Safe Reinforcement Learning via Recovery-based Shielding with Gaussian Process Dynamics Models

Alexander W. Goodall
Imperial College London, Department of Computing
London, United Kingdom
a.goodall22@imperial.ac.uk

Francesco Belardinelli
Imperial College London, Department of Computing
London, United Kingdom
francesco.belardinelli@imperial.ac.uk

ABSTRACT

Reinforcement learning (RL) is a powerful framework for optimal decision-making and control but often lacks provable guarantees for safety-critical applications. In this paper, we introduce a novel recovery-based shielding framework that enables safe RL with a provable safety lower bound for unknown and non-linear continuous dynamical systems. The proposed approach integrates a backup policy (shield) with the RL agent, leveraging Gaussian process (GP) based uncertainty quantification to predict potential violations of safety constraints, dynamically recovering to safe trajectories only when necessary. Experience gathered by the 'shielded' agent is used to construct the GP models, with policy optimization via internal model-based sampling – enabling unrestricted exploration and sample efficient learning, without compromising safety. Empirically our approach demonstrates strong performance and strict safety-compliance on a suite of continuous control environments. [1]

KEYWORDS

Safe Control; Reinforcement Learning; Gaussian Process

ACM Reference Format:
Alexander W. Goodall and Francesco Belardinelli. 2026. Safe Reinforcement Learning via Recovery-based Shielding with Gaussian Process Dynamics Models. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 10 pages. https://doi.org/10.65109/EAHU8566

1 INTRODUCTION

Safe reinforcement learning (RL) [20] is an active research area focused on training policies that strictly respect safety constraints during both learning and deployment. In high-stakes domains like robotics, autonomous driving, and healthcare, adhering to safety constraints is crucial, as even a single policy failure can lead to catastrophic consequences [5]. Provably safe RL [33] is a promising paradigm for these safety-critical settings, aiming to provide formal guarantees on an agent's performance. Such guarantees typically fall into two categories: *probabilistic* guarantees or *hard* (deterministic) guarantees. Probabilistic approaches leverage either known stochastic models of the environment [31, 53] or learned models of the environment to verify safety with high confidence [12, 23, 57], whereas hard-guarantee approaches incorporate strong prior knowledge (e.g. exact system dynamics or formal abstractions) to ensure absolute constraint satisfaction [3, 11]. Despite substantial progress, ensuring safety in *unknown* or *uncertain* environments remains highly challenging – existing methods often rely on idealized assumptions or suffer from scalability issues.

A widely-used paradigm for enforcing safety at runtime is *shielding* [3, 13]. In this paradigm, each action proposed by the RL agent is checked against safety constraints before execution; if an action is deemed unsafe, it is overridden by a backup (safe) action that keeps the system within a designated safe set. The backup policy is typically a simpler, trusted controller that ensures the state remains in an *operationally safe* region. This scheme allows the learned policy to operate freely when far from danger, and invokes the backup policy only near the boundary of the safe region. An important benefit of shielding is that formal guarantees can be obtained by verifying the backup policy (which is designed for safety) instead of the learned policy, greatly simplifying the verification burden, if for example, the backup policy is simpler (e.g., linear controller).

In the context of continuous control, shielding has been used in many prior works [8, 10, 11, 34, 54] and even extended to high-dimensional visual-input scenarios [22]. *Model Predictive Shielding (MPS)* [10] is a classic example of this approach: it combines a stabilizing linear controller with a recovery strategy to verify "recoverable" safe states on-the-fly. MPS demonstrated that high-performance RL is achievable with a safety net, but it also highlighted key limitations of prior shielding methods. Notably, MPS (and extensions thereof, e.g., *Robust MPS* [34], *Dynamic MPS* [8]) assumes the environment is known and deterministic; or relies on heuristics that break safety guarantees [34]. For stochastic systems, *Statistical MPS* [11], uses samples from a known stochastic model of the environment, for high levels of safety, these sampling based approaches scale poorly and loose any meaningful guarantees. Formal methods based on Hamilton-Jacobi (HJ) reachability analysis [2] compute invariant safe sets for continuous control tasks and use them to enforce safety via action replacement. While HJ-based approaches [19, 52] provide strong safety guarantees they scale exponentially in the state dimension, making them impractical for high-dimensional systems [35]. Other approaches to safe RL may encode safety as a constraint in the learning objective, e.g., via Constrained Markov Decision Process (CMDP) [4], however these methods only guarantee constraint satisfaction in expectation and are unsuitable for highly safety-critical domains [51].

Contributions. In this paper we deal with some of the limitations highlighted above; we introduce a novel *recovery-based shielding* framework for safe RL with a provable lower bound on safety probability. We summarize our key contributions as follows:

[1] Full paper & technical appendix available at: https://arxiv.org/abs/2602.12444

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/EAHU8566

(1) We combine shielding with online model learning, where the *unknown dynamics* of the system are and modelled directly via *Gaussian process (GP) dynamics models* [17], and a pre-computed safe backup policy is used to intervene on the agent's actions only when necessary. Our method does not require full knowledge of the system's dynamics, assuming only an initial backup controller, invariant set and standard smoothness conditions on the dynamics.

(2) We leverage analytic GP uncertainty estimates to predict potential constraint violations in advance, forgoing sampling entirely, and analytically verifying actions *on-the-fly*. This design enables *provable safety guarantees* up to an arbitrary level of safety without additional computational burden.

(3) We provide a concrete proof that our shielded RL agent maintains safety with high probability throughout learning and after proper calibration of the GP dynamics model (which is often quick, both in theory and in practice). In particular, we establish a formal lower bound on the probability of constraint satisfaction at all times, subject to the usual regularity assumptions for GP-based learning.

(4) We evaluate our approach on a suite of continuous control environments. The results show that our method can learn high-performing policies while strictly respecting safety constraints, outperforming a variety of baseline safe RL techniques in terms of both safety (zero constraint violations) and reward optimization.

2 PRELIMINARIES

We consider discrete time *non-linear continuous-state dynamical systems*, with both observation noise and bounded disturbances [29],

$$x(t+1) = f(x(t), u(t), w(t)) \tag{1}$$

where $x(t) \in \mathbb{R}^n$ is a *system state*, $u(t) \in \mathbb{R}^m$ is a *control input* or *action*, and $w(t) \in \mathbb{R}^k$ is a *system disturbance*, and $f : \mathbb{R}^{n+m+k} \to \mathbb{R}^n$ is an *unknown* Lipschitz continuous *transition function*.

We let $\mathcal{X} \subseteq \mathbb{R}^n$ denote the *state constraints*, $\mathcal{U} \subseteq \mathbb{R}^m$, denote the *control constraints*, and $\mathcal{W} \subseteq \mathbb{R}^k$ denote the set of disturbances. The disturbances are can be drawn from an arbitrary distribution $\mathcal{P}_\mathcal{W}$ but the support is assumed to be bounded and known. Furthermore, we denote $\mathcal{X}_0 \subset \mathcal{X}$ as the set of initial system states. The observed system state $\widehat{x}(t) \in \mathbb{R}^n$ is subject to observation noise $v(t) \in \mathbb{R}^n$, such that, $\widehat{x}(t) = x(t) + v(t)$, we assume that that $v(t)$ is drawn from the multivariate Gaussian distribution $\mathcal{P}_\mathcal{V} = \mathcal{N}(\mathbf{0}, \Sigma)$, where $\Sigma = \text{diag}([\sigma_1^2, \ldots, \sigma_n^2]) \in \mathbb{R}^{n\times n}$ and $\sigma_1^2, \ldots \sigma_n^2$ are known noise variances for each state dimension (e.g., small sensor noise).

2.1 Problem setup

Safety semantics. We formalize the safety semantics of our framework using a flexible class of state constraints. Let $\mathcal{X} \subseteq \mathbb{R}^n$ be the state space and $\mathcal{X}_{\text{safe}} \subseteq \mathcal{X}$ the admissible (safe) set. We consider four basic constraint types:

• **Box constraints.** Each dimension i of the state vector $x \in \mathbb{R}^n$ is bounded by an interval $[a_i, b_i]$:

$$\mathcal{X}_{\text{box}} = \{x \in \mathbb{R}^n : a_i \le x_i \le b_i,\ \forall i = 1, \ldots, n\} \tag{2}$$

• **Convex hull constraints.** More generally, the safe set may be defined as a convex polytope represented by a finite set of ℓ linear inequalities, where $A \in \mathbb{R}^{\ell\times n}$ and $b \in \mathbb{R}^\ell$:

$$\mathcal{X}_{\text{hull}} = \{x \in \mathbb{R}^n : Ax \le b\} \tag{3}$$

• **Inclusion (boundary) constraints.** For either box or convex hull regions, we may require that the state remains inside the region, i.e. $x(t) \in \mathcal{X}_{\text{incl}} \subseteq \mathbb{R}^n$.

• **Exclusion (obstacle) constraints.** To represent obstacles, we may require that the state avoids certain forbidden sets $\mathcal{X}_{\text{obs}}$, such as boxes or convex hulls. Formally,

$$\mathcal{X}_{\text{excl}} = \mathcal{X} \setminus \mathcal{X}_{\text{obs}} \tag{4}$$

The overall safe set $\mathcal{X}_{\text{safe}}$, is then defined as the intersection of inclusion constraints and the complement of exclusion constraints. This naturally yields non-convex feasible regions, e.g. by combining multiple disjoint polytopes with obstacle regions removed.

Safe RL objective. We consider a *deterministic reward function* $R : \mathcal{X}\times\mathcal{U}\times\mathcal{X} \to \mathbb{R}$ and a *discount factor* $\gamma \in [0, 1)$. Let Π be the set of all policies, where $\pi \in \Pi$ is a function from system states to actions $\pi : \mathcal{X} \to \mathcal{U}$. The goal is to maximize an objective function $J(\pi)$, while ensuring that, from any initial state $x(0) \in \mathcal{X}_0 \subseteq \mathcal{X}_{\text{safe}}$ and for any sequence of disturbances $\vec{w} = (w(0), w(1), \ldots) \in \mathcal{W}^\infty$ and noise $\vec{v} = (v(0), v(1), \ldots) \in \mathcal{V}^\infty$, we have $x(t) \in \mathcal{X}_{\text{safe}}$, for all $t = 0, 1, \ldots$, where $x(t+1) = f(x(t), \pi(\widehat{x}(t)), w(t))$, and $w(t) \sim \mathcal{P}_\mathcal{W}$ and $v(t) \sim \mathcal{P}_\mathcal{V}$ are drawn i.i.d. In this paper we consider the usual objective of *expected discounted return*: $J(\pi) = \mathbb{E}_\pi\left[\sum_{t=0}^\infty \gamma^t r(t)\right]$, if we denote $\Pi_{\text{safe}} \subseteq \Pi$ as the set of all safe policies, then our goal is to find a policy $\pi^* = \arg\max_{\pi\in\Pi_{\text{safe}}} J(\pi)$.

Safety is enforced with high-probability due to the unavoidable observation noise, state disturbances and epistemic GP uncertainty. We adopt of the notion of ϵ-safe policies [11]. Concretely, let $\xi(x(0), \pi, \vec{w}, \vec{v}) = (x(0), x(1), \ldots) \in \mathcal{X}^\infty$ denote a sequence of system states under policy π and from state $x(0) \in \mathcal{X}_0$, then,

Definition 2.1 (ϵ-safe policy)**.** A policy $\pi \in \Pi$ is ϵ-safe if,

$$\mathbb{P}_{\vec{w}\sim\mathcal{P}_\mathcal{W}, \vec{v}\sim\mathcal{P}_\mathcal{V}}(\xi(x(0), \pi, \vec{w}, \vec{v}) \subseteq \mathcal{X}_{\text{safe}}) \ge 1 - \epsilon \quad \forall x(0) \in \mathcal{X}_0$$

The admissible safe policy set Π_{safe} therefore consists of all ϵ-safe policies from the initial states, this is the standard requirement introduced in prior works [11].

2.2 Dynamics modelling

For modelling the unknown dynamics of the system f we use the same methodology as PILCO [17]. The dynamics model is implemented as a Gaussian process (GP) model with input $\tilde{x}(t) = (\widehat{x}(t), u(t)) \in \mathcal{X}\times\mathcal{U}$ and output targets $\Delta(t) = \widehat{x}(t) - \widehat{x}(t-1) \in \mathbb{R}^n$ which model the next step deltas. The GP model provides the following one-step predictions,

$$\begin{aligned} p(x(t) \mid x(t-1), u(t-1)) &= \mathcal{N}(x(t) \mid \mu(t), \Sigma(t)) \\ \mu(t) &= x(t-1) + \mathbb{E}_f[\Delta(t)] \\ \Sigma(t) &= \text{var}_f(\Delta(t)) \end{aligned}$$

In this paper we only consider the zero-mean prior function and radial basis function (RBF) kernel, whose covariance is given by,

$$k(\tilde{x}(\cdot), \tilde{x}'(\cdot)) = \alpha^2 \exp\left(-\frac{1}{2}(\tilde{x}(\cdot) - \tilde{x}'(\cdot))^T \Lambda (\tilde{x}(\cdot) - \tilde{x}'(\cdot))\right)$$

where α^2 models the variance of the transition function f, and $\Lambda = \text{diag}([l_1^2, \ldots l_n^2])$ corresponds to the length-scales l_i for automatic relevance determination. The posterior hyperparameters, which include the signal variance α^2, length scales l_i and noise variances

Σ_ε, are trained by evidence maximization [40]. For training and prediction we use GPJAX [37] a flexible library for GP implemented in JAX [14], which benefits from just-in-time (JIT) compilation and GPU compatibility.

Let $\tilde{X} = [\tilde{x}_1(\cdot), \ldots, \tilde{x}_D(\cdot)]$ denote the D training inputs and $y = [\Delta_1(\cdot), \ldots \Delta_D(\cdot)]$ denote the D training outputs. The posterior predictive distribution $p(\Delta(\cdot) \mid \tilde{x}(\cdot))$ for a known test input $x(\cdot)$, is Gaussian with mean and variance given by,

$$m_f(\tilde{x}(\cdot)) = \mathbb{E}_f[\Delta(\cdot)] = k_*^T (K + \sigma_\varepsilon^2 \mathbf{I})^{-1} y = k_*^T \beta \tag{5}$$

$$\sigma_f^2 = \mathrm{var}_f(\Delta(\cdot)) = k_{**} - k_*^T (K + \sigma_\varepsilon^2 \mathbf{I}) k_* \tag{6}$$

where $k_* := k(\tilde{X}, \tilde{x}(\cdot))$, $k_{**} := k(\tilde{x}(\cdot), \tilde{x}(\cdot))$, $\beta = (K + \sigma_\varepsilon^2 \mathbf{I})^{-1} y$ and K is the gram matrix with entries $K_{ij} = k(\tilde{x}_i(\cdot), \tilde{x}_j(\cdot))$. In the case where the output targets are multivariate (i.e., the state dimension $n > 1$), we train n conditionally independent GP models. Conditional independence holds for a known (deterministic) test input $x(\cdot)$, however, the joint predicative covariance becomes non-diagonal under uncertain inputs due to shared input uncertainty and cross-covariance terms, these details are covered in Sec. 4.

3 RECOVERY-BASED SHIELDING

Overview. For a candidate policy $\widehat{\pi} \in \Pi$, our goal is to construct a *shielded policy* π_{shield} that is provably ϵ-safe. In this paper, shielding is implemented by carefully switching between the learned policy $\widehat{\pi}$ and backup controller π_{backup}. The "switching criterion" depends only on the system state and analytic safety verification conditions derived from the GP uncertainty sets. The key advantage is that safety is certified without making assumptions about the structure of $\widehat{\pi}$; the shield can enforce constraints even if $\widehat{\pi}$ is an arbitrary neural network policy. Formally, π_{shield} coincides with $\widehat{\pi}$ whenever the state is judged ϵ_t-recoverable; otherwise, it defers to π_{backup} to drive the system back into a verified invariant set. This separation makes the safety proof policy-agnostic, relying only on the backup controller and uncertainty analysis, as alluded to earlier this is often easier if the backup policy comes from a restricted policy class (e.g., linear controller).

Control invariant sets. For proving infinite horizon safety for systems defined in terms of (1), a standard approach is to consider stable *equilibrium points* and *control invariant sets*. Formally, a state $x_{eq} \in \mathcal{X}$ is a (stable) *equilibrium point* iff there exists $u_{eq} \in \mathcal{U}$ such that $f(x_{eq}, u_{eq}, \mathbf{0}) = x_{eq}$. A (nominal) *control invariant set* $\mathcal{X}_{\text{inv}} \subseteq \mathcal{X}$ is a set of states such that if $x(\cdot) \in \mathcal{X}_{\text{inv}}$ then there exists $u_x \in \mathcal{U}$ such that $f(x(\cdot), u_x, \mathbf{0}) \in \mathcal{X}_{\text{inv}}$. In words, the control invariant set is a subset of system states for which there exists corresponding control inputs that can keep the system within this set indefinitely. For simply providing safety guarantees it is enough to determine an invariant set $\mathcal{X}_{\text{inv}}$ for the control policy and then establish that $\mathcal{X}_{\text{inv}} \subseteq \mathcal{X}_{\text{safe}}$. The computation of control invariant sets of general non-linear dynamics is usually based on Hamilton-Jacobi analysis [19, 35], which scale exponentially in the dimension of the state space n. In this paper we consider the class of linear (or RBF) backup controllers, for which more scalable methods exist [24, 42].

Backup policy. The goal of the backup policy π_{backup} is to drive the current system state $x(t) \in \mathcal{X}$ back into the control invariant set $\mathcal{X}_{\text{inv}}$. Crucially the backup policy π_{backup} should satisfy,

$$x(\cdot) \in \mathcal{X}_{\text{inv}} \Rightarrow \xi(x(\cdot), \pi_{\text{backup}}(x(\cdot)), \vec{w}, \vec{v}) \subseteq \mathcal{X}_{\text{inv}} \quad (\forall \vec{w}, \vec{v} \in \mathcal{W}^\infty \times \mathcal{V}^\infty) \tag{7}$$

In this paper, we consider a Zonotope-based approach [42] provided by AROC [29] for computation of control invariant sets with maximum volume around a given stable equilibrium point x_{eq}. The corresponding backup policy π_{backup} is constructed with Linear Quadratic Regulator (LQR) [6], and is defined by the feedback matrix $K \in \mathbb{R}^{n \times n}$, formally,

$$\pi_{\text{backup}}(\widehat{x}(\cdot)) = u_{eq} - K(\widehat{x}(\cdot) - x_{eq}) \tag{8}$$

so that π_{backup} satisfies, $f(x_{eq}, \pi_{\text{backup}}(x_{eq}), \mathbf{0}) = x_{eq}$. The control invariant set $\mathcal{X}_{\text{inv}}$ is then calculated by iteratively applying the control law in (8) and over-approximating the reachable sets by successive convexification, we refer the reader to [29, 42] for full details. Mathematically the control invariant set $\mathcal{X}_{\text{inv}}$ is represented as a convex polytope (or convex hull), c.f., (3) from earlier. We state our main assumptions here.

Assumption. We assume privileged access to a linear backup policy π_{backup}; and a corresponding control invariant set $\mathcal{X}_{\text{inv}}$ for which π_{backup} is invariant, i.e., (7) holds.

Recoverable states. Provided that $x_{eq} \in \mathcal{X}_{\text{safe}}$ and $\mathcal{X}_{\text{inv}} \subseteq \mathcal{X}_{\text{safe}}$ and ensuring that the system never leaves $\mathcal{X}_{\text{inv}}$ is sufficient for maintaining safety. However, since the computation of $\mathcal{X}_{\text{inv}}$ relies on convex over-approximations and linear controllers, the actual set of *admissible states* might be much larger. Rather we allow for states that are *recoverable* within some fixed time horizon $N \in \mathbb{N}$.

Definition 3.1 (Recoverable state)**.** For a given horizon N, disturbances $\vec{w} \in \mathcal{W}^{N+1}$ and observation noise $\vec{v} \in \mathcal{V}^{N+1}$ a state $x(\cdot) \in \mathcal{X}$ is *recoverable* if for the sequence $x(0), x(1), \ldots, x(N) \in \mathcal{X}^{N+1}$, given by,

$$\begin{aligned} x(0) &= x(\cdot) \\ \widehat{x}(t) &= x(t) + v(t) \quad \forall t = 0, \ldots, N \\ x(1) &= f(x(0), \widehat{\pi}(\widehat{x}(0)), w(t)) \\ x(t+1) &= f(x(t), \pi_{\text{backup}}(\widehat{x}(t)), w(t)) \quad \forall t = 1, \ldots, N \end{aligned}$$

we have $x(t) \in \mathcal{X}_{\text{safe}}$ for all $t = 0, \ldots, N$, and there exists $t' = 0, \ldots, N$, such that $x(t') \in \mathcal{X}_{\text{inv}}$.

Intuitively, a state $x(\cdot) \in \mathcal{X}$ is recoverable if for the control $u(\cdot) = \widehat{\pi}(\widehat{x}(\cdot))$ determined by the learned policy $\widehat{\pi}$, we can recover the system state within N timesteps (using the backup policy π_{backup}), back to the control invariant set $\mathcal{X}_{\text{inv}}$ (from which we have already established safety). Fig. 1 illustrates this phenomenon; noting both the *recoverable* (blue) and *irrecoverable* (red) start outside the control invariant set $\mathcal{X}_{\text{inv}}$ and inside the box constraint $\mathcal{X}_{\text{safe}}$; the backup policy drives the recoverable trajectory back into the control invariant set $\mathcal{X}_{\text{safe}}$ – establishing a high-probability safety certification, whereas the irrecoverable trajectory leaves the safe set $\mathcal{X}_{\text{safe}}$, intersecting with the explicit box constraint.

We denote $\mathcal{X}_{\text{rec}}^{\vec{w},\vec{v}}(N) \subseteq \mathcal{X}_{\text{safe}}$ as the set of all recoverable states given $\vec{w}, \vec{v} \in \mathcal{W}^{N+1} \times \mathcal{V}^{N+1}$ and within time horizon N. Given the randomness associated with the disturbances $\vec{w}$ and the observation noise $\vec{v}$, it becomes challenging to check whether the current system

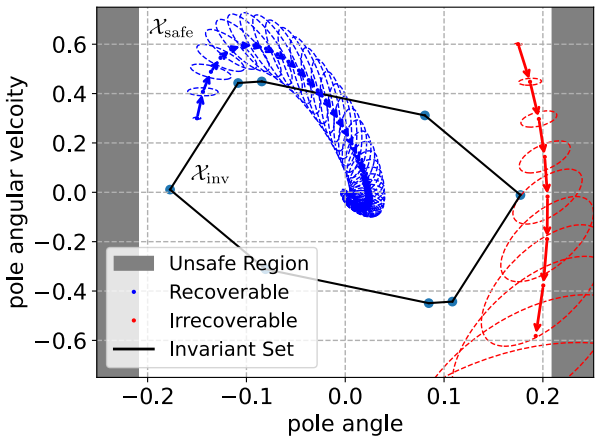


Figure 1: 99.99%-CI uncertainty sets for recoverable (blue) and irrecoverable (red) trajectories under the backup policy for `cartpole (i)/(ii)` environment.

state is recoverable given all possible realizations of $\vec{w}$ and $\vec{v}$. Rather we allow for states that are recoverable with high probability.

Definition 3.2 (ϵ-Recoverable state)**.** For a given horizon $N \in \mathbb{N}$ and tolerance $\epsilon \in [0, 1]$ a state $x(\cdot) \in \mathcal{X}$ is *ϵ-recoverable* if,

$$\mathbb{P}_{\vec{w}\sim\mathcal{P}_W,\vec{v}\sim\mathcal{P}_V}(x(\cdot) \in \mathcal{X}_{\text{rec}}^{\vec{w},\vec{v}}(N)) \geq 1-\epsilon$$

We denote $\mathcal{X}_{\text{rec}}^{\epsilon}(N) \subseteq \mathcal{X}_{\text{safe}}$ as the set of ϵ-recoverable states for some horizon N. The shielded policy π_{shield} is constructed by fixing a step-wise safety constraint $\epsilon_t \leq \epsilon$. Therefore, we write an expression for π_{shield} in terms of the ϵ_t-recoverable states. Formally,

$$\pi_{\text{shield}}(x(\cdot)) = \begin{cases} \widehat{\pi}(x(\cdot)) & \text{if } x(\cdot) \in \mathcal{X}_{\text{rec}}^{\epsilon_t}(N) \\ \pi_{\text{backup}}(x(\cdot)) & \text{otherwise} \end{cases} \tag{9}$$

Intuitively, if each state encountered during execution is verified to be ϵ_t-recoverable and the backup controller guarantees invariance within its certified region, then the resulting shielded policy π_{shield} is ϵ-safe. This is formalized in the following theorem.

Theorem 3.1 (ϵ-safe policy)**.** *Assume: (i) $\mathcal{X}_{inv}$ is an invariant set for π_{backup}, (ii) $\mathcal{X}_0 \subseteq \mathcal{X}_{inv} \subseteq \mathcal{X}_{safe}$ and (iii) $\sum_{t=0}^{T} \epsilon_t = \epsilon$, then, π_{shield} is ϵ-safe for the time horizon $T \in \mathbb{N}$.*

Remark. The proof of Theorem 3.1 is obtained by establishing the following invariant: "we can always return to $\mathcal{X}_{\text{inv}}$ with probability at least $1-\epsilon_t$ by using the backup policy π_{backup} for N timesteps". After establishing this invariant (via recursive feasibility), a union bound over the per-step safety tolerances ϵ_t is then taken. In practice we use either a constant $\epsilon_t \equiv \epsilon/T$ over a finite time horizon T, or a decreasing sequence satisfying $\sum_{t=0}^{\infty} \epsilon_t \leq \epsilon$.[2] This permits extending the guarantee to infinite horizons while keeping ϵ arbitrarily small.

In practice, both the recovery horizon N and step-wise tolerance ϵ_t are treated as hyperparameters, the choice of N has no consequences on the proof of ϵ-safety, whereas ϵ_t directly dictates the theoretical safety bound that can be obtained. The choice of N will however, directly impact the computational overhead of action selection and the restrictiveness of the shield. This overhead dominates training time when shielding is invoked at every timestep during training. Concrete runtime details can be found in the technical appendix. We note that the overhead grows roughly linearly in N for our analytic method, compared to $O(NK)$ for sampling baselines, where K is the number of samples. N should be treated as a compute-conservatism trade-off, smaller N will reduce the overhead, but might make the shield more conservative (less time to return to $\mathcal{X}_{\text{inv}}$). There is usually a sweet spot, which once reached, increasing N has no significant impact on the shield.

[2] E.g., setting $\epsilon_t = \epsilon/((t+1)^2\pi^2)$ yields $\sum_{t=0}^{\infty} \epsilon_t \leq \epsilon$.

4 GP UNCERTAINTY PREDICTION

As detailed in Sec. 2.2 the unknown system dynamics f are modelled using n GP models. We now detail how the uncertainty sets (c.f., Fig. 1) are analytically computed and propagated through the dynamics model for robustly identifying ϵ_t-recoverable states (i.e., checking that $\widehat{x}(\cdot) \in \mathcal{X}_{\text{rec}}^{\epsilon_t}$).

Prediction at uncertain inputs. We model each output dimension with an independent GP; when inputs are uncertain, output correlations arise through uncertainty propagation (moment matching). In particular, the goal is to predict $x(t)$ given $p(x(t-1))$, to do this we require the joint $p(x(t-1), u(t-1))$. This can be calculated by first integrating out the state $x(t-1)$, giving mean $\mu_u(t-1)$ and covariance $\Sigma_u(t-1)$. The cross covariance $\text{cov}[x(t-1), u(t-1)]$ is computed and then the joint $p(\tilde{x}(t-1)) = p(x(t-1), u(t-1))$ is approximated as a Gaussian with correct mean $\tilde{\mu}(t-1)$ and covariance $\tilde{\Sigma}(t-1)$. For linear controllers this computation can be done analytically [18].

We now assume a joint Gaussian distribution $p(\tilde{x}(t-1)) = \mathcal{N}(\tilde{x}(t-1) \mid \tilde{\mu}(t-1), \tilde{\Sigma}(t-1))$ at timestep $t-1$. The distribution of the next step deltas is given by the integral,

$$p(\Delta(t)) = \int p(f(\tilde{x}(t-1)) \mid \tilde{x}(t-1)) p(\tilde{x}(t-1)) d\tilde{x}(t-1) \tag{10}$$

This integral (10) is intractable, however a common approach is to approximate it as a Gaussian via moment matching [15]. The first two moments of $\Delta(t)$ are analytically computed, ignoring all higher order moments, furthermore, this makes predicting $\Delta(t+1)$ much easier as our distribution over $\Delta(t)$ is still Gaussian. Assuming the mean $\mu_\Delta(t)$ and covariance $\Sigma_\Delta(t)$ of the predictive distribution $p(\Delta(t))$ are known then the Gaussian approximation of the distribution $p(x(t))$ is given by $\mathcal{N}(x(t) \mid \mu(t), \Sigma(t))$, where,

$$\mu(t) = \mu(t-1) + \mu_\Delta(t) \tag{11}$$

$$\Sigma(t) = \Sigma(t-1) + \Sigma_\Delta(t) + \text{cov}[\Delta(t), x(t-1)] + \text{cov}[x(t-1), \Delta(t)] \tag{12}$$

$$\text{cov}[x(t-1), \Delta(t)] = \text{cov}[x(t-1), u(t-1)]\Sigma_u^{-1}(t-1)\text{cov}[u(t-1), \Delta(t)] \tag{13}$$

Again, the computation of the cross-covariances (13) here is dependent on the parametrization of the policy, e.g., for linear controllers the computation is analytical [18]. For more sophisticated policy parametrizations we would need to resort to sampling and approximate (10) directly. When we demand high-level of safety,

sampling, e.g., Monte Carlo methods, can suffer from poor sample complexity and are known to be inefficient for assessing the probability of rare events. For the analytic computation of the mean $\mu_\Delta(t)$ and covariance $\Sigma_\Delta(t)$ we refer the reader to [17], for completeness we also provide our own derivation in the technical appendix.

Uncertainty propagation. Combined with the linear backup policy, the n GP dynamics models can now provide analytic Gaussian confidence ellipsoids $\mathcal{E}(0), \ldots, \mathcal{E}(N) \subseteq \mathcal{X}$. The ellipsoids capture both aleatoric uncertainty (observed disturbances and noise) and epistemic uncertainty (lack of data), that enclose the state trajectory with probability $1 - \epsilon_t$. Formally, we let $\mu(0) = \widehat{x}(0)$ and $\Sigma(0) = \text{diag}([\sigma_1^2, \ldots \sigma_n^2])$, then the analytic means $\mu(0), \mu(1), \ldots \mu(N)$ and covariances $\Sigma(0), \Sigma(1), \ldots \Sigma(N)$ computed as described above, form the uncertainty sets $\mathcal{E}(0), \ldots, \mathcal{E}(N) \subseteq \mathcal{X}$, that are defined in terms of the z-sigma confidence ellipsoids,

$$\mathcal{E}(t) = \{x \in \mathbb{R}^n : (x - \mu(t))^T \Sigma(t)^{-1} (x - \mu(t)) \leq z^2\} \tag{14}$$

where $z > 0$ satisfies $\Phi(z) \geq 1 - \epsilon_t$ (where Φ is the CDF of the std. normal), and ϵ_t is the desired step-wise tolerance from (9). To certify that a state is ϵ_t-recoverable, we must check that these ellipsoids remain in $\mathcal{X}_{\text{safe}}$ and are eventually fully contained in X_{inv}.

Containment and exclusion checks. Given an ellipsoid $\mathcal{E}(t)$ with mean $\mu(t)$ and covariance $\Sigma(t)$:

• **Box inclusion:** $\mathcal{E}(t) \subseteq \mathcal{X}_{\text{box}}$ iff,

$$a_i \leq \mu_i(t) - z\sqrt{\Sigma_{ii}(t)}, \quad \mu_i(t) + z\sqrt{\Sigma_{ii}(t)} \leq b_i \tag{15}$$

for all $i = 1, \ldots, n$.

• **Convex hull inclusion:** $\mathcal{E}(t) \subseteq \{x : Ax \leq b\}$ iff,

$$\max_{x \in \mathcal{E}(t)} A_j x - b_j \leq 0 \quad \forall j = 1, \ldots, l \tag{16}$$

which can be solved with the following quadratic program (QP) via Sequential Least Squares Programming (SLSQP) [32]:

$$\max_x (A_i x - b_i) \text{ subject to } (x - \mu(t))^T \Sigma(t)^{-1} (x - \mu(t)) \leq z^2 \tag{17}$$

• **Exclusion (obstacle avoidance):** For an obstacle region $\mathcal{X}_{\text{obs}}$, we require

$$\mathcal{E}(t) \cap \mathcal{X}_{\text{obs}} = \varnothing \tag{18}$$

For convex $\mathcal{X}_{\text{obs}}$, this is checked by solving (17) with the reversed inequality.

• **Non-convex combinations:** If $\mathcal{X}_{\text{safe}}$ is a union/intersection of convex sets, containment reduces to checking each constituent.

We outline the full procedure in Algorithm 1. Starting from the current state distribution, uncertainty sets $\mathcal{E}(0), \ldots, \mathcal{E}(N)$ are propagated under the n GP models and backup controller (lines 1-4), see also Fig. 1. Lines 5-6 implement box and convex hull inclusion, and/or line 5 also implicitly encodes obstacle exclusion exclusion checks. Line 7 verifies recoverability before deferring to π or π_{backup}. We now state the central theorem of our paper.

Theorem 4.1. *Assume: (i) the unknown dynamics f and π_{backup} are both Lipschitz continuous in the L_1-norm, (ii) the n GP dynamics models are well calibrated in the sense that with probability (w.p.) at least $1 - \delta$ there exists $\beta(\tau) > 0$ such that $\forall t = 0 \ldots N$ and $\forall w(t) \in \mathcal{W}$ we have $\|f(x(t), u(t), w(t)) - \mu(t)\|_1 \leq \beta(\tau) tr(\Sigma(t))$, (iii) $\forall t = 0 \ldots N$ the higher order terms (e.g., skewness, kurtosis) of the true distribution $p(x(t))$ are negligible and can be ignored. Then if $\mathcal{E}(0), \ldots, \mathcal{E}(N) \subseteq \mathcal{X}_{safe}$ and if there exists $t \in \{0, \ldots N\}$ such that $\mathcal{E}(t) \subseteq \mathcal{X}_{inv}$ then $\widehat{x}(0)$ is ϵ_t-recoverable with probability at least $1 - \delta$. Thus π_{shield} is ϵ-safe by Theorem 3.1 (w.p., $1 - \delta$).*

Algorithm 1 `Shield` $(\widehat{x}(0), \widehat{\pi}, \pi_{\text{backup}}, \epsilon_t, N)$.

Input: $\widehat{x}(0) \in \mathcal{X}$, $\widehat{\pi}$, π_{backup}, $\epsilon_t \in [0, 1]$, $N \in \mathbb{N}$
Output: Safe action $u \in \mathcal{U}$

1: $z \leftarrow \Phi^{-1}(1 - \epsilon_t)$
2: $\mu(0) \leftarrow \widehat{x}(0)$, $\Sigma(0) \leftarrow \text{diag}([\sigma_1^2, \ldots, \sigma_n^2])$
3: **for** $t = 1, \ldots N$ **do**
4: Compute $\mathcal{E}(t)$ with (14).
5: Check $\mathcal{E}(t) \subseteq \mathcal{X}_{\text{safe}}$ via (15), (16) or (18).
6: Check $\mathcal{E}(N) \subseteq \mathcal{X}_{\text{inv}}$ via (15) or (16).
7: **if** $\mathcal{E}(0), \ldots, \mathcal{E}(N) \subseteq \mathcal{X}_{\text{safe}} \wedge \mathcal{E}(N) \subseteq \mathcal{X}_{\text{inv}}$ **then**
8: **return** $\widehat{\pi}(\widehat{x}(0))$
9: **else**
10: **return** $\pi_{\text{backup}}(\widehat{x}(0))$

Remark. We comment on the assumptions for Theorem 4.1, which generally hold in our experimental evaluation and analyses. First, (i) holds for many interesting control systems and π_{backup} is linear and thus Lipschitz; (ii) follows from [12] to ensure the confidence intervals we build are robust to the disturbance set $\mathcal{W}$, GPs typically satisfy this assumption under the usual regularity assumptions, more details are provided in the technical appendix; (iii) ensures that our analytic Gaussian approximation of the uncertainty sets $\mathcal{E}(t)$ capture at least $1 - \epsilon_t$ of the probability pass of the true integral (10), which is not always Gaussian [17], we provide a thorough empirical analysis in the technical appendix suggesting that the uncertainty sets broadly follow multivariate Gaussian distributions in most of our environments, thus validating this assumption in practice.

5 IMPLEMENTATION

Due to space constraints we defer the pseudocode of the full learning algorithm to the technical appendix. However, in this section, we outline the key implementation details and relevant technical definitions.

Replay buffer and policy optimization. During training we store all environment interactions of the shielded policy π_{shield} in a replay buffer $\mathcal{D} = \{(x_t, u_t, r_t, x_{t+1})\}$ consisting of tuples of state, action, reward and next state. During policy optimization, we sample *starting states* $x \sim \mathcal{D}$, from which we rollout the unshielded policy $\widehat{\pi}$ for a relatively short time horizon H inside the n GP dynamics models. These simulated rollouts provide on-policy samples without risking safety violations, since they do not interact with the real environment. Since policy optimization is done on relatively short rollouts we train $\widehat{\pi}$ with advantage actor-critic (A2C) [47]. A2C is often trained on shorter horizon rollouts and preferred here over other methods such as, PPO [46] and TRPO [44], which expect long horizon rollouts that may accumulate significant noise in the n GP models, thus, destabilising learning.

For a trajectory $\{(x_t, u_t, r_t, x_{t+1})\}_{t=0}^T$, the temporal-difference residual is,

$$\delta_t = r_t + \gamma V_\phi(x_{t+1}) - V_\phi(x_t), \tag{19}$$

where V_ϕ is the critic network and γ the discount factor. The generalized advantage estimate (GAE) [45] is defined as,

$$\widehat{A}_t^\lambda = \sum_{l=0}^{\infty} (\gamma\lambda)^l \, \delta_{t+l} \tag{20}$$

with $\lambda \in [0, 1]$ controlling the bias-variance trade-off. The policy $\widehat{\pi}$ is updated via policy gradient, i.e., $\nabla J(\widehat{\pi}) = \widehat{A}_t^\lambda \cdot \nabla \log \widehat{\pi}(u_t \mid x_t)$, and the critic parameters ϕ are optimized by minimizing the squared Bellman error: $(V_\phi(x_t) - \widehat{R}_t^\lambda)^2$, where $\widehat{R}_t^\lambda$ is the bootstrapped TD-λ return. Both the actor and critic are implemented as a two-layer feed-forward neural network with tanh activations. To improve stability, we use a target network $V_{\phi'}$ and *Polyak averaging* [38] to update a target network parameters: $\phi' \leftarrow \tau\phi + (1-\tau)\phi'$. We also use *symlog predictions* [25] to manage large or varying magnitude rewards. Given a raw return R, the symlog transform is,

$$\text{symlog}(R) = \text{sign}(R) \, \log(1 + |R|), \tag{21}$$

with inverse $\text{symexp}(z) = \text{sign}(z)(\exp(|z|) - 1)$. The critic is trained on the $\text{symlog}(R)$ targets, and regularized towards its target network predictions, reducing both sensitivity to reward magnitudes and critic overestimation.

GP Inference. Computing the exact GP posterior has cubic complexity $O(|\mathcal{D}|^3)$ in the dataset size $|\mathcal{D}|$. We therefore use sparse approximations: *Sparse GP regression* (SGPR) [48] introduces $M \ll |\mathcal{D}|$ inducing inputs $Z = \{z_m\}_{m=1}^M$ with corresponding inducing outputs u, yielding a low-rank approximation to the kernel Gram matrix; *Sparse variational GP regression* (SVGP) [28] maintains a variational distribution $q(u) = \mathcal{N}(m, S)$ over inducing outputs, minimizing $\text{KL}[q(u) \,\|\, p(u \mid y)]$. We found SGPR to be more effective in practice, but include both implementations for flexibility. The inducing inputs and other GP hyperparameters (e.g., length-scales, signal variance, noise variance) are jointly trained by marginal likelihood maximization [18].

Additional Features. To encourage better exploration we explored the use of *Prioritized replay*, inspired by [43], we implemented a novelty-based prioritized replay buffer. The state space is discretized into bins; the probability of sampling a starting state x for policy optimization is set inversely proportional to the number of visits to its bin. Furthermore, we dealt with unsafe simulated rollouts (under $\widehat{\pi}$), by zeroing out subsequent rewards and terminating the trajectory early. This corresponds to zeroing out the gradients from beyond the first unsafe timestep, preventing conflict between the learned policy and the shield, and biasing towards safe exploration.

6 EXPERIMENTAL EVALUATION

We now present our experimental results. We first summarize the environment and baselines used in this paper. For additional environment details we refer the reader to the technical appendix.

Environments. We evaluate our method on four types of continuous control tasks: (1) `cartpole` [9]: the goal is balance a pole on a moving cart; the safety constraint corresponds to preventing the pole from falling. For `cartpole` we consider two rewards: the usual +1 for maintaining upright balance, and `cartpole2`, where +1 is given for achieving a target velocity of +0.1. (2) `mountain_car` [36]: the goal is drive an under-actuated car to the top of a mountain (delayed reward); the safety constraint corresponds to avoiding a collision with a wall on the opposite side of the valley. (3) `obstacle`: a 2D navigation task where the goal is to reach a target position while avoiding a single obstacle; `obstacle2` is more challenging due to obstacle placement, `obstacle3` and `obstacle4` are more challenging still with a non-convex combination of constraints. (4) `road`, `road_2d`: are 1D and 2D road environments requiring the agent to reach a target location with a fixed speed limit.

Baselines. Due to assumption and guarantee mismatch we found it challenging to find relevant baselines. Regardless, we consider two model-free CMDP-based approaches: *Constrained Policy Optimization* (CPO) [1] and *PPO* with *Lagrangian relaxation* (PPO-Lag) [41]. These algorithms assume no knowledge of system dynamics, making them natural references for sample efficiency. However, since our approach leverages a precomputed backup controller and control invariant set, we make strictly stronger assumptions; thus the comparison is not directly fair, but favours the baselines in terms of assumptions. We also consider two model-based approaches: MPS [10] and a recent extension DMPS [8]. These methods assume knowledge of the *deterministic* dynamics of the system, a much stronger assumption than ours. To ensure compatibility, we omit disturbances and observation noise when running MPS/DMPS, effectively simplifying the environments. This adjustment makes the comparison unfair in the opposite direction; our method operates under strictly harder conditions.

Results. We summarize the results for our approach (A2C-GP-Shield), CPO, PPO-Lag, MPS and DMPS in Tab. 1. In all environments, A2C-GP-Shield achieves *perfect safety probability* once the n GP models are properly calibrated, demonstrating the effectiveness of our analytic shielding framework in enforcing strict safety. Importantly, this is achieved without sacrificing return: in 7/10 cases our method achieves the highest mean reward across the baselines that strictly enforce the safety constraint with probability 1. There is a clear trade-off in some environments; the CMDP-based approaches (CPO and PPO-Lag) optimize constraint satisfaction only in expectation; as a result, they often achieve higher raw returns (e.g., `cartpole2`, `obstacle2`) but at the cost of non-negligible safety violations (> 0 unsafe probability). This highlights their inability to prevent irrecoverable failures during training. In contrast, our approach guarantees strict constraint satisfaction at every timestep, effectively ruling out unsafe trajectories. On the other hand, MPS and DMPS also achieve perfect safety but under much stronger assumptions (e.g., known deterministic dynamics). Overall, A2C-GP-Shield closes the gap between the two extremes: unlike CMDP methods, it enforces safety strictly (not just in expectation), and unlike MPS/DMPS, it does not require access to exact dynamics. This balance explains why A2C-GP-Shield consistently achieves both high safety and competitive return across all tested environments.

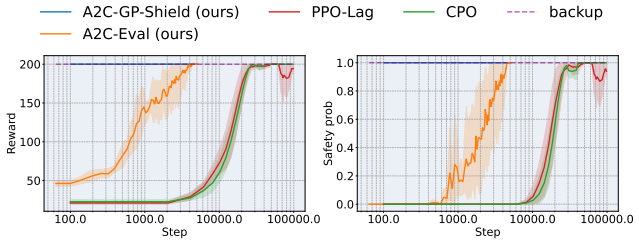
Assessing sample efficiency. To fairly assess sample efficiency we compare only against model-free baselines (CPO, PPO-Lag), excluding MPS and DMPS since they assume full knowledge of deterministic dynamics. We also report A2C-Eval, which simply executes the learned policy $\widehat{\pi}$ without shielding after each policy update. Although unsafe in practice (as $\widehat{\pi}$ cannot be verified), this

Table 1: Empirical mean return and safety probability (at the end of training), bold text denotes the best score, asterisk (*) denoted the best reward that also satisfies the safety constraint with probability 1, standard error (SE) bars (averaged over 5 independent runs) are reported. We also include the theoretical lower bound established by Thm. 3.1 (based on $\epsilon_t = 10^{-5}$).

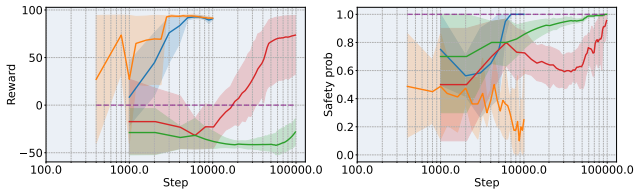
Env.	Metric	A2C-GP-Shield	MPS	DMPS	CPO	PPO-Lag	Thm. 3.1
cartpole	Return	**200.0 ± 0.00**	**200.0 ± 0.00**	**200.0 ± 0.00**	**200.0 ± 0.00**	188.0 ± 5.53	$1-\epsilon = 0.98$
	Safety Prob.	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	0.910 ± 0.04	
cartpole2	Return	30.7 ± 10.4	43.2 ± 11.1	65.1 ± 23.1*	152.0 ± 31.2	**171.0 ± 7.53**	$1-\epsilon = 0.98$
	Safety prob.	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	0.980 ± 0.01	0.811 ± 0.12	
mountain_car	Return	**91.3 ± 1.08***	85.1 ± 8.17	81.2 ± 0.28	−30.4 ± 6.74	73.2 ± 18.7	$1-\epsilon = 0.9$
	Safety prob.	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	0.995 ± 0.00	0.884 ± 0.05	
obstacle	Return	32.2 ± 0.10	8.31 ± 34.5	32.7 ± 0.21	32.5 ± 0.30	**32.9 ± 0.01***	$1-\epsilon = 0.98$
	Safety Prob.	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	
obstacle2	Return	22.9 ± 5.76*	−1.82 ± 3.24	20.2 ± 13.9	15.3 ± 6.08	**34.2 ± 0.01**	$1-\epsilon = 0.98$
	Safety Prob.	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	0.939 ± 0.02	0.000 ± 0.00	
obstacle3 (non-convex)	Return	4.69 ± 6.40*	−0.67 ± 2.12	4.28 ± 5.92	8.56 ± 12.3	**33.4 ± 0.114**	$1-\epsilon = 0.98$
	Safety Prob.	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	0.900 ± 0.141	0.148 ± 0.296	
obstacle4 (non-convex)	Return	15.5 ± 6.11*	−1.12 ± 3.13	11.3 ± 8.04	8.56 ± 12.3	**33.4 ± 0.114**	$1-\epsilon = 0.98$
	Safety Prob.	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	0.924 ± 0.0833	0.00 ± 0.00	
road	Return	**23.0 ± 0.01***	22.7 ± 0.04	22.8 ± 0.02	22.9 ± 0.05	22.9 ± 0.01	$1-\epsilon = 0.99$
	Safety Prob.	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	0.974 ± 0.01	0.000 ± 0.00	
road_2d	Return	23.9 ± 0.26	24.0 ± 0.22*	24.0 ± 0.22*	24.0 ± 0.09	**24.1 ± 0.02**	$1-\epsilon = 0.99$
	Safety Prob.	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	0.969 ± 0.01	0.073 ± 0.07	
Hopper-v5	Return	1000 ± 0*	1000 ± 0*	1000 ± 0*	**1005 ± 9.98**	624 ± 162	$1-\epsilon = 0.90$
	Safety Prob.	**1.000 ± 0.00**	**1.000 ± 0.00**	**1.000 ± 0.00**	0.976 ± 0.0150	0.00 ± 0.00	

baseline is useful to monitor the learning progress of the underlying policy. Results in Fig. 2 show that A2C-GP-Shield converges substantially faster than both CPO and PPO-Lag: in `cartpole`, $\widehat{\pi}$ reaches optimal performance more than one order of magnitude quicker, while in `mountain_car` it achieves rapid convergence after the n GP models become well calibrated. However, during early training, imperfect calibration leads to temporary safety violations. At the end of training, even though the underlying policy $\widehat{\pi}$ is unsafe while optimizing for rewards (demonstrating a clear trade-off), the shield prevents safety violations entirely. Full learning curves, including override statistics are provided in the technical appendix.

Higher-dimensional systems. We further evaluated our approach on the `Hopper-v5` environment from Gymnasium [50], a standard MuJoCo [49] benchmark featuring a simulated 2D robot that must hop forward without falling. The observation and action spaces are $n = 11$ and $m = 3$, respectively, with rewards consisting of a healthy survival bonus and forward velocity term. Each episode is limited to $T = 1000$ timesteps. We designed a linear feedback controller around the nominal equilibrium configuration using discrete-time LQR, with (A, B) matrices obtained from a finite-difference linearization of the MuJoCo dynamics (see the technical appendix). The corresponding control invariant set was computed via ellipsoidal expansion and convex-hull verification. With $\epsilon_t = 10^{-4}$ and recovery horizon $H = 100$, the shield intervened at every step, maintaining perfect safety and achieving the maximum healthy reward of +1000.0 as detailed in Tab. 1. These results demonstrate that while our framework is theoretically sound and guarantees safety by construction, its practical performance depends critically on the quality of the backup controller and the size of the verified control invariant set. For complex high-dimensional systems with



(a) `cartpole`



(b) `mountain_car`

Figure 2: Learning curves (log scale on the x-axis).

contact dynamics, limit cycles, and other non-linear effects, verifying sufficiently large invariant sets remains an open challenge orthogonal to our approach. Developing richer or learned backup controllers capable of stabilizing over larger regions may therefore be essential for extending recovery-based shielding to such settings.

7 RELATED WORK

Constrained RL. The most widely adopted formulation for safe RL is the CMDP framework [4]. In a CMDP, the agent optimizes

its policy to maximize reward while satisfying a cost constraint (typically an expected cost or chance constraint) at each step or over each episode. A number of safe RL algorithms build on this framework, for example by using Lagrange relaxations [7, 41], or policy projection (e.g., *Constrained Policy Optimization* [1]). While CMDP-based methods can improve the average safety of learning, they provide only indirect convergence guarantees, they often ensure constraints in expectation or asymptotically, which means they may still allow occasional, potentially dangerous violations during training. There exists GP methods that fall within this category, i.e., which model an unknown safety cost function through GPs [55, 56]. These methods are not without their limitations, making often unrealistic assumptions, e.g., assuming known deterministic dynamics or access to a privileged "emergency reset" button [54]. In summary, CMDP guarantees tend to be *weaker* and not suited for highly safety-critical environments [51].

Shielding. In contrast to CMDP methods, *shielding* often offers more rigorous guarantees. For RL, shielding was first introduced by Alshiekh et al. [3] for safeguarding against formal logic specifications in discrete-state systems. Provided with a suitable (logical) safety-abstraction of the environment, shielding filters out unsafe actions proposed by the agent, enjoying key properties, such as *correctness* and *minimal interference*. For stochastic environments, this approach has recently been extended to optimality preserving *probabilistic* guarantees for generic *reachability properties* [26]. In continuous control domains, implementing shielding requires computing a safe region of the state space and a policy to keep the agent within it. One line of work employs Hamilton-Jacobi (HJ) reachability analysis [2] to derive the maximal safe set and a corresponding safe controller. Fisac et al. [19] generalized this idea to RL with uncertain dynamics by formulating a safety value function (via an HJ partial differential equation) that triggers an override when the agent is about to exit the safe set. HJ-based shielding methods offer strong safety guarantees for nonlinear systems, including the ability to handle non-convex state constraints [52]. However, as noted earlier, they suffer from poor scalability. To improve scalability, other works focus on computing control invariant sets (often convex approximations of the safe region) for complex dynamics [10, 11, 34]. For instance, robust invariant set computation techniques have been applied to design shields that work for certain classes of non-linear systems [34]. Furthermore, The concept of shielding has also been extended to higher dimensional systems with learned *world models* [21, 27], although these methods come with some theory [21] they often any lack strict guarantees. Our work addresses this more general case where the system dynamics are initially unknown, requiring the agent to learn them, and thus we build on the concept of shielding combined with online model learning. Sampling based techniques, e.g., *Statistical MPS* [11], can be used for stochastic models of the environment; for a step-wise safety probability ϵ_t and confidence δ_t Bastani et al. [11] show that the required number of samples satisfies, $K(\epsilon_t, \delta) \geq \frac{\log(1/\delta_t)}{\log(1/(1-\epsilon_t))} + 1$. Substituting $\epsilon_t = 10^{-5}$ (which was used in our experiments) and $\delta_t = 0.01$ yields $K \approx 46{,}000$ samples per check, which is computationally infeasible, even with approximate sampling regimes [37]. Notably, our method does away with the confidence δ_t (essentially absorbing it into the GP learning) and as discussed earlier can handle arbitrarily small ϵ_t without additional overhead.

Gaussian Process (GP) models have long been used in RL for data-efficient learning of dynamics and rewards. PILCO [17] is a landmark model-based RL approach that uses GP dynamics models to achieve impressive sample efficiency. However, PILCO's original formulation had no safety considerations. An extension by Polymenakos et al. [39] incorporated a safety constraint into the PILCO framework; this approach is limited by considering only a small, specific class of policies (e.g., linear and RBF) and did not provide rigorous safety guarantees for those policies. More generally, GPs have been integrated into safe RL algorithms primarily through the CMDP lens discussed above [54–56]. A different approach is to use learned models within a control-theoretic safety scheme. *Learning-based Model Predictive Control* (MPC) has been explored as a way to enforce safety during RL. Koller et al. [30] pioneered a GP-based MPC for safe exploration, which uses GPs to construct high-confidence bounds on the system's behaviour and then optimizes a control sequence that satisfies state constraints with high probability. Although, their framework only considers optimization over a sequence of time-dependent linear controllers. Similar work from Cheng et al. [16] consider control barrier functions (CBF) with GP uncertainty estimates, solving a QP for high-probability guarantees. More recently, Zhao et al. [58] proposed a *Probabilistic Safeguard* framework that also leverages GP models for safety. In their approach, the agent first gathers an *offline dataset* of safe trajectories to train a GP model, then defines a *safety index* (analogous to a CBF). This safeguard method shares our philosophy of combining learning with a safety filter; however, it relies on designing an appropriate safety index and primarily addresses the offline training phase, whereas our work emphasizes online safe exploration and learning.

8 CONCLUSION

We proposed a recovery-based shielding framework that integrates Gaussian process (GP) dynamics models with formal safety semantics to guarantee a probabilistic lower bound on safety. Our approach propagates analytic uncertainty sets through the GP model and leverages a precomputed backup policy to certify ϵ-recoverability of states. This expands the set of *operationally safe* states beyond the control invariant set, while avoiding sampling-based verification and thus remaining efficient even under strict safety requirements. Our implementation, A2C-GP-Shield, supports both SGPR [48] and SVGP [28], enabling scalable training with large replay buffers. Nonetheless, GP inference can become computationally demanding in higher dimensions (e.g., $n > 20$), suggesting that deep kernel learning may be a promising direction for future work. Extending our framework to handle moving or dynamic obstacles is another natural step: while our method applies directly when obstacle dynamics are known, learning or estimating unknown obstacle dynamics presents an open challenge. Finally, removing the reliance on an a priori backup policy, by computing or optimizing the backup controller online, could make the framework more broadly applicable in scenarios where such a controller is unavailable.

ACKNOWLEDGMENTS

The research described in this paper was partially supported by the EPSRC (grant number EP/X015823/1).

REFERENCES

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

[2] Anayo K Akametalu, Jaime F Fisac, Jeremy H Gillula, Shahab Kaynama, Melanie N Zeilinger, and Claire J Tomlin. 2014. Reachability-based safe learning with Gaussian processes. In *53rd IEEE conference on decision and control*. IEEE, 1424–1431.

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

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

[5] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565* (2016).

[6] Brian DO Anderson and John B Moore. 2007. *Optimal control: linear quadratic methods*. Courier Corporation.

[7] Yarden As, Ilnura Usmanova, Sebastian Curi, and Andreas Krause. 2022. Constrained policy optimization via bayesian world models. *arXiv preprint arXiv:2201.09802* (2022).

[8] Arko Banerjee, Kia Rahmani, Joydeep Biswas, and Isil Dillig. 2024. Dynamic Model Predictive Shielding for Provably Safe Reinforcement Learning. *arXiv preprint arXiv:2405.13863* (2024).

[9] Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson. 1983. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics* SMC-13, 5 (1983), 834–846. https://doi.org/10.1109/TSMC.1983.6313077

[10] Osbert Bastani. 2021. Safe reinforcement learning with nonlinear dynamics via model predictive shielding. In *2021 American control conference (ACC)*. IEEE, 3488–3494.

[11] Osbert Bastani, Shuo Li, and Anton Xu. 2021. Safe Reinforcement Learning via Statistical Model Predictive Shielding.. In *Robotics: Science and Systems*. 1–13.

[12] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. 2017. Safe model-based reinforcement learning with stability guarantees. *Advances in neural information processing systems* 30 (2017).

[13] Roderick Bloem, Bettina Könighofer, Robert Könighofer, and Chao Wang. 2015. Shield synthesis: Runtime enforcement for reactive systems. In *International conference on tools and algorithms for the construction and analysis of systems*. Springer, 533–548.

[14] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. 2018. JAX: composable transformations of Python+NumPy programs. http://github.com/jax-ml/jax

[15] Joaquin Quinonero Candela, Agathe Girard, Jan Larsen, and Carl Edward Rasmussen. 2003. Propagation of uncertainty in bayesian kernel models-application to multiple-step ahead forecasting. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, Vol. 2. IEEE, II–701.

[16] Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. 2019. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 3387–3395.

[17] Marc Deisenroth and Carl E Rasmussen. 2011. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*. 465–472.

[18] Marc Peter Deisenroth. 2010. *Efficient reinforcement learning using Gaussian processes*. Vol. 9. KIT Scientific Publishing.

[19] Jaime F Fisac, Anayo K Akametalu, Melanie N Zeilinger, Shahab Kaynama, Jeremy Gillula, and Claire J Tomlin. 2018. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Trans. Automat. Control* 64, 7 (2018), 2737–2752.

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

[21] Alexander W Goodall and Francesco Belardinelli. 2023. Approximate Model-Based Shielding for Safe Reinforcement Learning. In *ECAI 2023*. IOS Press, 883–890.

[22] 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*. 2291–2293.

[23] Sven Gronauer, Tom Haider, Felippe Schmoeller da Roza, and Klaus Diepold. 2024. Reinforcement Learning with Ensemble Model Predictive Safety Certification. *arXiv preprint arXiv:2402.04182* (2024).

[24] Felix Gruber and Matthias Althoff. 2020. Computing safe sets of linear sampled-data systems. *IEEE Control Systems Letters* 5, 2 (2020), 385–390.

[25] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. 2023. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104* (2023).

[26] Edwin Hamel-De le Court, Francesco Belardinelli, and Alexander W Goodall. 2025. Probabilistic Shielding for Safe Reinforcement Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 16091–16099.

[27] Chloe He, Borja G León, and Francesco Belardinelli. 2022. Do Androids Dream of Electric Fences? Safety-Aware Reinforcement Learning with Latent Shielding. (2022). https://ceur-ws.org/Vol-3087/paper_50.pdf

[28] James Hensman, Alexander G Matthews, Maurizio Filippone, and Zoubin Ghahramani. 2015. MCMC for Variationally Sparse Gaussian Processes. In *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (Eds.), Vol. 28. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2015/file/6b180037abbebea991d8b1232f8a8ca9-Paper.pdf

[29] Niklas Kochdumper, Felix Gruber, Bastian Schürmann, Victor Gaßmann, Moritz Klischat, and Matthias Althoff. 2021. AROC: A toolbox for automated reachset optimal controller synthesis. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*. 1–6.

[30] Torsten Koller, Felix Berkenkamp, Matteo Turchetta, and Andreas Krause. 2018. Learning-based model predictive control for safe exploration. In *2018 IEEE conference on decision and control (CDC)*. IEEE, 6059–6066.

[31] Bettina Könighofer, Julian Rudolf, Alexander Palmisano, Martin Tappler, and Roderick Bloem. 2021. Online shielding for stochastic systems. In *NASA Formal Methods Symposium*. Springer, 231–248.

[32] Dieter Kraft. 1988. A software package for sequential quadratic programming. *Forschungsbericht- Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt* (1988).

[33] Hanna Krasowski, Jakob Thumm, Marlon Müller, Lukas Schäfer, Xiao Wang, and Matthias Althoff. 2023. Provably safe reinforcement learning: Conceptual analysis, survey, and benchmarking. *Transactions on Machine Learning Research* (2023).

[34] Shuo Li and Osbert Bastani. 2020. Robust model predictive shielding for safe reinforcement learning with stochastic dynamics. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 7166–7172.

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

[36] Andrew William Moore. 1990. *Efficient Memory-based Learning for Robot Control*. Technical Report. University of Cambridge.

[37] Thomas Pinder and Daniel Dodd. 2022. GPJax: A Gaussian Process Framework in JAX. *Journal of Open Source Software* 7, 75 (2022), 4455. https://doi.org/10.21105/joss.04455

[38] Boris T Polyak and Anatoli B Juditsky. 1992. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization* 30, 4 (1992), 838–855.

[39] Kyriakos Polymenakos, Alessandro Abate, and Stephen Roberts. 2019. Safe policy search using Gaussian process models. In *Proceedings of the 18th international conference on autonomous agents and multiagent systems*. 1565–1573.

[40] Carl Edward Rasmussen. 2003. Gaussian processes in machine learning. In *Summer school on machine learning*. Springer, 63–71.

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

[42] Lukas Schäfer, Felix Gruber, and Matthias Althoff. 2023. Scalable computation of robust control invariant sets of nonlinear systems. *IEEE Trans. Automat. Control* 69, 2 (2023), 755–770.

[43] Tom Schaul. 2015. Prioritized Experience Replay. *arXiv preprint arXiv:1511.05952* (2015).

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

[45] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* (2015).

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

[47] Richard S Sutton. 2018. Reinforcement learning: An introduction. *A Bradford Book* (2018).

[48] Michalis Titsias. 2009. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*. PMLR, 567–574.

[49] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 5026–5033.

[50] Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. 2024. Gymnasium: A Standard Interface for Reinforcement Learning Environments. *arXiv preprint arXiv:2407.17032* (2024).

[51] Cameron Voloshin, Hoang Le, Swarat Chaudhuri, and Yisong Yue. 2022. Policy optimization with linear temporal logic constraints. *Advances in Neural Information Processing Systems* 35 (2022), 17690–17702.

[52] Kim P Wabersich, Andrew J Taylor, Jason J Choi, Koushil Sreenath, Claire J Tomlin, Aaron D Ames, and Melanie N Zeilinger. 2023. Data-driven safety filters: Hamilton-jacobi reachability, control barrier functions, and predictive methods for uncertain systems. *IEEE Control Systems Magazine* 43, 5 (2023), 137–177.

[53] Kim P Wabersich and Melanie N Zeilinger. 2018. Linear model predictive safety certification for learning-based control. In *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 7130–7135.

[54] Akifumi Wachi, Wataru Hashimoto, Xun Shen, and Kazumune Hashimoto. 2024. Safe exploration in reinforcement learning: A generalized formulation and algorithms. *Advances in Neural Information Processing Systems* 36 (2024).

[55] Akifumi Wachi and Yanan Sui. 2020. Safe reinforcement learning in constrained markov decision processes. In *International Conference on Machine Learning*. PMLR, 9797–9806.

[56] Akifumi Wachi, Yanan Sui, Yisong Yue, and Masahiro Ono. 2018. Safe exploration and optimization of constrained mdps using gaussian processes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[57] Yixuan Wang, Simon Sinong Zhan, Ruochen Jiao, Zhilu Wang, Wanxin Jin, Zhuoran Yang, Zhaoran Wang, Chao Huang, and Qi Zhu. 2023. Enforcing hard constraints with soft barriers: Safe reinforcement learning in unknown stochastic environments. In *International Conference on Machine Learning*. PMLR, 36593–36604.

[58] Weiye Zhao, Tairan He, and Changliu Liu. 2023. Probabilistic safeguard for reinforcement learning using safety index guided gaussian process models. In *Learning for Dynamics and Control Conference*. PMLR, 783–796.