a_j),$$

where $P_1, P_2, a_1, a_2, y_1, y_2$ are defined using the image of the optimal policy $\bar{\pi}_*$ at $(s, x - \mathcal{E})$ as in

$$\bar{\pi}_*(s, x - \mathcal{E}) = P_1(a_1, y_1) + P_2(a_2, y_2),$$

and s_1 (resp. s_2) is the state reached when taking a_1 (resp. a_2) in $\mathcal{M}$, since $\mathcal{M}$ is assumed to be deterministic. Note that for $x \geq Q_b^*(s) + \mathcal{E}$, we have $x - \mathcal{E} \geq Q_b^*(s)$, so $\bar{\pi}_*$ is well defined.

We first remark that the initial state $(s_0, \mathcal{E})$ is in $\tilde{D}$. Moreover, consider $\bar{\mathcal{S}}_{n, \bar{\pi}}$, the set of states reachable by $\bar{\pi}$ in n steps. We already have $\bar{\mathcal{S}}_{0, \bar{\pi}} \subseteq \tilde{D}$. Assume now that $\bar{\mathcal{S}}_{n-1, \bar{\pi}} \subseteq \tilde{D}$. We consider a path $\zeta = (s_0 \dots s_n)$ of length n , and look at the last transition. Since $s_{n-1} \in \tilde{D}$, it is of the form $s_{n-1} = (s, x)$ $\bar{\pi}$ with $x \geq Q_b^*(s) + \mathcal{E}$. $\bar{\pi}$ is defined as

$$\bar{\pi}(s, x) = P_1(a_1, z_1) + P_2(a_2, z_2), \quad z_j = y_j + \mathcal{E} + Q_b(s, a_j) - Q_b(s_j) - c(s, a_j),$$

where

$$\bar{\pi}_*(s, x - \mathcal{E}) = P_1(a_1, y_1) + P_2(a_2, y_2).$$

Using the definition of the augmented MDP for the Q-value Q_b , the state reached after taking the action (a_j, z_j) is (s_j, x_j) with

$$x_j = z_j - Q_b(s, a_j) + Q_b(s_j) = y_j - c(s, a_j) + \mathcal{E}.$$

Since $y_j \geq Q_b^*(s, a_j) = c_j + Q_b^*(s_j)$ as $\bar{\pi}_*$ is Q_b^* -shielded, we have

$$x_j \geq Q_b(s_j) + \mathcal{E},$$

so $s_n \in \tilde{D}$.

$\bar{\pi}$ is Q_b -shielded.

Now, we check if the policy is Q_b -shielded. Using the definition in the case where $x \geq Q_b(s)$, $\bar{\pi}$ is Q_b -shielded if and only if

$$x \geq \gamma_c(P_1 z_1 + P_2 z_2), \quad z_j = y_j + \mathcal{E} + Q_b(s, a_j) - Q_b(s_j) - c(s, a_j). \quad (3)$$

We use the fact that $\bar{\pi}^*$ is Q_b^* shielded, so

$$x - \mathcal{E} \geq \gamma_c(P_1 y_1 + P_2 y_2).$$

Plugging this in (3) gives

$$\begin{aligned} & \gamma_c(P_1 y_1 + P_2 y_2) + \gamma_c(P_1(-Q_b(s_1) + Q_b(s, a_1) - c(s, a_1)) \\ & \quad + P_2(-Q_b(s_2) + Q_b(s, a_2) - c(s, a_2))) + \gamma_c \mathcal{E} \\ & \leq x - \mathcal{E} + \gamma_c(P_1(-Q_b(s_1) + Q_b(s, a_1) - c(s, a_1)) \\ & \quad + P_2(-Q_b(s_2) + Q_b(s, a_2) - c(s, a_2))) + \gamma_c \mathcal{E} \\ & \leq x + (\gamma_c - 1)\mathcal{E} + 2\gamma_c \Delta_b. \end{aligned}$$

Here, we used that

$$\begin{cases} Q_b^*(s, a_j) = Q_b(s_j) + c(s, a_j), \\ |Q_b(s, a_j) - Q_b^*(s, a_j)| \leq \Delta_b, \\ |Q_b(s_j) - Q_b^*(s_j)| \leq \Delta_b, \end{cases}$$

to obtain that $|-Q_b(s_2) + Q_b(s, a_2) - c(s, a_2)| \leq 2\Delta_b$.

Hence, $\bar{\pi}$ is Q_b -shielded when

$$2\gamma_c \Delta_b \leq (1 - \gamma_c)\mathcal{E},$$

so when $\mathcal{E} \geq \frac{2\Delta_b \gamma_c}{1 - \gamma_c}$.

We also need to show that for everytime the policy $\bar{\pi}$ takes an action (a, z) , then $z \geq Q_b(s, a)$. We consider $(s, x) \in \tilde{D}$ and denote

$$\bar{\pi}(s, x) = P_1(a_1, z_1) + P_2(a_2, z_2).$$

Since

$$z_j = y_j + Q_b(s_j) - Q_b(s, a_j) + c(s, a_j) + \mathcal{E},$$

and

$$y_j \geq Q_b^*(s, a_j)$$

since $\bar{\pi}_*$ is Q_b^* -shielded, we get

$$z_j \geq Q_b^*(s, a_j) - Q_b(s, a_j) + Q_b(s_j) + c(s, a_j) + \mathcal{E} \geq Q_b(s_j) + c(s, a_j) - \Delta_b + \mathcal{E}.$$

Since $Q_b^*(s_j) + c(s, a_j) = Q_b^*(s, a_j)$ and $|Q_b(s_j) - Q_b^*(s_j)| \leq \Delta_b$, we finally obtain

$$z_j \geq Q_b(s, a_j) - 2\Delta_b + \mathcal{E}.$$

This is satisfied as long as $\frac{2\gamma c}{1-\gamma c} \geq 2$.

The rewards of $\bar{\pi}$ and $\bar{\pi}_*$ are equal.

We now define $\mathcal{M}_1$ and $\mathcal{M}_2$ the two Markov Chains as

$$\begin{cases} \mathcal{M}_1 = \overline{\mathcal{M}}_{\bar{\pi}_*}^{Q_b^*}, \text{ the Markov Chain induced by } \bar{\pi}_* \text{ on } \overline{\mathcal{M}}^{Q_b^*}, \\ \mathcal{M}_2 = \overline{\mathcal{M}}_{\bar{\pi}}^{Q_b}, \text{ the Markov Chain induced by } \bar{\pi} \text{ on } \overline{\mathcal{M}}^{Q_b}. \end{cases}$$

Since $\mathcal{M}$ is deterministic, $\overline{\mathcal{M}}^{Q_b^*}$ and $\overline{\mathcal{M}}^{Q_b}$ are also deterministic. $\mathcal{M}_1$ and $\mathcal{M}_2$ however are not deterministic, since the policies $\bar{\pi}_*$ and $\bar{\pi}$ are flipping.

The Markov Chain $\mathcal{M}_1$: Has one initial state, (x_0, d) . For any (s, x) reachable by $\bar{\pi}_*$, $x \geq Q_b^*(s)$ so $\bar{\pi}_*$ is defined as

$$\bar{\pi}_*(s, x) = P_1(a_1, y_1) + P_2(a_2, y_2),$$

for some P_1, P_2, a_1, a_2, y_1 and y_2 . Call s_1 (resp. s_2) the state reached when taking a_1 (resp. a_2) in $\mathcal{M}$ from s . Then,

$$P_{\bar{\pi}_*}((s_j, y_j - c_j)|(s, x)) = P_j, \quad c_j = c(s, a_j),$$

and $P_{\bar{\pi}_*}(\bar{z}|(s, x)) = 0$ for any other $\bar{z}$. Hence, the Markov Chain $\mathcal{M}_1$ has two transitions starting from (s, x) :

- $(s, x) \rightarrow (s_1, y_1 - c_1)$ with probability P_1 ,
- $(s, x) \rightarrow (s_2, y_2 - c_2)$ with probability P_2 .

The Markov Chain $\mathcal{M}_2$: Has one initial state, $(x_0, d + D)$. For any (s, x) reachable by $\bar{\pi}_*$, $x \geq Q_b^*(s)$ so $\bar{\pi}_*$ is defined as

$$\bar{\pi}(s, x + D) = P_1(a_1, z_1) + P_2(a_2, z_2), \quad z_j = y_j - Q_b(s_j) + Q_b(s, a_j) + D,$$

where $P_1, P_2, a_1, a_2, y_1, y_2$ are defined with $\bar{\pi}_*$ as

$$\bar{\pi}_*(s, x) = P_1(a_1, y_1) + P_2(a_2, y_2).$$

and where s_1 (resp. s_2) is the state reached when taking a_1 (resp. a_2) in $\mathcal{M}$ from s . Then,

$$P_{\bar{\pi}}((s_j, y_j - c_j + D)|(s, x)) = P_j, \quad c_j = c(s, a_j),$$

and $P_{\bar{\pi}}(\bar{z}|(s, x)) = 0$ for any other $\bar{z}$. Hence, the Markov Chain $\mathcal{M}_2$ has two transitions starting from (s, x) :

- $(s, x + D) \rightarrow (s_1, y_1 - c_1 + D)$ with probability P_1 ,
- $(s, x + D) \rightarrow (s_2, y_2 - c_2 + D)$ with probability P_2 .

With $\phi : \overline{\mathcal{S}}^{Q_b^*} \rightarrow \overline{\mathcal{S}}^{Q_b}$, defined as

$$\phi(s, x) = (s, x + D).$$

We have that

$$\begin{cases} \phi(s_0, d) = (s_0, d + D), \\ P_{\mathcal{M}_1}((s', x')|(s, x)) = P_{\mathcal{M}_2}(\phi(s', x')|\phi(s, x)), \\ r_{\mathcal{M}_1}((s, x) \rightarrow (s', x')) = r_{\mathcal{M}_2}(\phi(s, x) \rightarrow \phi(s', x')). \end{cases}$$

Hence, we have for the expected discounted reward, as it is defined as an expectation, that

$$\mathcal{R}(\bar{\pi}) = \mathcal{R}(\bar{\pi}_*).$$

A.4.3 Proof of Remark 2: better safety bound in the deterministic case.

REMARK 2. In the deterministic case, one can choose instead, in the definition of the Risk-Augmented MDP, to take the risk $x_i = y_i - c(s, a)$ instead of the risk $x_i = y_i - Q_b(s, a) + Q_b(s')$ after taking the action (a, y_i) and reaching s' . This is what we used in practice in that case, as the performances are slightly better. In that case, the optimality bound is improved, and we have

$$\max_{\pi \in \Pi, C(\pi) \leq d} \mathcal{R}^{s_0}(\pi) \leq \max_{\bar{\pi} \in \bar{\Pi}, x_0 \leq d + 2\Delta_b} \mathcal{R}^{(s_0, x_0)}(\bar{\pi}).$$

The safety results are identical.

PROOF. (Of Remark 2)

The proof is very similar to the proof of 3. Using Lemma 2 again, we have the existence of a Q_b^* policy $\bar{\pi}^*$ of the Risk-Augmented MDP $\overline{\mathcal{M}}^{Q_b^*}$, satisfying

$$\begin{cases} \mathcal{R}(\bar{\pi}^*) = \max_{\pi \in \Pi, C(\pi) \leq d} \mathcal{R}(\pi), \\ x_0 = d, \\ C(\bar{\pi}^*) \leq d. \end{cases}$$

Moreover, $\bar{\pi}^*$ is Q_b^* -shielded, valued, and flipping, so we can write for any $(s, x) \in \bar{S}$ such that for $x \geq Q_b^*(s)$,

$$\bar{\pi}^*(s, x) = P_1(a_1, y_1) + P_2(a_2, y_2),$$

where the risks satisfy

$$\begin{cases} x = \gamma_c(P_1 y_1 + P_2 y_2), \\ y_1 \geq Q_b^*(s, a_1), y_2 \geq Q_b^*(s, a_2). \end{cases}$$

Definition of $\bar{\pi}$.

We define $\tilde{D} = \{(s, x), x \geq Q_b^* + 2\Delta_b\}$, and define the policy $\bar{\pi}$ on the augmented MDP $\overline{\mathcal{M}}^{Q_b}$ as follows.

- The starting state of $\bar{\pi}$ is $(s_0, d + 2\Delta_b)$.
- For every $(s, x) \in \tilde{D}$,

$$\bar{\pi}(s, x) = P_1(a_1, z_1) + P_2(a_2, z_2), \quad z_j = y_j + 2\Delta_b,$$

where P_1, P_2, a_1, a_2, y_1 , and y_2 are defined via the image of the optimal policy $\bar{\pi}_*$ as

$$\bar{\pi}_*(s, x - 2\Delta_b) = P_1(a_1, y_1) + P_2(a_2, y_2).$$

First, Note that again, by a quick induction, all the states reachable from the initial state belong to $\tilde{D}$. This means that $\bar{\pi}$ is well defined despite having been only defined on $\tilde{D}$.

$\bar{\pi}$ is Q_b -shielded.

We verify now that $\bar{\pi}$ is Q_b -shielded. For every $(s, x) \in \tilde{D}$, $x \geq Q_b^*(s) + 2\Delta_b \geq Q_b(s)$. So the shielding conditions become

- (1) For every (a_j, z_j) that the policy takes, we have $z_j \geq Q_b(s, a_j)$.
- (2) The weights satisfy

$$x \geq \gamma_c(P_1 z_1 + P_2 z_2).$$

For the first one, the optimal policy π_* is Q_b^* shielded, so that we have

$$y_j \geq Q_b^*(s, a_j).$$

Hence,

$$z_j \geq Q_b^*(s, a_j) + 2\Delta_b \geq Q_b(s, a_j) + \Delta_b.$$

For the second one, we have, again because $\bar{\pi}_*$ is Q_b^* -shielded

$$x - 2\Delta_b \geq \gamma_c(P_1 y_1 + P_2 y_2),$$

so

$$\gamma_c(P_1 z_1 + P_2 z_2) = \gamma_c(P_1 y_1 + P_2 y_2 + 2\Delta_b) \leq (x - 2\Delta_b) + 2\gamma_c \Delta_b \leq x.$$

So $\bar{\pi}$ is Q_b -shielded. Again, we prove that the rewards are identical by showing that the Markov Chain are identical, the details are identical to the proof of theorem 3.

□

A.4.4 Proof of Theorem 5: Safety preserved with additional noise. **Theorem 5.**(Safety of the shielding with noise) We consider a MDP $\mathcal{M}$ and any two policies $\bar{\pi}_1$ and $\bar{\pi}_2$, and $\xi \in [0, 1]$ a small number. Then, with $\bar{\pi}$ the policy defined as

$$\bar{\pi}(s) = (1 - \xi)\bar{\pi}_1(s) + \xi\bar{\pi}_2(s)$$

for all $s \in \mathcal{S}$, the discounted cost of the policy $\bar{\pi}$ satisfies

$$C(\bar{\pi}) \leq C(\bar{\pi}_1) + \frac{\xi c_{\max}}{1 - \gamma_c} \frac{1}{1 - (1 - \xi)\gamma_c}.$$

PROOF. We let $C_n = \gamma_c D_{n-1}$, where D_n is the discounted cost of the n first steps. We will use an induction to show

$$C_n(\bar{\pi}, s) \leq C_n(\bar{\pi}_1, s) + \xi \frac{\gamma_c c_{\max}}{1 - \gamma_c} \sum_{k=1}^n (1 - \xi)^k \gamma_c^k.$$

For $n = 1$, we have

$$\begin{aligned} C_1(\bar{\pi}, s) &= (1 - \xi) \sum_{a \in A(s)} P_{\bar{\pi}_1}(a|s) P(s'|a) \gamma_c (c(s, a) + C_0(\bar{\pi}, s')) \\ &\quad + \xi \sum_{a \in A(s)} P_{\bar{\pi}_2}(a|s) P(s'|a) \gamma_c (c(s, a) + C_0(\bar{\pi}, s')) \leq (1 - \xi) \gamma_c c_{\max}. \end{aligned}$$

It is in particular smaller than the induction formula for $n = 1$.

Now, assuming the property holds for $k \leq n - 1$. We write

$$\begin{aligned} C_n(\bar{\pi}, s) &= (1 - \xi) \sum_{a \in A(s)} P_{\bar{\pi}_1}(a|s) P(s'|a) \gamma_c (c(s, a) + C_{n-1}(\bar{\pi}, s')) \\ &\quad + \xi \sum_{a \in A(s)} P_{\bar{\pi}_2}(a|s) P(s'|a) \gamma_c (c(s, a) + C_{n-1}(\bar{\pi}, s')), \end{aligned}$$

so

$$\begin{aligned} C_n &\leq (1 - \xi) \sum_{a \in A(s)} P_{\bar{\pi}_1}(a|s) P(s'|a) \gamma_c (c(s, a) + C_{n-1}(\bar{\pi}, s')) + \xi \gamma_c \frac{c_{\max}}{1 - \gamma_c} \\ &\leq (1 - \xi) \sum_{a \in A(s)} P_{\bar{\pi}_1}(a|s) P(s'|a) \gamma_c \left(c(s, a) + C_{n-1}(\bar{\pi}_1, s') + \xi \frac{\gamma_c c_{\max}}{1 - \gamma_c} \sum_{k=1}^n (1 - \xi)^k \gamma_c^k \right) \\ &\quad + \xi \gamma_c \frac{c_{\max}}{1 - \gamma_c}, \end{aligned}$$

Now, we remark that

$$\sum_{a \in A(s)} P_{\bar{\pi}_1}(a|s) P(s'|a) \gamma_c (c(s, a) + C_{n-1}(\bar{\pi}_1, s')) = C_n(\bar{\pi}_1, s),$$

so the inequality becomes

$$\begin{aligned} C_n(\bar{\pi}, s) &\leq (1 - \xi) C_n(\bar{\pi}_1, s) + (1 - \xi) \xi \gamma_c \frac{\gamma_c c_{\max}}{1 - \gamma_c} \sum_{k=1}^{n-1} (1 - \xi)^k \gamma_c^k + \xi \gamma_c \frac{c_{\max}}{1 - \gamma_c} \\ &\leq C_n(\bar{\pi}_1, s) + \frac{\xi \gamma_c c_{\max}}{1 - \gamma_c} \sum_{k=1}^n (1 - \xi)^k \gamma_c^k, \end{aligned}$$

which concludes the induction and the proof. $\square$