Approximate Shielding of Atari Agents for Safe Exploration

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

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

ABSTRACT

Balancing exploration and conservatism in the constrained setting is an important problem if we are to use reinforcement learning for meaningful tasks in the real world. In this paper, we propose a principled algorithm for *safe exploration* based on the concept of *shielding*. Previous approaches to shielding assume access to a safety-relevant abstraction of the environment or a high-fidelity simulator. Instead, our work is based on *latent shielding* - another approach that leverages world models to verify policy roll-outs in the latent space of a learned dynamics model. Our novel algorithm builds on this previous work, using safety critics and other additional features to improve the stability and farsightedness of the algorithm. We demonstrate the effectiveness of our approach by running experiments on a small set of Atari games with state dependent safety labels. We present preliminary results that show our approximate shielding algorithm effectively reduces the rate of safety violations, and in some cases improves the speed of convergence and quality of the final agent.

KEYWORDS

Safe Reinforcement Learning, Formal Verification, World Models

1 INTRODUCTION

Reinforcement learning (RL) [53] has become a principled and powerful tool for training agents to complete tasks in complex and dynamic environments. While RL promises a lot in theory, it unfortunately comes with no guarantees on worst-case performance. In safety-critical applications such as healthcare, robotics, autonomous driving and industrial control systems, it is imperative that decision making algorithms avoid unsafe or harmful situations [6]. Formal verification [10] poses as a mathematically precise technique for verifying system performance and can be used to verify that learned policies respect safety-constraints during training and deployment.

Recently there has been increasing interest in applying model-based RL (MBRL) algorithms in the constrained setting. This increase in interest can be attributed in part to exciting developments in MBRL [30, 31] and the superior sample complexity of model-based approaches [28, 34]. With better sample-complexity, MBRL algorithms should in theory commit far fewer safety violations during training than their model-free counterparts. This is important in the problem of *safe exploration* [6] where collecting experience is costly and unsafe behaviour can lead to catastrophic consequences in the real world.

In this work we focus on a method for safe exploration called *shielding* [3, 35]. In its original form, shielding forces hard constraints on the actions performed by the agent to ensure that the agent stays within a verified boundary on the state space. To compute this boundary we typically require a safety-relevant abstraction of the environment that is compact enough to efficiently perform exact verification techniques. Instead we opt to be less restrictive and make minimal assumptions about what we have access to a priori. As in previous work [32], we only assume that there exists some expert labelling of the states and we do not have access to a compact model or a safety-relevant abstraction of the environment. The key motivation for making these minimal assumptions is to obtain a more general algorithm that can be applied in many real-life applications where an abstraction is typically not available, as the system might be too complex or unknown in advance.

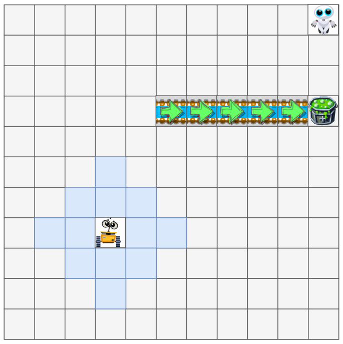


Figure 1: Simple grid-world with goal (Eve) in the top right corner. With a Manhattan distance look-ahead of 2 (blue squares) Wall-E is forever doomed to fall into the acid during exploration, as he is unable to determine that the conveyor belt leads to an unavoidable unsafe state. With safety critics Wall-E can learn the cost value of the conveyor belt squares and avoid them without a further look-ahead horizon. [*]

[*] This image was created with the assistance of DALL·E 2

Bounded prescience shielding (BPS) [25] is an approach to shielding that removes the requirement of access to a compact representation or abstraction of the environment. Instead, BPS assumes access to a black-box simulator of the environment which can be queried for look-ahead shielding of the learned policy. Giacobbe et al. demonstrated that pre-trained state-of-the-art model-free Atari agents consistently violate safety-constraints provided by domain experts. And that BPS with a look-ahead horizon of $H = 5$ reduced the rate of several shallow safety properties. In this paper we use the same state-dependent safety-labels for Atari games provided by Giacobbe et al., although we note that our approach has several distinct advantages over BPS: (1) we do not assume access to a black-box model to assist decision making, (2) we are able to apply our shielding algorithm during training without substantial

Proc. of the Adaptive and Learning Agents Workshop (ALA 2023), Cruz, Hayes, Wang, Yates (eds.), May 29-30, 2023, London, UK, https://alaworkshop2023.github.io/. 2023.

computational overhead, (3) we are able to look-ahead further into the future (> 5) with deeper model roll-outs and safety critics.

In a similar fashion to *latent shielding* [32], we use world models [27, 28, 30] to learn a dynamics model of the environment for policy optimisation and approximate shielding. The key differences between *latent shielding* and our approach are outlined in Section 3. Most notably we utilise safety critics, which are crucial for obtaining further look-ahead ability without explicitly increasing the shielding horizon, see Fig. 1.

Contributions. Our main contributions are summarised as follows: (1) we augment *latent shielding* [32] with safety critics used to bootstrap the end of imagined trajectories for look-ahead shielding further into the future, (2) we use twin delayed target critics to reduce the overestimation of expected costs and reduce overly conservative behaviour, (3) we ground our approach in a logical formalism, namely probabilistic computation tree logic (PCTL) [10], (4) we derive PAC-style bounds on the probability of accurately estimating a constraint violation under the assumption of perfect transition dynamics, (5) we empirically show that our approach dramatically reduces the rate of safety violations on a small set of Atari games with state-dependent safety labels and in some cases our algorithm greatly improves the speed of convergence and quality of the learned policy with respect to accumulated reward.

2 PRELIMINARIES

In this section we describe the relevant background material and notation required to understand the main results of this paper. We start by introducing the problem setup and the specification language used to formalise the notion of safety used throughout this paper. We then continue with an outline of the world model components and the policy optimisation scheme.

2.1 Problem Setup

Atari games in the Arcade Learning Environment (ALE) [42] are built on top of the Atari 2600 Stella emulator. The emulator manipulates 128-bytes of RAM which represent the underlying state of the game. However, Agents typically only observe $3 \times 210 \times 160$ dimensional tensors representing each of the pixel values of the screen. Therefore, we model the system as a partially observed Markov decision processes (POMDP) [47], which in this case is more appropriate than the traditional MDP formulation.

For our purposes we also extend the POMDP tuple with an additional labelling function [10]. Formally, we define a POMDP as a tuple $\mathcal{M} = (S, A, p, \iota_{init}, R, \Omega, O, AP, L)$ where, S is a finite set of states, A is a finite set of actions, $p : S \times A \times S \to [0, 1]$ is the probabilistic state-action transition function, $\iota_{init} : S \to [0, 1]$ is the initial state distribution such that $\sum_{s \in S} \iota_{init}(s) = 1$, $R : S \times A \to \mathbb{R}$ is the reward function, Ω is a finite set of observations, $O : S \times A \times \Omega \to [0, 1]$ is the observation probabilistic function, which defines the probability of an observation conditional on the previous state-action pair, AP is a set of atomic propositions which maps to the set of states by an 'expert' labelling function $L : S \to 2^{AP}$.

In particular, at each timestep t the agent receives an observation $o_t \in \Omega$, a reward r_t and a set of labels $L(s_t) \in 2^{AP}$. Given some state formula Φ, the agent can determine if the underlying state s_t satisfies Φ with the following relation,

$$\begin{aligned} s &\models \text{true for all } s \in S \\ s &\models a \quad \text{iff} \quad a \in L(s) \\ s &\models \neg\Phi \quad \text{iff} \quad s \not\models \Phi \\ s &\models \Phi_1 \wedge \Phi_2 \quad \text{iff} \quad s \models \Phi_1 \wedge s \models \Phi_2 \end{aligned}$$

The goal is to find a policy π that maximises expected reward, i.e. $\pi^* = \arg\max_\pi \mathbb{E}[\sum_{t=1}^{\infty} \gamma^{t-1} R(s_t, \pi(s_t))]$, while minimising violations of the state formula Φ (that encodes the safety-constraints) during training. Here γ is the discount factor [53].

2.2 Probabilistic Computation Tree Logic

Probabilistic computation tree logic (PCTL) is a branching time temporal logic that extends CTL with probabilistic quantifiers [10]. PCTL is particularly useful for specifying reachability and safety properties for discrete stochastic systems which makes it useful for our purposes. A valid PCTL formula can be constructed as follows,

$$\begin{aligned} \Phi &::= \text{true} \mid a \mid \neg\Phi \mid \Phi \wedge \Phi \mid \mathbb{P}_J(\phi) \\ \phi &::= X\Phi \mid \Phi U \Phi \mid \Phi U^{\leq n} \Phi \end{aligned}$$

where $a \in AP$ is an atomic proposition, negation ($\neg$) and conjunction ($\wedge$) are the familiar logical operators, $J \subset [0, 1]$, $J \neq \emptyset$ is a non-empty subset of the unit interval, and next (X), until (U) and bounded until ($U^{\leq n}$) are temporal operators. We distinguish here between state formula Φ and path formula ϕ which are interpreted over states and paths respectively.

We write $s \models \Phi$ to indicate that a state s satisfies a state formula Φ, where the satisfaction relation is defined as before, see [10] for details. Similarly, we can define the satisfaction relation for path formula ϕ, this is given in the next section for the specific fragment of PCTL that we require. Also note that the common operators eventually ($\Diamond$) and always ($\Box$) and their bounded counter parts ($\Diamond^{\leq n}$ and $\Box^{\leq n}$) can be defined in a familiar way, see [10].

The reason behind using PCTL as our safety specification language is because it allows us to meaningfully trade-off safety and progress by specifying the probability with which we force the agent to satisfy to a given temporal logic formula.

2.3 Bounded Safety

The notion of bounded safety for Atari agents introduced by Giacobbe et al. can be straightforwardly grounded in PCTL. Consider some fixed (stochastic) policy $\pi : O \times A \to [0, 1]$ and POMDP $\mathcal{M} = (S, A, p, \iota_{init}, R, \Omega, O, AP, L)$. Together π and $\mathcal{M}$ define a transition system $\mathcal{T} : S \times S \to [0, 1]$, where $\sum_{s' \in S} \mathcal{T}(s, s') = 1$. A finite trace with length n of the transition system $\mathcal{T}$, is a sequence of states $s_0 \to s_1 \to ... \to s_n$ denoted τ, the i^{th} state of τ is given by $\tau[i]$. A trace τ satisfies bounded safety if and only if all of its states satisfy the state formula Φ that encodes the safety constraints. Formally,

$$\tau \models \Box^{\leq n}\Phi \quad \text{iff} \quad \text{for all } 0 \leq i \leq n, \tau[i] \models \Phi \tag{1}$$

for some bounded look-ahead n. Now in PCTL we can say that a state $s \in S$ satisfies ε-bounded safety as follows,

$$\begin{aligned} &s \models \mathbb{P}_{1-\varepsilon}(\Box^{\leq n}\Phi) \;\; \text{iff} \\ &\quad \mu_s(\{\tau \mid \tau[0] = s, \text{ for all } 0 \leq i \leq n, \tau[i] \models \Phi\}) \in [1 - \varepsilon, 1] \end{aligned} \tag{2}$$

where μ_s is a well-defined probability measure induced by the transition system $\mathcal{T}$, over the set of traces staring from s and with finite length n, see [10] for details. We denote $\mu_{s \models \phi}$ as shorthand for the measure $\mu_s(\{\tau \mid \tau[0] = s, \text{ for all } 0 \leq i \leq n, \tau[i] \models \Phi\})$, where $\phi ::= \square^{\leq n}\Phi$ is the path formula we care about. By framing bounded safety in this way, we obtain a meaningful way to trade off safety and progress with the ε parameter.

2.4 World Models

To learn a world model for behaviour learning and look-ahead shielding we leverage DreamerV2 [30], which was used to master Atari games in the ALE [42]. DreamerV2 is composed of the following components: an image encoder $z_t \sim q_\theta(z_t \mid o_t, h_t)$ that learns a posterior latent representation conditional on the current observation o_t and recurrent state h_t, the recurrent state space model (RSSM) [29] which is a mixture of deterministic and stochastic categorical latents, and the image, reward and discount predictors.

The RSSM consists of two main components: the recurrent model $h_t = f_\theta(h_{t-1}, z_{t-1}, a_{t-1})$, which computes the next deterministic latents given the past state $s_{t-1} = (h_{t-1}, z_{t-1})$ and action a_{t-1}, and the transition predictor $\hat{z}_t \sim p_\theta(\hat{z}_t \mid h_t)$, which is used as the prior distribution over the stochastic latents conditional on the deterministic latents.

The image predictor or decoder $\hat{o}_t \sim p_\theta(\hat{o}_t \mid h_t, z_t)$ is trained to predict the current observation o_t with a reconstruction loss. The image predictor provides useful self-supervised gradients that help the world model learn a structured latent space for effective policy optimisation [30]. The reward predictor $\hat{r}_t \sim p_\theta(\hat{r}_t \mid h_t, z_t)$ and discount predictor $\hat{\gamma}_t \sim p_\theta(\hat{\gamma}_t \mid h_t, z_t)$, also provide useful self-supervised gradients. However, they are primarily used to construct targets for policy optimisation.

All components of the world model are implemented as neural networks and jointly trained with backpropagation and straight through gradients [12]. In addition, KL-balancing [30] is used to stop the prior and posterior being regularised at the same rate to prevent instability during training.

2.5 Behaviour Learning

In DreamerV2 [30], policy optimisation is performed entirely on experience 'imagined' by rolling out the world model with a fixed (stochastic) policy. A replay buffer $\mathcal{D}$ is used to retain experience from the real environment. At each training step a batch B is sampled from the replay buffer $\mathcal{D}$ and the RSSM is used to sample sequences of compact latent states $\hat{s}_{1:H}$, using each of the observations in B as a starting point. Here H refers to the 'imagination' horizon, which is typically set to a relatively small number ($H = 15$) to avoid compounding model errors that are likely to harm the learned policy.

The task policy π^{task} parameterised by ψ^{task} is trained to maximise accumulated reward. In addition, a task critic v^{task} parameterised by ξ^{task} is used to guide the learning of the policy. TD-λ targets [53] are constructed by rolling out the world model with the task policy π^{task},

$$V_t^{\text{task},\lambda} = \hat{r}_t + \hat{\gamma}_t \begin{cases} (1-\lambda)v^{\text{task}}(\hat{s}_{t+1}) + \lambda V_{t+1}^{\text{task},\lambda} & \text{if } t < H, \\ v^{\text{task}}(\hat{s}_H) & \text{if } t = H \end{cases} \tag{3}$$

The λ parameter trades of the bias and variance of the estimate, with $\lambda = 0.0$ giving the high variance n-step Monte-Carlo return and $\lambda = 1.0$ giving the high bias one-step return. The task critic v^{task} is regressed towards the value estimates with the following loss function,

$$\mathcal{L}(\xi^{\text{task}}) = \mathbb{E}_{\pi^{\text{task}}, p_\theta}\left[\sum_{t=1}^{H-1} \frac{1}{2}(v^{\text{task}}(\hat{s}_t) - sg(V_t^{\text{task},\lambda}))^2\right] \tag{4}$$

where the $sg(\cdot)$ operator stops the flow of gradients to the input argument. The task policy π^{task} is trained with reinforce gradients [53] and an entropy regulariser to encourage exploration. In addition, the difference of the TD-λ targets $V^{\text{task},\lambda}$ and the critic estimates v^{task} are used as a baseline to reduce the variance of the reinforce gradients. This gives the following loss function for the task policy π^{task},

$$\mathcal{L}(\psi^{\text{task}}) = \mathbb{E}_{\pi^{\text{task}}, p_\theta}\left[\sum_{t=1}^{H-1} \underbrace{-\log \pi^{\text{task}}(a_t \mid \hat{s}_t) sg(V_t^{\text{task},\lambda} - v^{\text{task}}(\hat{s}_t))}_{\text{reinforce}} \underbrace{-\eta H(\pi^{\text{task}}(\cdot \mid \hat{s}_t))}_{\text{entropy}}\right] \tag{5}$$

3 APPROXIMATE SHIELDING

In this section we introduce our approximate shielding algorithm for Atari agents. The general idea is to learn a world model for task policy optimisation, safe policy synthesis and bounded look-ahead shielding. The world model of choice is DreamerV2 [30] which has demonstrated state-of-the-art performance on the Atari benchmark. While our approach is similar to *latent shielding* [32] we note that it has following key differences:

- We learn a cost predictor to estimate state dependent costs, rather than a labelling function $L_\vartheta : S \longrightarrow \{\text{safe}, \text{unsafe}\}$.
- We train a safe policy to minimise expected costs, which is used as the backup policy if a safety-violation is detected.
- We use safety critics to obtain further look-ahead capabilities without having to roll-out the world model further into the future.
- We don't need to use intrinsic punishment [3] or any sort of shield introduction schedule [32].
- We test our approach in a much more sophisticated domain, specifically the ALE [42].

In what follows, we describe the notable components used in our approach, followed by a precise description of the shielding procedure and an outline of the full learning algorithm. However, we will first present some PAC-style bounds on the probability of accurately predicting a constraint violation using our the shielding procedure. It should then become clear in what sense our algorithm approximate. Specifically, 'approximate' comes from the fact that we use a learned approximation of the true environment dynamics and we use Monte-Carlo estimation to predict constraint violations.

3.1 Probabilistic Guarantees

Recall that to ensure ε-bounded safety we are interested in verifying PCTL formula of the form $\mathbb{P}_{1-\varepsilon}(\square^{\leq n}\Phi)$, where Φ is the state formula that encodes the safety-constraints. To do so we fix the task policy π^{task} and the learned world model p_θ to obtain an approximate transition system $\widehat{\mathcal{T}} : S \times S \longrightarrow [0,1]$. Even with the 'true' transition system $\mathcal{T} : S \times S \longrightarrow [0,1]$, exact PCTL verification is $O(\text{poly}(size(\mathcal{T})) \cdot n \cdot |\Phi|)$, which is much too big for Atari games. Instead we rely on Monte-Carlo estimation of the measure $\mu_{s \models \phi}$ by sampling traces τ from the approximate transition system $\widehat{\mathcal{T}}$.

Proposition 3.1. *Given access to the 'true' transition system $\mathcal{T}$, with probability $1-\delta$ we can estimate the measure $\mu_{s\models\phi}$ up to some approximation error ϵ, by sampling m traces $\tau \sim \mathcal{T}$, provided,*

$$m \geq \frac{1}{2\epsilon^2} \log\left(\frac{2}{\delta}\right) \tag{6}$$

Proof. The proof is a straightforward application of Hoeffding's inequality. We can estimate $\mu_{s\models\phi}$ by sampling m traces $\langle \tau_j \rangle_{j=1}^{m}$ from $\mathcal{T}$. Let $X_1, ..., X_m$ be indicator r.v.s such that,

$$X_j = \begin{cases} 1 & \text{if } \tau_j \models \square^{\leq n}\Phi, \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Let,

$$\hat{\mu}_{s\models\phi} = \frac{1}{m}\sum_{j=1}^{m} X_j, \text{ where } \mathbb{E}_{\mathcal{T}}[\hat{\mu}_{s\models\phi}] = \mu_{s\models\phi} \tag{8}$$

Then by Hoeffding's inequality,

$$\mathbb{P}\left[|\hat{\mu}_{s\models\phi} - \mu_{s\models\phi}| \geq \epsilon\right] \leq 2\exp\left(-2m\epsilon^2\right)$$

Bounding the RHS from above with δ and rearranging completes the proof. □

With these probabilistic guarantees we can set m appropriately for some domain specific requirements. In the following proposition, we demonstrate how we may be sure that a given state s satisfies ε-bounded safety given our estimate $\hat{\mu}_{s\models\phi}$.

Proposition 3.2. *Suppose we have an estimate $\hat{\mu}_{s\models\phi} \in [\mu_{s\models\phi} - \epsilon, \mu_{s\models\phi} + \epsilon]$, if $\hat{\mu}_{s\models\phi} \in [1-\varepsilon+\epsilon, 1]$ then it must be the case that $\mu_{s\models\phi} \in [1-\varepsilon, 1]$ and that $s \models \mathbb{P}_{1-\varepsilon}(\square^{\leq n}\Phi)$.*

Note that ε is the bounded safety parameter used to trade-off exploration and progress and ϵ is the approximation error from Proposition 3.1.

Proof. Suppose $\hat{\mu}_{s\models\phi} \in [\mu_{s\models\phi} - \epsilon, \mu_{s\models\phi} + \epsilon]$, $\hat{\mu}_{s\models\phi} \in [1-\varepsilon+\epsilon, 1]$ and $\mu_{s\models\phi} \notin [1-\varepsilon, 1]$. Then $\hat{\mu}_{s\models\phi} - \mu_{s\models\phi} > \epsilon$ which contradicts $\hat{\mu}_{s\models\phi} \in [\mu_{s\models\phi} - \epsilon, \mu_{s\models\phi} + \epsilon]$. This implies that indeed $\mu_{s\models\phi} \in [1-\varepsilon, 1]$ and that $s \models \mathbb{P}_{1-\varepsilon}(\square^{\leq n}\Phi)$ by Eq. 2. □

It is important to note that checking the condition $\hat{\mu}_{s\models\phi} \in [1 - \varepsilon + \epsilon, 1]$ could lead to overly conservative behaviour, if ϵ is not very small. This is because for $\mu_{s\models\phi} \in [1-\varepsilon, 1-\varepsilon+\epsilon]$ we may falsely predict that $s \not\models \mathbb{P}_{1-\varepsilon}(\square^{\leq n}\Phi)$ with some probability up to $1-\delta$. Instead we could check that $\hat{\mu}_{s\models\phi} \in [1-\varepsilon-\epsilon, 1]$, although this may lead to overly permissive behaviour. In words, the former configuration admits no false positives and the latter admits no false negatives (with probability $1-\delta$). Either configuration can be used, although we opt for the former.

To get similar bounds for the approximate transition system $\widehat{\mathcal{T}}$ we can try to get a bound on the total variation (TV) distance between $\widehat{\mathcal{T}}$ and $\mathcal{T}$. However, this is left for future work.

3.2 RSSM with Costs

We augment the RSSM of DreamerV2 [30] with a cost predictor $\hat{c}_t \sim p_\theta(\hat{c}_t \mid h_t, z_t)$ used to predict state dependent costs and a safety-discount predictor $\hat{\gamma}_t^{\text{safe}} \sim p_\theta(\hat{\gamma}_t^{\text{safe}} \mid h_t, z_t)$ which is used to help improve the stability of the safety critics.

In the same fashion as the reward predictor $\hat{r}_t \sim p_\theta(\hat{r}_t \mid h_t, z_t)$ the cost predictor $\hat{c}_t \sim p_\theta(\hat{c}_t \mid h_t, z_t)$ parameterises a Gaussian distribution. We construct targets for the cost predictor as follows,

$$c_t = \begin{cases} 0, & \text{if } s_t \models \Phi \\ C, & \text{otherwise} \end{cases} \tag{9}$$

where s_t refers to the true underlying state of the environment, Φ is the state formula that encodes the safety-constraints, and $C > 0$ is an arbitrary hyperparameter that determines the cost incurred at a violating state. Using a cost predictor in this way allows the agent to distribute its uncertainty about a constraint violation over several consecutive states.

The safety-discount predictor $\hat{\gamma}_t^{\text{safe}} \sim p_\theta(\hat{\gamma}_t^{\text{safe}} \,|\, h_t, z_t)$ is a binary classifier trained, in a similar way, to predict if a state is violating or not. We construct targets for the safety-discount predictor as follows,

$$\gamma_t^{\text{safe}} = \begin{cases} \gamma, & \text{if } s_t \models \Phi \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

The key purpose of the safety-discount factor is to reduce the overestimation of the expected costs by the safety critics. Using the safety-discount predictions $\hat{\gamma}_t^{\text{safe}}$ to construct targets for the safety critics, instead of the usual discount predictions $\hat{\gamma}_t$, effectively transforms the MDP into one where violating states are terminal states. This means the safety critics should always be upper bounded by C. The full RSSM loss function can now be written as follows,

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{image}} + \mathcal{L}_{\text{reward}} + \mathcal{L}_{\text{discount}} + \mathcal{L}_{\text{cost}} + \mathcal{L}_{\text{safe-discount}} + \mathcal{L}_{\text{KL-B}} \tag{11}$$

3.3 Safe Policy

The safe policy π^{safe} is used as the backup policy if we detect that the task policy π^{task} is likely to commit a safety violation in the next T steps. Since we have no access to an abstraction of the environment we cannot synthesise a shield before training and so the safe policy must be learned.

The safe policy π^{safe} is only concerned with minimising expected costs and so we use the cost predictor $\hat{c}_t \sim p_\theta(\hat{c}_t | h_t, z_t)$ to construct TD-λ targets as follows,

$$V_t^{\text{safe},\lambda} = \hat{c}_t + \hat{\gamma}_t \begin{cases} (1-\lambda)v^{\text{safe}}(\hat{s}_{t+1}) + \lambda V_{t+1}^{\text{safe},\lambda} & \text{if } t < H, \\ v^{\text{safe}}(\hat{s}_H) & \text{if } t = H \end{cases} \tag{12}$$

The safe critic v^{safe} parameterised by ξ^{safe} is regressed towards the TD-λ targets with a similar loss function as before,

$$\mathcal{L}(\xi^{\text{safe}}) = \mathbb{E}_{\pi^{\text{safe}},p_\theta}\left[\sum_{t=1}^{H-1}\frac{1}{2}(v^{\text{safe}}(\hat{s}_t) - sg(V_t^{\text{safe},\lambda}))^2\right] \tag{13}$$

The safe policy π^{safe} parameterised by ψ^{safe} is also trained with biased reinforce gradients and an entropy regulariser as before,

$$\mathcal{L}(\psi^{\text{safe}}) = \mathbb{E}_{\pi^{\text{safe}},p_\theta}\left[\sum_{t=1}^{H-1}\underbrace{\log\pi^{\text{safe}}(a_t \mid \hat{s}_t)sg(V_t^{\text{safe},\lambda} - v^{\text{safe}}(\hat{s}_t))}_{\text{reinforce}} \underbrace{-\eta H(\pi^{\text{safe}}(\cdot \mid \hat{s}_t))}_{\text{entropy}}\right] \tag{14}$$

Note that the sign is flipped here, so that the safe policy π^{safe} minimises expected costs rather than maximises them.

3.4 Safety Critics

Safety critics estimate the expected costs under the task policy π^{task}. They give us an idea of how safe specific states are under the task policy state distribution. Additionally, we can use them to bootstrap the end of 'imagined' trajectories for further look-ahead capabilities.

To estimate the expected costs under the task policy π^{task} we use two safety critics v_1^C and v_2^C parameterised by ξ_1^C and ξ_2^C respectively. To prevent overestimation, the safety critics are jointly trained with a TD3-style algorithm [23] to estimate the following quantity,

$$\mathbb{E}_{\pi^{\text{task}},p_\theta}\left[\sum_{t=1}^{\infty}(\hat{\gamma}_t^{\text{safe}})^{t-1}\cdot\hat{c}_t\right] \tag{15}$$

Each of the two safety critics v_1^C and v_2^C, has its own target critic $v_1^{C'}$ and $v_2^{C'}$, that are updated periodically with slow updates. The TD-λ targets are constructed by taking a minimum of the two target critics $v_1^{C'}$ and $v_2^{C'}$ as follows,

$$V_t^{C,\lambda} = \hat{c}_t + \hat{\gamma}_t^{\text{safe}}\begin{cases}(1-\lambda)\min\{v_1^{C'}(\hat{s}_t), v_2^{C'}(\hat{s}_t)\} + \lambda V_{t+1}^{C,\lambda} & \text{if } t < H,\\ \min\{v_1^{C'}(\hat{s}_H), v_2^{C'}(\hat{s}_H)\} & \text{if } t = H\end{cases} \tag{16}$$

Both safety critics v_1^C and v_2^C are regressed towards the TD-λ targets with the following loss function,

$$\mathcal{L}(\xi^C, v^C) = \mathbb{E}_{\pi^{\text{task}},p_\theta}\left[\sum_{t=1}^{H-1}\frac{1}{2}(v^C(\hat{s}_t) - sg(V_t^{C,\lambda}))^2\right] \tag{17}$$

where $(\xi^C, v^C) \in \{(\xi_1^C, v_1^C), (\xi_2^C, v_2^C)\}$.

3.5 Algorithm

The full learning algorithm is split into two distinct phases: world model learning and policy optimisation (including safe policy synthesis and safety critic learning). To generate experience for world model learning we need to interact with the real environment and to mitigate safety violations in the real environment we pick actions with the shielded policy,

$$\pi^{\text{shield}}(\cdot \mid s) = \begin{cases}\pi^{\text{task}}(\cdot \mid s) & \text{if } \hat{\mu}_{s\models\phi} \in [1-\varepsilon+\epsilon, 1]\\ \pi^{\text{safe}}(\cdot \mid s) & \text{otherwise}\end{cases} \tag{18}$$

To estimate $\mu_{s\models\phi}$ we roll-out the world model p_θ with the task policy π^{task} to generate a batch of m sequences of compact latent states $\langle\hat{s}_{1:H}^{(i)}\rangle_{i=1}^m$. For each trace $\tau^{(i)} = \hat{s}_1^{(i)}, ..., \hat{s}_H^{(i)}$ we compute the discounted cost as follows,

$$\text{cost}(\tau^{(i)}) = \sum_{t=1}^{H}(\hat{\gamma}_t^{(i)})^{t-1}\cdot\hat{c}_t^{(i)} \tag{19}$$

Proposition 3.3. *Under the 'true' transition system $\mathcal{T}$ if cost$(\tau) < \gamma^{H-1}\cdot C$ then necessarily $\tau \models \square^{\leq H}\Phi$*

Proof. The proof is a straightforward argument. By construction $c_t = C$ if and only if $\tau[t] \not\models \Phi$, therefore $\text{cost}(\tau) < \gamma^{H-1}\cdot C$ implies that $\forall\ 1 \leq t \leq H\ c_t = 0$ which implies that $\forall\ 1 \leq t \leq H\ \tau[t] \models \Phi$. □

Using this idea, our estimate $\hat{\mu}_{s\models\phi} \approx \mu_{s\models\phi}$ is then computed as follows,

$$\hat{\mu}_{s\models\phi} = \frac{1}{m}\sum_{i=1}^{m}\mathbb{1}\left(\text{cost}(\tau^{(i)}) < \gamma^{H-1}\cdot C\right) \tag{20}$$

If we train safety critics then we can use the bootstrapped costs instead,

$$\text{b-cost}(\tau^{(i)}) = \left(\sum_{t=1}^{H-1}(\hat{\gamma}_t^{(i)})^{t-1}\cdot\hat{c}_t^{(i)}\right) + \min\left\{v_1^C(\hat{s}_H^{(i)}), v_2^C(\hat{s}_H^{(i)})\right\} \tag{21}$$

And we can estimate $\mu_{s\models\phi}$ with a larger horizon $T > H$, since the safety critics should capture the expected costs from $\hat{s}_H^{(i)}$ and beyond,

$$\hat{\mu}_{s\models\phi} = \frac{1}{m}\sum_{i=1}^{m}\mathbb{1}\left(\text{b-cost}(\tau^{(i)}) < \gamma^{T-1}\cdot C\right) \tag{22}$$

After several environment interactions with the shielding policy π^{shield}, a batch of data B is sampled from the replay buffer $\mathcal{D}$, for world model learning, task policy optimisation, safe policy optimisation and safety critic learning. The full algorithm is presented on the following page.

4 EXPERIMENTS

In this section we conduct a simple analysis and compare our algorithm, DreamerV2 with shielding, to DreamerV2 without shielding. We present results for two Atari games with state dependent labels: Assault and Seaquest (see Fig. 2). We start by giving a summary of the environments, followed by the experimental results and an accompanying discussion.

4.1 Assault

Assault is a fixed shooter game similar to *Space Invaders*. The goal is to shoot and destroy alien ships continuously deployed by a mothership. The smaller ships shoot lasers at the player which the player must avoid, otherwise they loose a life. In addition, the player's weapon can overheat if they fire too often, which also

Algorithm 1 DreamerV2 [30] with Shielding

```
Initialise: replay buffer $\mathcal{D}$ with $S$ random episodes.
Initialise: $\theta, \psi^{\text{task}}, \psi^{\text{safe}}, \xi^{\text{task}}, \xi^{\text{safe}}, \xi_1^C, \xi_2^C, \xi_1^{C'}, \xi_2^{C'}$ randomly.
  while not converged do
      // World model learning
      Sample $B \sim \mathcal{D}$.
      For every $o_t \in B$ compute sequences $\hat{s}_{t:t+H}$ with RSSM.
      Update RSSM parameters $\theta$ with Eq. 11
      // Task policy optimisation
      From every $o_t \in B$ imagine sequences $\hat{s}_{t:t+H}$ with $\pi^{\text{task}}$.
      Compute TD-$\lambda$ targets with Eq. 3.
      Update task critic parameters $\xi^{\text{task}}$ with Eq. 4.
      Update task policy parameters $\psi^{\text{task}}$ with Eq. 5.
      // Safety critic optimisation
      Compute safety critic targets with Eq. 16.
      Update safety critic parameters $\xi_1^C$ and $\xi_1^C$ with Eq. 17.
      For $i \in [1, 2]$ $\xi_i^{C'} \leftarrow \nu\xi_i^C + (1 - \nu)\xi_i^{C'}$ (soft update [23]).
      // Safe policy optimisation
      From every $o_t \in B$ imagine sequences $\hat{s}_{t:t+H}$ with $\pi^{\text{safe}}$.
      Compute TD-$\lambda$ targets with Eq. 12.
      Update safe critic parameters $\xi^{\text{safe}}$ with Eq. 13.
      Update safe policy parameters $\psi^{\text{safe}}$ with Eq. 14.
      // Environment interaction
      for $k = 1, ..., K$ do
          Observe $o_t$ from environment and compute $\hat{s}_t = (z_t, h_t)$.
          From $\hat{s}_t$ sample $m$ sequences $\langle \hat{s}_{1:H}^{(i)} \rangle_{i=1}^m$ with $\pi^{\text{task}}$.
          Estim. $\hat{\mu}_{s \models \phi} \approx \mu_{s \models \phi}$ with safety critics, Eq. 21 and Eq. 22.
          Play $a \sim \pi^{\text{shield}}(a \mid \hat{s}_t)$ and observe $r_t, o_{t+1}$ and $L(s_t)$.
          Construct $c_t$ with Eq. 9 and $\gamma_t^{\text{safe}}$ with Eq. 10.
          Append $\langle o_t, a_t, r_t, c_t, \gamma_t^{\text{safe}}, o_{t+1} \rangle$ to $\mathcal{D}$.
      end for
  end while
```

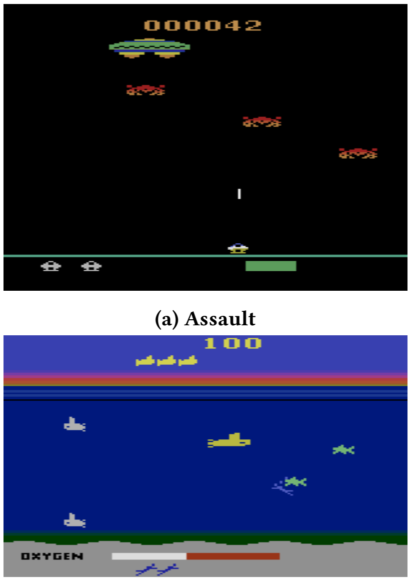
results in them loosing a life. The state dependent formula Φ that the agent aims to satisfy at each timestep is given as follows,

$$\Phi = \neg\textbf{hit} \wedge \neg\textbf{overheat} \tag{23}$$

where **hit** = true iff the player is hit by a laser and **overheat** = true iff the player's weapon overheats. We chose this environment because Giacobbe et al. demonstrated state-of-the-art agents only concerned with reward overheat the weapon frequently and that BPS [25] alleviated the issue to some degree. The idea is that when the task policy π^{task} is about to overheat the weapon the shield kicks in and the safe policy π^{safe} prevents the agent from firing the weapon while avoiding any incoming lasers.

4.2 Seaquest

Seaquest is an underwater shooter in which the player controls a submarine equipped with an infinite supply of missiles. The goal is to rescue divers, shoot enemy sharks and submarines, while managing a limited supply of oxygen and resurfacing when necessary. The player receives points and a full supply of oxygen when they surface with a diver on board and if they surface with six divers they are awarded additional points based on the amount of oxygen they have left. However, surfacing without any divers is not permitted and results in the player loosing a life. The state formula Φ for Seaquest is a little more involved and is defined as follows,



(a) Assault

(b) Seaquest

Figure 2: Screenshots from the two Atari environments.

$$\Phi = (\textbf{surface} \Rightarrow ((\textbf{diver} \wedge \textbf{low-oxygen}) \vee \textbf{very-low-oxygen} \vee \textbf{six-divers})) \wedge \neg\textbf{out-of-oxygen} \wedge \neg\textbf{hit} \tag{24}$$

where **surface** = true iff the submarine surfaces, **diver** = true iff the submarine has at least one diver on board, **low-oxygen** = true iff the players oxygen supply < 16, **very-low-oxygen** = true iff the players oxygen supply < 4, **six- divers** = true iff the submarine has six divers on board, **out-of-oxygen** = true iff the player runs out of oxygen and **hit** is defined similarly as before.

In words, it is only permissible to surface if the agent has a diver and is low on oxygen, has six divers or has very low on oxygen (with or without a diver). Surfacing with a diver when oxygen supplies are plentiful is deemed unsafe since it makes the game unnecessarily harder.

With Seaquest the agent needs to balance multiple objectives at once which is why it is a useful environment to test our approach. In our experiments we demonstrate that the learned safe policy π^{safe} is able to deal with a slightly more complex set of constraints and prevent the agent from making costly mistakes during training.

4.3 Training Details

The agents are trained on a single Nvidia Tesla A30 (24GB RAM) GPU and a 24-core/48 thread Intel Xeon CPU with 256GB RAM. Due to time constraints and limited compute resources all agents are trained on one seed and for precisely 40M frames on Atari environments provided by the ALE [11, 42].

All the hyperparameters for DreamerV2 are set as their default values for Atari games, which are given in [30]. Notably, for all experiments we set the imagination horizon $H = 15$, TD-λ discount $\lambda = 0.95$ and discount factor $\gamma = 0.999$.

The cost and safety-discount predictors are implemented as neural networks with identical architectures to the reward and discount predictors used in DreamerV2. The safe policy, critic and safety critics are also implemented as neural networks in the same way that the task policy and critic are implemented in DreamerV2. See [30] for all details. The shielding hyperparameters are also fixed in all experiments as follows, specifically we set the bounded safety parameter $\varepsilon = 0.1$, number of samples $m = 512$, approximation error $\epsilon = 0.09$[1], shield horizon $T = 30$ (2 seconds in real time) and safety critic smooth parameter updates $\nu = 0.005$.

4.4 Results

We evaluate our algorithm by comparing the performance of DreamerV2 [30] with and without shielding. Specifically, we compare the reward curves during training, the best episode return and the cumulative violations during training. Table 1 presents the best episode scores and total violations during training for DreamerV2 and DreamerV2 with shielding. In addition, Fig. 3 displays the learning curves for both algorithms.

Table 1: Best episode scores and cumulative violations for for DreamerV2 [30] and DreamerV2 with approximate shielding.

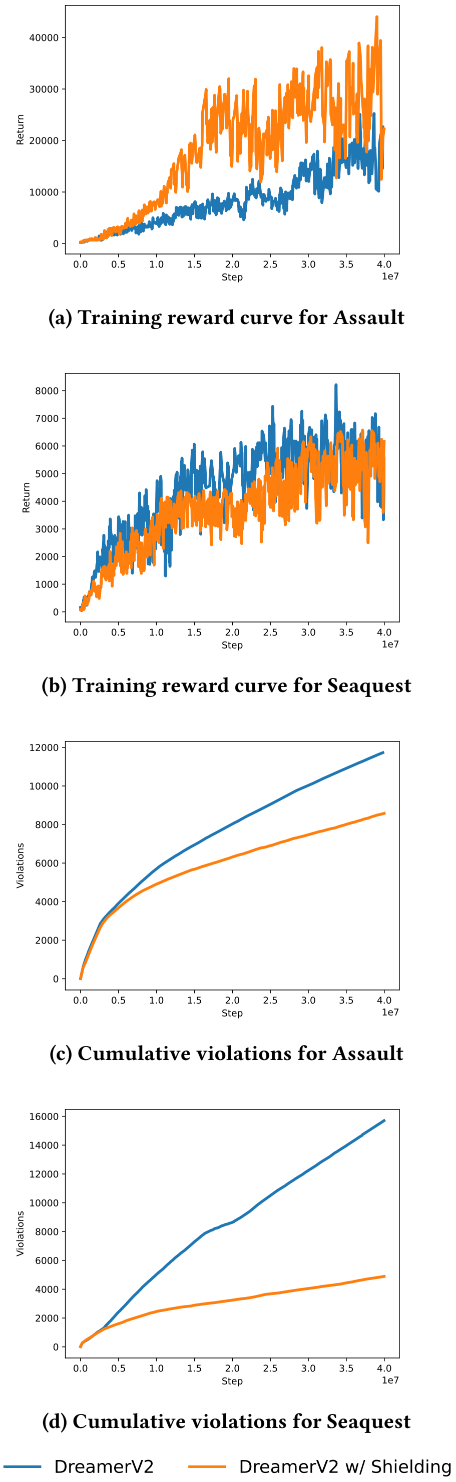
Env	*DreamerV2*		*DreamerV2 w/ Shielding*	
	Best Score	# Violations	Best Score	# Violations
Assault	34753	11726	**57504**	**8579**
Seaquest	**11400**	15697	7040	**4889**

Discussion. As seen in Table 1 and Fig. 3 our approximate shielding algorithm reduces the rate of safety violations for both *Assault* and *Seaquest.* In terms of reward, our shielding procedure has dramatically improved the speed of convergence for *Assault* and maintained comparable performance for *Seaquest.* We must note that these results are far from complete as we have not these run experiments over multiple random seeds or for the typical 200M frames, which is used as a common benchmark [30]. Nevertheless, we claim that our results provide compelling evidence that something is going on, which should motivate further investigation.

5 RELATED WORK

In this section we provide a discussion on the three main areas of research that our contribution is based on: world models, safe RL, and shielding.

World Models were first introduced by Ha and Schmidhuber [27] in a paper of the same name, although their inspiration is much more deeply rooted in psychology [55] and Bayesian theories of the brain [22]. Dyna – “an integrated architecture for learning, planning and reacting” – proposed in [52], introduced the idea of not only utilising reward signals to learn good policies, but also



(a) Training reward curve for Assault

(b) Training reward curve for Seaquest

(c) Cumulative violations for Assault

(d) Cumulative violations for Seaquest

Figure 3: Training curves for DreamerV2 [30] and DreamerV2 with approximate shielding (ours).*

* The reward curves are smoothed with simple exponential smoothing with $w = 0.6$.

[1] The gives us roughly $\delta = 0.1$, using a tighter bound than Eq. 6 which bounds the probability of overestimating $\mu_{s\models\phi}$.

learning a dynamics model through observed transitions [52]. In theory, planning with the learned dynamics model could speed up convergence of the policy, but many early approaches suffered from model bias [9]. *Gaussian processes* (GPs) [57] were quickly used as the stand-in dynamics model for the Dyna architecture as they reduced model bias by quantifying their own uncertainty [18]. However, GPs struggle in high-dimensional settings and so the use neural architectures has been increasingly explored instead.

More recently, with the neural architecture Dreamer [28], Hafner et al. demonstrated that policies can be learnt purely from imagined experience and transfer well to the original environment. Additionally, DreamerV2 [30] and DreamerV3 [31] demonstrated state-of-the-art performance in a variety of domains including the Atari benchmark [11, 42] and MineRL [26] both of which have been notorious challenges for MBRL.

Once a world model is learned it can be used in a flexible manner for policy optimisation [28], online planning schemes [29, 33, 59], risk measures [62], and defining intrinsic rewards for improved exploration [38, 49]. As a result world model have been applied in a variety of domains, such as robotics [58], imitation learning [19], continual learning [36] and safe RL [8].

Safe RL is typically categorised as the problem of maximising reward, while maintaining some reasonable system performance during learning and deployment of the agent [24]. This definition has been interpreted in many different ways, stemming from different objectives in different domains. For example, reward hacking [6, 51] refers to an agent 'gaming' or exploiting a misspecified reward function, which can lead to undesired outcomes. Robustifying policies to distributional shift [7, 46, 56] and the alignment problem [20, 48] are also important areas of research in safe RL. However, we tackle the problem of *safe exploration* [24, 45] which can be described as the problem of minimising the violation of safety-constraints during the exploratory phase of training and beyond.

The constrained Markov decision process (CMDP) [4] is a widely used framework for modeling decision-making problems with safety constraints. In addition to maximising expected reward, agents must satisfy a set of constraints encoded as a cost function that penalises unsafe state-action pairs. In the tabular case, linear programs can be used to solve CMDPs [4]. In the non-tabular case, a variety of model-free algorithms with function approximation have been proposed [1, 15, 17, 39, 60].

Model-based approaches for safe RL utilise a variety of different techniques for dynamics modelling and policy optimisation. Berkenkamp et al. use GPs to quantify model uncertainty in a principled way to safely learn neural network policies. Other approaches use ensembles of neural networks (NNs) to quantify uncertainty and either deploy MPC [40, 54], perform policy optimisation within a certified region of the state space [41], or use constrained policy optimisation with Lagrangian relaxation [61] to learn safety-aware policies. Notable work by As et al. leverages Dreamer [30] and stochastic weight averaging Gaussian (SWAG) [43] to obtain a Bayesian predictive distribution over possible world models that explain the dynamics of the environment. As et al. also stress the importance of policy optimisation with safety critics over shortsighted MPC schemes.

Shielding for RL has been introduced as a correct by construction reactive (*shield*), which prevents the learned policy from entering unsafe states defined by some temporal logic formula [3]. The shield itself can be applied before the agent picks an action (preemptive), modifying the action space of the agent. Alternatively, the shield can be applied after the agent picks an action (post-posed), overriding actions proposed by the agent if they lead to a violation. Both types of shield require the ability to construct and solve a safety game [14] on a relatively compact representation of the MDP. Similar to control barrier functions (CBFs) [5] from optimal control theory [37], the shield projects the learned policy back into a verified safe set on the state space.

Recent work on shielding includes generalising it to partially observed [16] and multi-agent [21] settings, as well as resource constrained partially observed MDPs [2]. Many of these methods still require a suitable abstraction of the environment or sufficient domain knowledge for synthesising a shield. However, these strong assumptions come hand in hand with strong guarantees on safety of the learned policy, specifically [3] show that by construction their shield synthesis procedure guarantees safety with minimal interference.

Learning a shield online is an alternative approach to shielding RL policies without requiring significant prior knowledge. For example, Shperberg et al. propose tabular and parametric shields which are learning online to prevent agents from repeating catastrophic mistakes in the partially observed setting [50]. Other online shielding approaches include *latent shielding* [32] and BPS, both of which have substantially influenced our work. For a more complete review of reactive methods based on shielding we refer the interested reader to [44].

6 CONCLUSIONS

In this paper we presented an approximate shielding algorithm for safe exploration of Atari agents and more general RL policies. Building on DreamerV2 [30] and previous work, such as, *latent shielding* [32] and BPS [25], we propose a more general algorithm that uses safety critics and policy roll-outs to perform look-ahead shielding in the latent space of a learned world model.

In contrast to previous work, we are able to successfully apply our approximate shielding algorithm with minimal hyperparameter tuning and no shielding introduction schedules. While we loose the benefit of strict and formal guarantees obtained by earlier shielding approaches [3], we are able to derive some probabilistic guarantees, although this is incomplete and further work should be done to derive bounds for the approximate transition system.

Nevertheless, our empirical results are promising and provide some good evidence that general RL agents can benefit from shielding in certain settings, not only in terms of complying with safety specifications, but also in terms of improved performance. The aim of this research is to shed light on the promise of this approach and we hope this opens the door to further investigation.

ACKNOWLEDGMENTS

This work was supported by UK Research and Innovation [grant number EP/S023356/1], in the UKRI Centre for Doctoral Training in Safe and Trusted Artificial Intelligence (www.safeandtrustedai.org).

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] Michal Ajdarów, Šimon Brlej, and Petr Novotnỳ. 2022. Shielding in Resource-Constrained Goal POMDPs. *arXiv preprint arXiv:2211.15349* (2022).

[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] Aaron D Ames, Xiangru Xu, Jessy W Grizzle, and Paulo Tabuada. 2016. Control barrier function based quadratic programs for safety critical systems. *IEEE Trans. Automat. Control* 62, 8 (2016), 3861–3876.

[6] 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).

[7] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. 2020. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research* 39, 1 (2020), 3–20.

[8] Yarden As, Ilnura Usmanova, Sebastian Curi, and Andreas Krause. 2022. Constrained Policy Optimization via Bayesian World Models. In *International Conference on Learning Representations*. https://openreview.net/forum?id=PRZoSmCinhf

[9] Christopher G Atkeson and Juan Carlos Santamaria. 1997. A comparison of direct and model-based reinforcement learning. In *Proceedings of international conference on robotics and automation*, Vol. 4. IEEE, 3557–3564.

[10] Christel Baier and Joost-Pieter Katoen. 2008. *Principles of model checking*. MIT press.

[11] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. 2013. The Arcade Learning Environment: An Evaluation Platform for General Agents. *Journal of Artificial Intelligence Research* 47 (jun 2013), 253–279.

[12] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432* (2013).

[13] 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).

[14] Roderick Bloem, Bettina Könighofer, Robert Könighofer, and Chao Wang. 2015. Shield synthesis: Runtime enforcement for reactive systems. In *Tools and Algorithms for the Construction and Analysis of Systems: 21st International Conference, TACAS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015, Proceedings 21*. Springer, 533–548.

[15] Steven Bohez, Abbas Abdolmaleki, Michael Neunert, Jonas Buchli, Nicolas Heess, and Raia Hadsell. 2019. Value constrained model-free continuous control. *arXiv preprint arXiv:1902.04623* (2019).

[16] Steven Carr, Nils Jansen, Sebastian Junges, and Ufuk Topcu. 2022. Safe Reinforcement Learning via Shielding under Partial Observability. *arXiv preprint arXiv:2204.00755* (2022).

[17] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. 2017. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research* 18, 1 (2017), 6070–6120.

[18] 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.

[19] Branton DeMoss, Paul Duckworth, Nick Hawes, and Ingmar Posner. 2023. DITTO: Offline Imitation Learning with World Models. *arXiv preprint arXiv:2302.03086* (2023).

[20] Lauro Langosco Di Langosco, Jack Koch, Lee D Sharkey, Jacob Pfau, and David Krueger. 2022. Goal misgeneralization in deep reinforcement learning. In *International Conference on Machine Learning*. PMLR, 12004–12019.

[21] Ingy ElSayed-Aly, Suda Bharadwaj, Christopher Amato, Rüdiger Ehlers, Ufuk Topcu, and Lu Feng. 2021. Safe Multi-Agent Reinforcement Learning via Shielding. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*. 483–491.

[22] Karl Friston. 2003. Learning and inference in the brain. *Neural Networks* 16, 9 (2003), 1325–1352.

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

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

[25] Mirco Giacobbe, Mohammadhosein Hasanbeig, Daniel Kroening, and Hjalmar Wijk. 2021. Shielding Atari Games with Bounded Prescience. 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, 1507–1509. https://doi.org/10.5555/3463952.3464141

[26] William H Guss, Cayden Codel, Katja Hofmann, Brandon Houghton, Noboru Kuno, Stephanie Milani, Sharada Mohanty, Diego Perez Liebana, Ruslan Salakhutdinov, Nicholay Topin, et al. 2019. NeurIPS 2019 competition: the MineRL competition on sample efficient reinforcement learning using human priors. *arXiv preprint arXiv:1904.10079* (2019).

[27] David Ha and Jürgen Schmidhuber. 2018. Recurrent World Models Facilitate Policy Evolution. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf

[28] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. 2020. Dream to Control: Learning Behaviors by Latent Imagination. In *International Conference on Learning Representations*. https://openreview.net/forum?id=S1lOTC4tDS

[29] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. 2019. Learning latent dynamics for planning from pixels. In *International conference on machine learning*. PMLR, 2555–2565.

[30] Danijar Hafner, Timothy P Lillicrap, Mohammad Norouzi, and Jimmy Ba. 2021. Mastering Atari with Discrete World Models. In *International Conference on Learning Representations*. https://openreview.net/forum?id=0oabwyZbOu

[31] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. 2023. Mastering Diverse Domains through World Models. *arXiv preprint arXiv:2301.04104* (2023).

[32] Peter He, Borja G León, and Francesco Belardinelli. 2022. Do Androids Dream of Electric Fences? Safety-Aware Reinforcement Learning with Latent Shielding. In *Proceedings of the Workshop on Artificial Intelligence Safety 2022 (SafeAI 2022) (CEUR Workshop Proceedings, 3087)*, Gabriel Pedroza, José Hernández-Orallo, Xin Cynthia Chen, Xiaowei Huang, Huáscar Espinoza, Mauricio Castillo-Effen, John McDermid, Richard Mallah, and Seán Ó hÉigeartaigh (Eds.). Aachen. https://ceur-ws.org/Vol-3087/paper_50.pdf

[33] Chia-Man Hung, Shaohong Zhong, Walter Goodwin, Oiwi Parker Jones, Martin Engelcke, Ioannis Havoutis, and Ingmar Posner. 2022. Reaching through latent space: From joint statistics to path planning in manipulation. *IEEE Robotics and Automation Letters* 7, 2 (2022), 5334–5341.

[34] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. 2019. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems* 32 (2019).

[35] Nils Jansen, Bettina Könighofer, Sebastian Junges, and Roderick Bloem. 2018. *Shielded Decision-Making in MDPs*. WorkingPaper. Cornell University Library.

[36] Samuel Kessler, Piotr Miłoś, Jack Parker-Holder, and Stephen J. Roberts. 2022. The Surprising Effectiveness of Latent World Models for Continual Reinforcement Learning. In *Deep Reinforcement Learning Workshop NeurIPS 2022*. https://openreview.net/forum?id=-lHOOgHuWwu

[37] Donald E Kirk. 2004. *Optimal control theory: an introduction*. Courier Corporation.

[38] Artem Latyshev and Aleksandr I Panov. 2023. Intrinsic Motivation in Model-based Reinforcement Learning: A Brief Review. *arXiv preprint arXiv:2301.10067* (2023).

[39] Yongshuai Liu, Jiaxin Ding, and Xin Liu. 2020. Ipo: Interior-point policy optimization under constraints. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 4940–4947.

[40] Zuxin Liu, Hongyi Zhou, Baiming Chen, Sicheng Zhong, Martial Hebert, and Ding Zhao. 2020. Constrained model-based reinforcement learning with robust cross-entropy method. *arXiv preprint arXiv:2010.07968* (2020).

[41] Yuping Luo and Tengyu Ma. 2021. Learning barrier certificates: Towards safe reinforcement learning with zero training-time violations. *Advances in Neural Information Processing Systems* 34 (2021), 25621–25632.

[42] Marlos C. Machado, Marc G. Bellemare, Erik Talvitie, Joel Veness, Matthew J. Hausknecht, and Michael Bowling. 2018. Revisiting the Arcade Learning Environment: Evaluation Protocols and Open Problems for General Agents. *Journal of Artificial Intelligence Research* 61 (2018), 523–562.

[43] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. 2019. A simple baseline for bayesian uncertainty in deep learning. *Advances in neural information processing systems* 32 (2019).

[44] Haritz Odriozola-Olalde, Maider Zamalloa, and Nestor Arana-Arexolaleiba. 2023. Shielded Reinforcement Learning: A review of reactive methods for safe learning. In *2023 IEEE/SICE International Symposium on System Integration (SII)*. IEEE, 1–8.

[45] Martin Pecka and Tomas Svoboda. 2014. Safe exploration techniques for reinforcement learning–an overview. In *Modelling and Simulation for Autonomous Systems: First International Workshop, MESAS 2014, Rome, Italy, May 5-6, 2014, Revised Selected Papers 1*. Springer, 357–375.

[46] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. 2018. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 3803–3810.

[47] Martin L Puterman. 1990. Markov decision processes. *Handbooks in operations research and management science* 2 (1990), 331–434.

[48] Stuart Russell and Peter Norvig. 2021. (4 ed.). Pearson Education Limited, 49–52.

[49] Ramanan Sekar, Oleh Rybkin, Kostas Daniilidis, Pieter Abbeel, Danijar Hafner, and Deepak Pathak. 2020. Planning to explore via self-supervised world models. In *International Conference on Machine Learning*. PMLR, 8583–8592.

[50] Shahaf S Shperberg, Bo Liu, and Peter Stone. 2022. Learning a Shield from Catastrophic Action Effects: Never Repeat the Same Mistake. *arXiv preprint arXiv:2202.09516* (2022).

[51] Joar Max Viktor Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. 2022. Defining and Characterizing Reward Gaming. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). https://openreview.net/forum?id=yb3HOXO3lX2

[52] Richard S Sutton. 1991. Dyna, an integrated architecture for learning, planning, and reacting. *ACM Sigart Bulletin* 2, 4 (1991), 160–163.

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

[54] Garrett Thomas, Yuping Luo, and Tengyu Ma. 2021. Safe reinforcement learning by imagining the near future. *Advances in Neural Information Processing Systems* 34 (2021), 13859–13869.

[55] Edward C Tolman. 1948. Cognitive maps in rats and men. *Psychological review* 55, 4 (1948), 189.

[56] Núria Armengol Urpí, Sebastian Curi, and Andreas Krause. 2021. Risk-Averse Offline Reinforcement Learning. In *International Conference on Learning Representations*. https://openreview.net/forum?id=TBIzh9b5eaz

[57] Christopher KI Williams and Carl Edward Rasmussen. 2006. *Gaussian processes for machine learning*. Vol. 2. MIT press Cambridge, MA.

[58] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. 2022. DayDreamer: World Models for Physical Robot Learning. In *6th Annual Conference on Robot Learning*. https://openreview.net/forum?id=3RBY8fKjHeu

[59] Zifan Wu, Chao Yu, Chen Chen, Jianye HAO, and Hankz Hankui Zhuo. 2022. Plan To Predict: Learning an Uncertainty-Foreseeing Model For Model-Based Reinforcement Learning. In *Advances in Neural Information Processing Systems*, Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (Eds.). https://openreview.net/forum?id=L9YayWPcHA_

[60] Tsung-Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge. [n.d.]. Projection-Based Constrained Policy Optimization. In *International Conference on Learning Representations*.

[61] Moritz A Zanger, Karam Daaboul, and J Marius Zöllner. 2021. Safe continuous control with constrained model-based policy optimization. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 3512–3519.

[62] Jesse Zhang, Brian Cheung, Chelsea Finn, Sergey Levine, and Dinesh Jayaraman. 2020. Cautious adaptation for reinforcement learning in safety-critical settings. In *International Conference on Machine Learning*. PMLR, 11055–11065.