2024 ACM/IEEE 15th International Conference on Cyber-Physical Systems (ICCPS) | 979-8-3503-6927-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICCPS61052.2024.00014

Vulnerability Analysis for Safe Reinforcement Learning in Cyber-Physical Systems

Shixiong Jiang*
University of Notre Dame
sjiang5@nd.edu

Mengyu Liu*
University of Notre Dame
mliu9@nd.edu

Fanxin Kong
University of Notre Dame
fkong@nd.edu

***Abstract*—Safe reinforcement learning (RL) has been recently employed to train a control policy that maximizes the task reward while satisfying safety constraints in a simulated secure cyber-physical environment. However, the vulnerability of safe RL has been barely studied in an adversarial setting. We argue that understanding the safety vulnerability of learned control policies is essential to achieve true safety in the physical world. To fill this research gap, we first formally define the adversarial safe RL problem and show that the optimal policies are vulnerable under observation perturbations. Then, we propose novel safety violation attacks that induce unsafe behaviors by adversarial models trained using reversed safety constraints. Finally, both theoretically and experimentally, we show that our method is more effective in violating safety than existing adversarial RL works which just seek to decrease the task reward, instead of violating safety constraints.**

I. Introduction

Cyber-physical systems (CPS) employ computing and networking components to interact with the physical world via sensors and actuators. Recently, CPS has been starting to integrate more intelligence that enables promising applications such as autonomous vehicles, drones, and other robotic systems [1]. Meanwhile, the increased autonomy comes with new security and safety issues for CPS [2]–[6].

The great success of deep reinforcement learning (RL) in recent years has motivated many research efforts that adopt it to synthesize control policies (i.e. learning-enabled controllers) for CPS. However, it is challenging to ensure safety when deploying them to the real-world CPS. Safe RL thus is drawing much attention, of which the goal is to maximize the task reward while satisfying safety constraints. There are two major research threads of safe RL. One thread handles the problem by solving a constrained optimization, where they rely on the knowledge of a mathematical model that characterizes the system dynamics [7]–[10]. The other thread needs no such knowledge and instead, is guided by a set of formal specifications using linear temporal logic (LTL) [11] or signal temporal logic (STL) [12].

Both research threads above take advantage of the power of neural networks. However, neural networks have been shown to be vulnerable to adversarial attacks, i.e. a small perturbation of the input may cause the output to vary drastically [13]. This may cause safety violations when deploying a neural network RL control policy to CPS. On the one hand, existing safe RL methods work well to respect safety constraints in simulated secure environments, but their vulnerability has been barely studied under adversarial perturbations. We consider an adversarial setting where the observation perturbations come from the physical world such as sensing noises and sensor attacks [14]. We believe that studying the vulnerability of safe RL in the adversarial setting will be essential to achieving true safety in the physical world.

On the other hand, existing adversarial RL works are not suitable to address the vulnerability of safe RL. Their robustness concept and training methods follow standard RL settings, where attacks of observation perturbations aim to decrease their rewards as much as possible [15]–[17]. However, safe RL has an additional dimension that captures the cost of safety constraint violations. We argue that the cost should be more critical than the task reward in the safe RL setting because the constraint violations can cause catastrophic consequences in real-world CPS. Consider an example where the navigation task of an autonomous vehicle has the reward as ‘to reach a target as soon as possible’ and the safety constraint as ‘to avoid obstacles’ [18]. Existing adversarial RL methods, which reduce the reward, may cause the vehicle to arrive at the target late or steer away from the target. However, they do not necessarily make the vehicles violate the safety constraint, e.g. to crash into obstacles, which is more critical after all.

Given the research gap, we thus investigate the vulnerability of safe RL with adversarial observation perturbations. This paper focuses on the formal specification guided safe RL and its safety specification (i.e. formally specified safety constraints) violations. Unlike traditional RL, which relies on hand-engineered reward functions, formal specification guided RL automatically transfers task and constraint specifications to reward and cost functions for policy training. This has been proved to be effective by recent works such as [19]–[22]. We aim to address two key questions i) How vulnerable will a learned control policy be under adversarial observation perturbations? ii) how to design effective and stealthy attacks to violate safety specifications? To answer these questions, we first formally define the adversarial formal specification guided RL problem and describe how to analyze the safety vulnerability of a learned control policy. Then, we propose diverse safety violation attacks that can drift a system to the unsafe region. We also discuss possible mitigation methods to address the vulnerability in the end. Our major contributions are summarized below.

- Targeting signal temporal logic, we formally analyze the vulnerability of control policies in STL-guided safe RL and show that the optimal policies of safe RL are

* denotes equal contribution.

979-8-3503-6927-4/24/$31.00 ©2024 IEEE
DOI 10.1109/ICCPS61052.2024.00014

vulnerable to adversarial observation attacks.

- We propose multiple safety violation attacks that apply to adversaries with different levels of knowledge about the system. Our method novelly reverses the STL specifications to train adversarial models that provide attackers with observation perturbations to induce unsafe behaviors. We also present a formal analysis to show that existing adversarial RL works of minimizing the task reward do not always work on violating safety.
- We conduct extensive experiments using multiple benchmarks including the OpenAI Safety Gym. The evaluation results show that our method is much more effective in violation safety than existing adversarial RL works while staying stealthy.

The rest of the paper is organized as follows. Section II discusses related work. Section III introduces preliminaries. Section IV defines the problem and proposes safety violation attacks with theoretical analysis. Section V evaluates the proposed method. Section VI discusses the limitations of our work and mitigation. Section VII concludes the paper.

II. Related Work

Safe RL focuses on developing RL algorithms that incorporate safety constraints during both the learning and testing phases. The main objective of Safe RL is to ensure that the agent's learning and decision-making processes do not lead to unsafe or undesirable outcomes. In this section, we discuss literature related to safe RL, especially STL-guided safe RL. Furthermore, we also introduce existing works about how to design adversary attacks to break RL safety.

Temporal logic guided safe RL. Temporal logic provides a precise and unambiguous expression of the system's intended behaviors. Aside from the liveness(something good happens eventually) properties, the safety(something bad never happens) constraints can be formed into explicit specifications and must be strictly adhered to [23]. Donze et al. propose quantitative semantics to map the degree of the robustness of an STL specification to a real value [24]. This mapping enables STL-guided safe RL without the need to manually craft the reward function. Existing works focus on using STL-guided RL to complete control tasks such as reach&avoid and achieve liveness and safety at the same time [18], [21], [25].

Temporal logic offers the capability to tailor safety constraints according to scenarios and settings. Liu et al. and Li et al. form the safety as a specification of not entering a ball-shaped unsafe set during the navigation to the target set [18], [25]. Singh et al. define safety as a specification of not entering an unsafe set which is a conjunction of half-spaces [21]. Researchers in [19] define safety with formal specifications and human demonstrations jointly.

Adversary attack on RL. Adversary attack refers to situations where an external agent (the adversary) intentionally manipulates the environment or the input data to mislead the RL agent. Some existing works focus on attacking the observation space [15], [16], [26]. Researchers of [15] apply the Fast Gradient Decent method (FGSM) to generate adversarial observations to mislead the agent. Researchers of [27] apply a universal perturbation on observation at every step. Zhang et al. proposed an adversary attack on the observation that causes maximum action difference [16].

Some other works focus on designing attacks to affect the rewarding process which provides feedback to the learning agent in the form of rewards [17], [28]. Pattanaik et al. propose a method that integrates the information from the value function and the information from the loss function to degrade the agent's performance [17]. Researchers of [28] propose TrojDRL which generates backdoor attacks for DRL by taking advantage of hacking the rewards.

It is also important to make the attack stealthy to detectors to make it adversarial [29], [30]. Liu et al. design a framework to attack safe RL by maximizing the cost to enlarge the effect and maximizing the rewards to keep stealthy [29]. Researchers of [30] propose two attacks utilizing the control and observation information with predictive models to keep the attack stealthy.

As safety concerns have become increasingly apparent, [29] is the first study to address attacking safe RL during the training phase, compromising the obtained control policy. Our research diverges from this work as we concentrate on attacking a well-trained safe RL policy and demonstrating its vulnerability to observation attacks.

III. Preliminaries

This section provides a brief introduction to the preliminary concepts utilized in the paper. We start with the signal temporal logic, then define the safe reinforcement learning problem model by constraint Markov Decision Process and define formal specification guided RL. Finally, we discuss the threat model used in the paper.

A. Signal temporal logic

STL serves as a logical framework for expressing temporal properties concerning signals with real-valued data. STL formulas are defined using Boolean formulas that combine sub-formulas recursively or through the application of temporal operators to sub-formulas [24]. In this paper, we consider the system behavior specified by the STL with the following fragment:

$$\Phi ::= \mu|\neg\phi|\phi \wedge \phi|\mathbf{G}\phi|\mathbf{F}\phi \mid \phi_1 \mathbf{U} \phi_2 \tag{1}$$

where the $\wedge$ and $\neg$ are logic conjunction and negation. G, F, and U are the always, finally, and until operators respectively. Operators U, G and F can be transformed from each other, for example $\mathbf{G}\phi = \neg(\mathbf{F}\neg\phi)$ [31].

The STL uses quantitative semantics to compute the robustness value that maps the signal to a real value. The quantitative semantics functioned by ρ transform the boolean specification of the STL into a real value that measures how much satisfaction the system meets the STL formula. A positive value of the function ρ at time t the system observation s_t indicates satisfaction with the specification, whereas a negative value implies a violation of the system's specification. We show the function ρ in terms of a robustness value as below refer to [18], [24]:

$$\begin{aligned}
&\rho(\bar{s}_t, (f(\bar{s}) < d)) = d - f(s_t) \\
&\rho(\bar{s}_t, \neg\phi) = -\rho(\bar{s}, \phi, t) \\
&\rho(\bar{s}_t, \phi_1 \wedge \phi_2) = \min(\rho(\bar{s}_t, \phi_1), \rho(\bar{s}_t, \phi_2)) \\
&\rho(\bar{s}_t, \phi_1 \vee \phi_2) = \max(\rho(\bar{s}_t, \phi_1), \rho(\bar{s}_t, \phi_2)) \\
&\rho\left(\bar{s}_t, F_{[t_0, t_0+T]}\phi\right) = \max_{t \in [t_0, t_0+T]} \rho(\bar{s}_t, \phi) \\
&\rho\left(\bar{s}_t, G_{[t_0, t_0+T]}\phi\right) = \min_{t \in [t_0, t_0+T]} \rho(\bar{s}_t, \phi) \\
&\rho\left(\bar{s}_t, \phi_1 U_{[t_0, t_0+T]}\phi_2\right) \\
&= \max_{t \in [t_0, t_0+T]} \left(\min\left(\rho(\bar{s}_t, \phi_2), \min_{t'' \in [t, t')} \rho(\bar{s}_{t''}, \phi_1)\right)\right)
\end{aligned} \tag{2}$$

Where t_0 and T are the time that the task starts and the duration of the task respectively, $\bar{s}$ is a trajectory containing continuous system states in discrete time.

B. Safe reinforcement learning

Definition III.1. A Finite Horizon Constraint Markov Decision Process (CMDP) is defined as a tuple $M := (S, U, p, r, c, \gamma)$, where $S \subseteq \mathbb{R}^n$ is the state space, $U \subseteq \mathbb{R}^m$ is the action space. $p : S \times U \times S \to [0, 1]$ is the transition function that represents the probability $p(s_{t+1} \mid s_t, u_t)$ from state s_t to s_{t+1} by taking action u_t. $r : S \times U \times S \to \mathbb{R}$ is the reward function. $S \times U \times S \to [0, C_m]$ is the cost function that measures the cost once the violating the constraint, where C_m is the maximum cost. $\gamma \in [0, 1]$ is the discounting parameter.

We suppose that a control problem for a CPS a is a process of finding an optimal policy $\pi^\star : S \to A$ that maximizes the expected cumulative reward and minimizes the total cost:

$$\pi^\star = \arg\max_\pi \mathbb{E}^\pi \sum_{t=0}^{T-1} \gamma^t r(s_t, a_t, s_{t+1}) \tag{3}$$

$$\pi^\star = \arg\min_\pi \mathbb{E}^\pi \sum_{t=0}^{T-1} \gamma^t c(s_t, a_t, s_{t+1}) \tag{4}$$

We use $horizon(M) = T$ as the time horizon that represents the max execution time steps for the CMDP. $\mathbb{E}^\pi$ is the expected reward(cost) returned by π.

In this CPS context, we consider the real state s_t as challenging to directly access. Instead, the system's state is estimated on sensor observations, which inherently come with bounded noise. For simplicity, we assume negligible noise in this paper and henceforth use "observation" interchangeably with s_t.

C. Formal Specification Guided RL

Using formal specification for safe exploration to guide the RL has been explored. The existing work uses the robustness value of the quantitative semantics as the reward function. So the RL problem is to find a policy that maximizes the robustness value or increases the probability of satisfying the STL specification. This approach largely reduces the difficulty of designing specific reward functions in complex tasks or environments.

In this paper, we focus on safety-critical CPSs characterized by a pre-defined task objective and multiple safety constraints. To illustrate, in the case of an autonomous vehicle, the task objective might be reaching a specific destination eventually, while the constraints would involve avoiding obstacles. Similarly, in a robot arm control scenario, the controller's objective is to control the arm to grab a box while ensuring it doesn't collide with any other objects. We consider using STL to specify the goal and safety constraint. These requirements can be formally expressed as :

Definition III.2 (Goal)**.** The goal is the set ϕ_g of STL specifications which specify the system's control objective. Given the start time t_0 and time horizon $horizon(\phi_g) = T$, the system achieves its goal only if the $\rho(\bar{s}_t, F_{[t_0,t_0+T]}\phi_g) > 0$

Definition III.3 (Safety constraint)**.** The goal is the set ϕ_c of STL specifications which specify the system's safety constraint. Given the start time t_0 and time horizon $horizon(\phi_g) = T$, the system satisfies the safety constraint as long as $\rho(\bar{s}_t, G_{[t_0,t_0+T]}\phi_c) > 0$

According to the above definition, the STL specification of such a task with goal and safety constraints can be expressed as the following:

$$\Phi = F_{[t_0,t_0+T]}\phi_g \wedge G_{[t_0,t_0+T]}\phi_c \tag{5}$$

Based on the Equation 5, the system is required to satisfy ϕ_g before time $t_0 + T$ and also satisfy the safety constraints specified by ϕ_c during the time horizon $horizon(\Phi) = T$. Then We define an STL-guided safe-RL task which aims to find the optimal policy $\pi^\star$ that maximizes the robustness degree of the STL specification. The STL specification of the safe-RL agent is presented as:

Definition III.4. (STL-guided RL) Given an STL specification $\Phi = F_{[t_0,t_0+T]}\phi_g G_{[t_0,t_0+T]}\phi_c$ with a horizon $horizon(\Phi) = T$, a CMDP $M := (S, A, p, r, c, \gamma)$ with unknown p and an initial state trajectory $s_{0:T}$, the STL-guided RL problem is to find a policy $\pi^\star$ that maximize the expected cumulative robustness value of the specified STL specification Φ:

$$\pi^\star = \arg\max_\pi E^\pi \sum_{t=0}^{T} \gamma^t \rho(\bar{s}_t, \Phi) \tag{6}$$

D. Threat model

In this paper, we consider various scenarios with different levels of known system knowledge of the adversary. Specifically, the adversary can access the system's transition function p and the control policy π. If the adversary possesses knowledge of the p, the adversary can approximate the subsequent s_{t+1} of the system given the action and current observation s_t. If π is accessible to the adversary, they can derive the action u_t based on the observation s_t.

Attacker's knowledge. Regarding the adversary knowledge, we consider three scenarios: (1) White-box attack: The attacker has full access to both the system's control policy and transition function. (2) Grey-box attack: The attacker knows either the system's control policy or transition function. (3) Black-box attack: The attacker has no access to either.

Attacker's capability. We assume that the adversary knows the STL specification Φ used by the system when training the control policy. The adversary can also access all the sensors of the system and can modify all the sensor values.

IV. Safety Violation Attack

In this section, we introduce our framework for adversary attacks on the STL-guided RL-based control policy. We also provide the theoretical analysis to prove that our framework is effective.

A. Problem Formulation

We assume that there is an adversary that maliciously changes the observation value of the system observation s_t to $s'_t = h(s_t)$ where h is the adversary policy. We define the effectiveness and stealthiness of the adversary problem to better understand the property of the safety violation attack.

Definition IV.1 (Attack Effectiveness)**.** Given the safety constraint ϕ_c, start time t_0 and time horizon $horizon(\phi_c) = T$, denote the $\bar{s}'$ as the trajectory of perturbed observation s'_t from t_0 to T. the attack is effective if $\rho(\bar{s}', G_{[t_0,t_0+T]}\phi_c) < 0$.

The effectiveness describes that the adversary's objective is to force the system to violate the safety constraint of STL specification. Then we introduce another metric to measure the attacker's stealthiness.

Definition IV.2 (Attack Stealthiness)**.** Denote the $\bar{s}'$ as the trajectory of perturbed observation s'_t from t_0 to T. The perturbation range is limited within a ℓ_α-ball around the initial observation where $\beta^\epsilon_\alpha(s_t) := \|s'_t - s_t\|_\alpha \leq \epsilon$ and ϵ is the size of the perturbation range. Given the manipulated observation trajectory $\bar{s}'$ and a perturbation range $\beta^\epsilon_\alpha(s)$, the attack is stealthy if $\rho(\bar{s}'_t, \phi_g) > \rho(\bar{s}_t, \phi_g)$.

The concept of stealthiness, as defined in previous studies, takes on various perspectives. For example, [32] characterizes it as the range of perturbations around the original observation. On the other hand, for those works focus on the system safety [33] [34] [35], stealthiness is assessed in systems equipped with a detector, which implies avoiding detection. The work by [29] introduces an additional level of stealthiness called reward stealthiness. They consider the reward stealthiness as 'the agent might easily detect a dramatic reward drop', which inspires us that, in the CPS domain, if there is a huge drop in the robustness of ϕ_g, the system may notice the anomaly behavior and detect there is an adversary.

We add a new dimension of stealthiness within the context of STL-guided safe RL. The Definition IV.2 considers an attack as more stealthy if it can maintain the robustness value of ϕ_g from Equation 2 after the attack. Therefore, it cannot be detected by monitoring the robustness score of ϕ_g. Additionally, we introduce the perturbation set $\beta^\epsilon_p(s)$ to confine s' within specified bounds, thus delineating that the perturbation in observation adheres to established standards of stealthiness, as prior literature [15] [29]. In general, the problem is that the adversary wants to find the observation perturbation s'_t to force the system to take a malicious action u' which minimizes the robustness of the safety specification $\rho(\bar{s}_{t+1}, \phi_c)$ bounded by the stealthiness.

$$\begin{aligned} s'_t = \underset{s'_t}{\operatorname{argmin}}\, & \rho(\bar{s}_{t+1}, \phi_c) \\ \text{s.t. } \| s'_t - s_t \|_\alpha & \leq \epsilon \\ \rho(\bar{s}'_t, \phi_g) & > \rho(\bar{s}_t, \phi_g) \\ u'_t & = \pi(s'_t) \\ s_{t+1} & = p(s_t, u'_t) \end{aligned} \tag{7}$$

While adversary attacks directed at RL-based control have been extensively researched, our specific problem remains distinct and relatively unexplored. Previous studies have primarily concentrated on manipulating system observations to reduce the overall rewards, primarily impacting agent performance. These approaches often do not account for the crucial safety constraints of the system. We denote these methods as reward(value) decreasing (RD) methods and we show these methods can't achieve attack effectiveness.

Theorem IV.1. Suppose there is a RD adversary policy h_{rd} method manipulates the observation as $s'_t = s_t + h_{rd}(s_t)$. The adversary policy h_{rd} cannot guarantee to achieve attack effectiveness.

We provide proof for the Theorem IV.1 in subsection IV-C. To address the problem, we propose the **Safety Violation Attack (SVA)** framework where the adversary deliberately forces the system to violate the safety constraint under the limitation of stealthiness.

B. Safety Violation Attack Framework

White-box attack. We begin with the white-box attack. Since the adversary knows the transition function p and control policy π, the process can be formalized to an optimization problem as below:

However, directly solving the optimization function in Equation 7 is hard since an NN-based control policy is typically nonlinear and nonconvex [36]. We construct an alternate way of solving the s'_t. We divided Equation 7 into two parts. First, The attacker initiates the process by obtaining a malicious action u'_t which is designed to compromise the robustness of the safety constraint in the STL specification:

$$\begin{aligned} u'_t &= \underset{u'_t}{\operatorname{argmin}}\, \rho(\bar{s}_{t+1}, \phi_c) \\ s_{t+1} &= p(s_t, u'_t) \end{aligned} \tag{8}$$

The u'_t serves as a targeted action to guide the subsequent observation perturbation. The next step involves executing the observation perturbation to induce the system to perform action u'_t. Then the observation perturbation s' can be generated using solvers like FGSM [37] and PGD [32] by minimizing $\ell(u_t, u'_t)$ where ℓ is a distance function that measures the distance between current action u_t with the adversary desired u'_t. We present our SVA framework under the white-box setting in Algorithm 1.

Algorithm 1: SVA(White-box version)

1 **Input:** The observation s_t, control policy π, STL specification ϕ_g and ϕ_c, distance function ℓ, update budget n, step size η
2 **Output:** Observation perturbation s'_t
3 $u'_t \leftarrow \mathrm{argmin}\, \rho(\bar{s}_{t+1}, \phi_c)$
4 $\Gamma(s_t) \leftarrow \{s'_t | \rho(\bar{s}'_t, \phi_g) > \rho(\bar{s}_t, \phi_g)\}$
5 $B(s_t) \leftarrow \beta^{\epsilon}_{\alpha}(s_t) \cap \Gamma(s_t)$
6 **for** $i = 0 : n$ **do**
7 $\quad u_t = \pi(s'_t)$
8 $\quad grad = \nabla_{s'_t} \ell\left(u_t, u'_t\right)$
9 $\quad s'_t = s'_t - \eta * \epsilon * sign(grad)$
10 $\quad s'_t \leftarrow \mathrm{Proj}_{B(s_t)}\left[s'_t\right]$
11 **end**
12 **return** s'_t

Line 3 in Algorithm 1 calculates the optimal action u'_t that maliciously forces the system to violate ϕ_c. Line 4 computes the $\Gamma(s_t)$ which is the set of s' that is constrained by stealthiness. Line 5 gets the final admissible set $B(s_t)$ which is the intersection of $\Gamma(s_t)$ and the set of the perturbation range $\beta^{\epsilon}_{\alpha}(s_t)$. Note that the set $\Gamma(s_t)$ is available because we have assumed the adversary knows the predefined STL specification. Line 7 computes the current malicious action u'_t and line 8 obtains the gradient of the 2-norm pairwise distance between u'_t and u'_t to the s'_t. Line 9 iteratively updates the s'_t and line 10 projects the s'_t within the admissible set $B(s_t)$. Finally, the algorithm returns the observation perturbation s'_t at time t.

Grey-box attack and Black-box attack. We consider the situation when the adversary has no knowledge of one of these two or has no knowledge of both defined in section III.

The grey-box attacks refer to the scenario in section III where the transition function or the control policy is unknown, the black-box attack scenarios assume both the transition function and the control policy are unknown.

In cases where the adversary lacks knowledge of the control policy, but has access to the transition function, a common solution is to train a surrogate control policy π' to substitute the π. This approach has been widely known [15] [38] as the transferability for an adversarial attack on supervised learning neural networks and RL policy. Since we assume the adversary can access the environment and the STL specification, the adversary can train such a surrogate control policy π' without knowing the origin control policy's algorithm and parameters.

In cases where the transition function p is not available to the adversary, it is infeasible to obtain a malicious action u' by solving equation 8. Existing studies like [26] [39] form this to a MDP problem and apply RL to train the adversary model to obtain an adversarial control input $u' = \pi_{adv}(s_t)$. The adversary's objective is to reduce the reward earned by the system. Therefore, an adversary policy is trained using the reward function $\hat{r}_t = -r_t$ [39] where r_t is the reward function of the victim policy. Instead of reducing the reward to degrade the control performance, we propose an alternative approach that leverages the control policy and reverses safety constraints.

Definition IV.3 (Safety Violated Adversary model)**.** Given a CMDP $M := (S, A, p, r, c)$ and an STL-guided RL policy π with STL specification $\phi = F_{[t_0,t_0+T]}\phi_g \wedge G_{[t_0,t_0+T]}\phi_c$, a safety violated adversary policy π_{adv} can be trained using the reversed version of the safety specification $F_{[t_0,t_0+T]}\phi_{adv}$ where $\phi_{adv} = \neg\phi_c$. The adversary policy can always obtain the malicious action $u'_t = \pi_{adv}(s_t)$ that forces the system to violate the safety constraint.

The Definition IV.3 gives a solution to obtain the malicious action when the adversary does not access the transition function. The malicious action u'_t is used to compute the s'_t as the Algorithm 1 lines 4-8 does. We provide the details of the algorithm in Algorithm 2.

When the attacker lacks both the knowledge of the transition function and control policy, we refer to it as a black-box scenario. In this case, the adversary can employ both the aforementioned methods (surrogate control policy and adversary model) to implement the SVA framework.

Algorithm 2: SVA(Black-box version)

1 **Input:** The current system state s_t, STL specification ϕ_g and ϕ_c, surrogate control policy π', adversary model π_{adv}, distance function ℓ, update budget n, step size η
2 **Output:** Observation perturbation s'_t
3 $u'_t \leftarrow \pi_{adv}(s_t)$
4 $\Gamma(s_t) \leftarrow \{s'_t | \rho(\bar{s}'_t, \phi_g) > \rho(\bar{s}_t, \phi_g)\}$
5 $B(s_t) \leftarrow \beta^{\epsilon}_{\alpha}(s_t) \cap \Gamma(s_t)$
6 **for** $i = 0 : n$ **do**
7 $\quad u_t = \pi'(s'_t)$
8 $\quad grad = \nabla_{s'_t} \ell\left(u_t, u'_t\right)$
9 $\quad s'_t = s'_t - \eta * \epsilon * sign(grad)$
10 $\quad s'_t \leftarrow \mathrm{Proj}_{B(s_t)}\left[s'_t\right]$
11 **end**
12 **return** s'_t

C. Theoretical Analysis

We conduct a comparative analysis between the proposed Safety Violation Attack (SVA) and the reward decreasing (RD) attack introduced by existing works [40] [39] [16]. We demonstrate an actor-critic RL algorithm (DDPG, SAC, and A2C) to show that reward-decreasing(RD) attack can't violate the safety specification effectively. The underlying intuition of the RD attack is to induce a sub-optimal action characterized by a lower observation-action value function $Q(s, a)$ output by the critic network. Next, we explain why SVA achieves better attack performance than RD attack.

We denote h_{rd} as the reward-decrease adversary algorithm. The adversary h_{rd} is a group of methods such as Researchers in [40] leverage the gradients of the critic network to guide the observation adversary towards minimizing the value function $Q(s,a)$. Researchers in [16] learn an NN-based model $Q_\pi(s,a)$ to reflect the action value.

We start with the Bellman Equations and a basic victim policy π where the policy is trained with the STL specification from Equation 5. Despite different RD attack algorithms, the Bellman equations under such adversary can be obtained as follows:

Definition IV.4. (Bellman equations with adversary). Given a victim control policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ and adversary algorithm $h_{rd} \in h : \mathcal{S} \to \mathcal{S}'$, we get

$$\tilde{V}_{\pi\circ h}(s) = \mathbb{E}_{\pi\circ h} \sum_{t=0}^{T} \gamma^t \rho(\bar{s}_t, \Phi)$$

The Definition IV.4 gives the value function $\tilde{V}$ when there is an adversary h who manipulates the observation s to s'. Based on the definition of the STL specification Φ from the Equation 5, we have:

$$\begin{aligned}
&\tilde{V}_{\pi\circ h}(s) \\
&= \mathbb{E}_{\pi\circ h} \sum_{t=0}^{T} \gamma^t \min\left(\rho(\bar{s}_t, F_{[0,t]}\phi_g), \rho(\bar{s}_t, G_{[0,t]})\phi_c\right) \\
&= \mathbb{E}_{\pi\circ h} \sum_{t=0}^{T} \gamma^t \min(\underbrace{\max_{t'\in[0,t]} (\rho(\bar{s}'_t, \phi_g))}_{\text{denote this term } A_t}, \underbrace{\min_{t'\in[0,t]} (\rho(\bar{s}'_t, \phi_c))}_{\text{denote this term } B_t})
\end{aligned}$$

We consider the initial state of the agent to be safe but not satisfy the goal yet, i.e. it satisfies the ϕ_c but not ϕ_g, then the original robustness values $A_t < 0$ and $B_t > 0$ without attacks. We give an example of how the RD adversary changes the value function. For a specific time step t_1, the value function under the adversary attack is the following:

$$\tilde{V}_{\pi\circ h}(s_{t_1}) \quad = \mathbb{E}_{\pi\circ h} \sum_{t=0}^{t_1-1} \gamma^t A_t + \gamma^{t_1} \rho(\bar{s}_{t_1}, \Phi)$$

Note that Equation 9 shows the value function when there is no attack at time step t_1. For the time step t_1, the RD adversary finds the s'_t to minimize the value function that:

$$h_{rd}(s_{t_1}) = \operatorname*{argmin}_{s'_{t_1}} \rho(\bar{s}_{t_1}, \Phi)$$
$$h_{rd}(s_{t_1}) = \operatorname*{argmin}_{s'_{t_1}} \min(A_{t_1}, B_{t_1})$$

As claimed before, $A_{t_1} < 0$ and $B_{t_1} > 0$, we have:

$$h_{rd}(s_{t_1}) = \operatorname*{argmin}_{s'_{t_1}} A_{t_1}$$

Note that the A_{t_1} is related to the ϕ_g instead of ϕ_c, which shows that the RD attack only decreases the robustness of ϕ_g but can't guarantee to minimize the robustness of ϕ_c. In other words, the RD adversary influences the goal completion not violating the safety constraint.

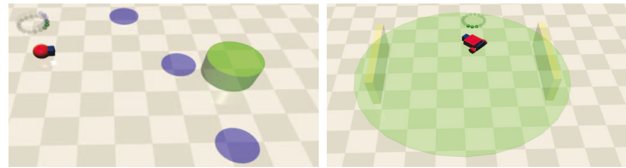


Figure 1: The PointGoal (left) and CarCircle (right) benchmarks.

Instead, our SVA denoted as h_{sva} finds s'_{t_1} to reduce the robustness of the ϕ_c:

$$h_{sva}(s_t) = \operatorname*{argmin}_{s'_t} \rho(\bar{s}_t, \phi_c)$$

Although there is no guarantee that SVA can violate the safety constraint for every trajectory $\bar{s}$, in section V, evaluation results show that SVA is more efficient than the existing methods.

V. Experiments

This section outlines our experimental methodology for evaluating the SVA framework across various benchmarks. All experiments were conducted on a machine equipped with an Intel Core i7-13700F processor running at 2.10 GHz with 16 cores and 16 GB of RAM.

A. Benchmarks

Safety Gym. We first perform experiments on the OpenAI platform, specifically using the Safety Gym environment [41] PointGoal and CarCircle. The PointGoal task is to control the point to reach the goal (green) while avoiding the hazard (purple) shown in Fig 1. The CarCircle platform is to control the car navigation inside the green circle and avoids colliding with the wall (yellow).

For the PointGoal benchmark, the Point has sensors to observe the distance to the goal and the unsafe region. We set one goal and three hazards in the environment. We denote the d_g and d_c is the distance to the goal and the closest unsafe region. We define the STL specification for the task as below:

$$\Phi = F(d_g < r_g) \wedge G(d_c > r_c)$$

Where the r_g and r_c are the radius of the goal and hazards. Note that the PointGoal is a typical reach-avoid task. We use the dense reward function from existing work [42] that returns the robustness value every time step. The reward function is defined as:

$$R_t = \min_{t'\in[0,t]} \left(\max_{t'\in[0,t]} (r_g - d_g), \min_{t'\in[0,t]} (d_h - r_h) \right)$$

The original reward function in the CarCircle benchmark incentivizes the car to accelerate while ensuring it stays within the circular track. However, the benchmark lacks a specific directive regarding the car's velocity, as STL cannot explicitly formalize this requirement. To address this, we introduce an additional criterion: the car must attain a predefined baseline velocity and maintain this speed once achieved. We consider that the car colliding with the wall is the safety constraint. We emulate this task in accordance with the STL specification as outlined below:

$$\Phi = F(\frac{v}{|r_{car} - r_{circle}|} > v_0) \wedge G(d_c > 0)$$

Where v is the current velocity of the car and v_0 is the required velocity that the car would reach. r_{car} is the distance of the car to the center of the circle. r_{circle} is the radius of the circle. $d_c > 0$ is the requirement that the car should keep a distance from the wall. We generate the reward function as:

$$R_t = \min_{t' \in [0,t]} \left(\max_{t' \in [0,t]} (\frac{v}{|r_{car} - r_{circle}|} - v_0), \min_{t' \in [0,t]} (d_c - r_c) \right)$$

For the two benchmarks, we train the control policy π using PPO [43] for 10 million steps and the reach rate for the control point is greater than 95% with less than 2% violation rate.

Classical control system. We also perform the SVA framework to two classical control systems from the CPS community: DC Motor Position [5], Bicycle [44]. DC Motor position controls the motor angle to a desired position by using the current as control input. The Bicycle has two control inputs: acceleration and steering angle with the goal of speed and steer angle.

We establish a control objective along with two designated unsafe regions for each benchmark. For instance, in the case of the DC Motor position, we define a target motor angle as the objective and designate two other motor angles as unsafe thresholds. The control policy's task is to navigate the angle toward the objective while steering clear of unsafe angles within a predefined time horizon. We quantify the goal and unsafe regions using the 2-norm Euclidean distance metric.

To train the control policy, we employ the Soft Actor-Critic (SAC) algorithm. The training process is carried out for 500,000 iterations, utilizing the reward function which is the robustness of the STL specification similar to Equation 9. Both of the control policies have a greater than 96% reach rate and have less than 1% violation rate.

B. Experiment Setting

We first introduce the SVA performance details under different adversary knowledge levels for the SVA framework. Then we demonstrate the baseline methods for comparison.

SVA framework Setting. We set up SVA methods with different levels of adversary knowledge: White box(WB), grey box with known control policy(GB-C), grey box with known transition function(GB-P), and black box(BB). For all the experience, the set of perturbation set $\beta_\alpha^\epsilon(s)$ is defined as a l_∞ norm ball around s with the budget ϵ. We set different time horizons T for the four benchmarks: 10,000 steps for PointGoal, 5,000 for CarCircle, and 50 for DC motor and Bicycle. The adversary is considered to be effective when it can violate the safety constraint within the predefined time horizon T. We use the SAC algorithm to train the surrogate control policy and PPO to train the adversary model for grey-box and black-box attacks.

Gradient-based Attack. Gradient-based Attack(GA) refers the group of methods from [16], [17]. GA minimizes the value function of the critic $Q^{\text{target}}(s, a)$. It first gets the gradient: $grad = \nabla_s Q^{\text{target}}(s, a)$ and updates the potential adversarial state as $s_i = s - n_i * \frac{grad}{\|grad\|}$ where n_i is the sampled noise.

Learning-based adversary attack. The learning-based adversary attack (LAA) [39] [26] leverages the idea that a control policy π with an observation adversary can be formalized to an SA-MDP $SA-MDPM = (\mathcal{S}, \mathcal{A}, r, \mathcal{B}, p, \gamma)$. The adversary's task can also be viewed as a MDP $\hat{M} = (\mathcal{S}, \hat{\mathcal{A}}, \hat{r}, \hat{p}, \gamma)$ where $\hat{r}(\cdot \mid \cdot) = -r(\cdot \mid \cdot)$. The intuition behind this is that the adversary policy wants to reduce the reward obtained by the victim system resulting in a negative reward for the adversary when the victim system obtains a positive reward.

Maximal Action Difference Attack. Some works consider decreasing the RL return reward by attacking the observation resulting in the system taking suboptimal action. We consider the Maximal Action Difference Attack (MAD) from [40] which has proved to be efficient and simple. The MAD obtains the s' by minimizing: $L_{\text{MAD}}(s') := -D_{\text{KL}}(\pi(s) \| \pi(s'))$ where D_{KL} is the KL-divergence.

Noted, for a fair comparison, all three baseline observation perturbations are constrained by the stealthiness requirement. Additionally, the three baseline methods are under a black box setting. For example, the GA uses the surrogate control policy to compute the gradient to keep in the black box scenario. This ensures that each method is evaluated under consistent conditions, allowing for meaningful comparisons of their effectiveness.

C. Result

We first show the attack results of our SVA and three baseline methods on each benchmark. We evaluate SVA on each benchmark with 500 experiments with random initial points and show the percentage of violations within the time horizon T varies with the perturbation range ϵ in Table II. The experiment is ended if the system reaches the goal or violates the safety. We use perturbation range $\epsilon \in [0.01, 0.05, 0.10, 0.15]$. Results show that even with a small perturbation range, the observation perturbation is still effective.

Observation 1: The white-box SVA demonstrates the highest rate of violations, outperforming all the other methods. The SVA can successfully force the victim system into unsafe even when the perturbation range $\epsilon = 0.01$. Notably, the WB SVA can achieve 90.4% violation rate with $\epsilon = 0.15$ for CarCircle while all the three baseline methods only have less than 17% violation rate. The black-box SVA has the worst performance within the four SVA scenarios but still outperforms the three baselines, which meets our expectations. The performance difference between BB and WB shows that obtaining knowledge of the transition function and control policy indeed increases the possibility of violating safety. This observation also proves that the transition function and control policy are useful to the adversary.

ϵ	PointGoal							CarCircle						
	WB	GB-C	GB-P	BB	GA	LAA	MAD	WB	GB-C	GB-P	BB	GA	LAA	MAD
0.01	5.6%	4.6%	5.4%	5.2%	2%	3.6%	3.6%	23.2%	14%	14.8%	12.8%	0%	5.6%	6.8%
0.05	14%	12%	10%	9.8%	6%	4.8%	4.8%	48.8%	16.4%	17.2%	15.6%	3.4%	7.6%	8.4%
0.10	45%	34.2%	30.6%	20.8%	7.2%	6%	7.2%	84.2%	24.8%	21%	16.8%	9.2%	8%	9.2%
0.15	74.6%	62.4%	52%	38.6%	7.6%	6.8%	11.4%	90.4%	32.6%	26.8%	19.2%	16.8%	12%	10.4%
	DC Motor							Bicycle						
0.01	18.8%	13.2%	15.6%	13.2%	5.6%	8.8%	9.6%	18.4%	16.4%	17.4%	13%	7.2%	8.8%	10.2%
0.05	19.4%	14.8%	16.2%	14.8%	6.4%	12.8%	14%	32.2%	24.6%	23.6%	20%	12.6%	12.6%	13.6%
0.10	33.6%	28.4%	22.8%	19%	13.2%	16.8%	16%	44.8%	38%	35%	25.8%	14%	16.8%	18.2%
0.15	46%	41.2%	24.4%	22.4%	18%	21.2%	16.4%	46.2%	38.6%	35.6%	28.8%	23%	21.2%	20%

Table I: Attack performance measured by the violation rate. ϵ is the perturbation range. The WB, GB-C, GB-P, and BB represent our SVA framework under the white box, a grey box with a known control policy, a grey box with a known transition function, and black box respectively. GA, LAA, and MAD are the three baseline methods introduced in the former subsection. The higher the percentage, the better attack performance to violate the safety.

ϵ	PointGoal							CarCircle						
	WB	GB-C	GB-P	BB	GA	LAA	MAD	WB	GB-C	GB-P	BB	GA	LAA	MAD
0.01	86.4%	89.6%	89.6%	89.6%	93.2%	91.2%	93.6%	72%	84.4%	84%	86.4%	98%	92%	91.6%
0.05	82.8%	86.4%	86.4%	87.6%	91.2%	90%	90.8%	51%	81.6%	79.8%	82.6%	93.4%	89.6%	88%
0.10	52%	60%	80.8%	69.6%	90%	88%	90%	11.8%	71.6%	75.6%	80.4%	86.6%	87.4%	84.6%
0.15	19.6%	32.2%	42.4%	50.4%	88%	84%	85.4%	4.8%	62.6%	70.8%	77.4%	81%	82.8%	83.2%
	DC Motor							Bicycle						
0.01	76%	79.2%	79.2%	79.6%	82%	86.8%	85.2%	73.6%	79.8%	78.2%	83%	90.6%	86.8%	85.8
0.05	75.2%	80%	80%	80.4%	81.8%	82.4%	82.4%	64%	71.8%	73.2%	77%	85.2%	82.4%	78.6
0.10	56.8%	61.2%	73.6%	76%	72.4%	77.2%	82%	48.6%	59.4%	62%	72.4%	80.8%	77.2%	77.4
0.15	45.8%	48.4%	64.4%	68.4%	64.4%	66.8%	76%	46.2%	57%	35.6%	60.2%	72%	66.8%	76.8

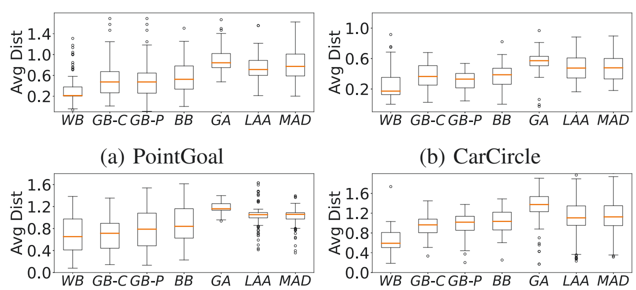
Table II: The reach rate for each benchmark with different adversary algorithms. The lower percentage represents the adversary has more impact on the task completion.

Note that although the SVA does not specifically target a decrease in the reach rate, the SVA still has a significant impact on the reach rate. The WB achieves the highest performance in reducing the reach rate. The WB attack precisely found the vulnerability of the system, making the system incapable of reaching the goal and violating safety.

Observation 2: There is an intriguing observation that when ϵ is small, GB-P demonstrates better efficiency than GB-C. However, this trend does not hold for larger ϵ values, where GB-C is shown to be more efficient. We illustrate the observation using the following claim:

- The adversary knows the control policy means that the generated observation perturbation s' is more potent in reducing the robustness of the safety constraint compared to the action obtained through the adversary model π_{adv} used in GB-P and BB setting.
- When the adversary knows the control policy, it signifies that the produced observation perturbation s' can induce the system to take an action that is closer to the adversary's intended action, while a surrogate control policy which may lead to an action that is not as closely aligned, potentially due to neural network transferability problem. The adversarial perturbation can not always directly transfer to another policy with different algorithms and parameters.

When the perturbation range is small, it becomes challenging for the adversary to generate a corresponding observation perturbation s' with given malicious action u'. This mitigates the significance of knowing the control policy, resulting in GB-C being less efficient compared to GB-P. However, when the perturbation range is large, knowing the control policy becomes crucial. This enables the generation of s' that leads to the precise malicious action, making the attack more effective. Despite GB-P producing more serious action, the observation perturbation generated from the surrogate control policy still lacks some essential information.

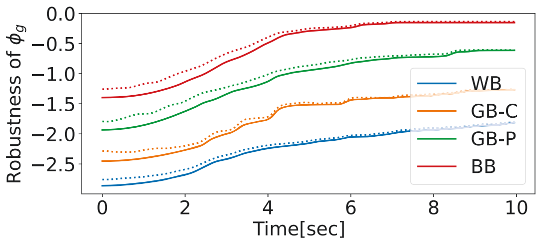


(a) PointGoal (b) CarCircle

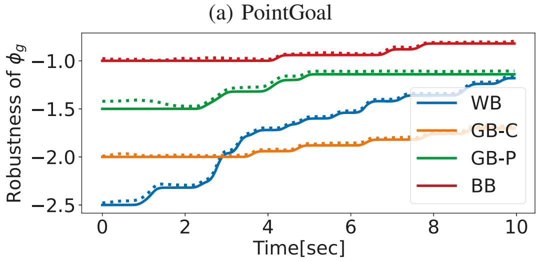
(c) DC Motor (d) Bicycle

Figure 2: The average distance to unsafe for the SVA with four benchmarks under the white box (WB), a grey box with control policy (GB-C), a grey box with transition function (GB-P), and black box (BB).

Observation 3: Fig. 2 displays the average distance to the unsafe in a whole trajectory with 100 experiments using four



(a) PointGoal



(b) CarCircle

Figure 3: The stealthiness measured by the observed robustness value of the goal ϕ_g. The dotted line is the observed robustness after the attack, the solid line is the robustness before the attack. The dotted lines in four attack scenarios are all greater than the solid line which means that our SVA keeps stealthiness.

SVA methods with $\epsilon = 0.1$. The figure shows that the WB achieves the lowest average distance, which means that the WB has the most probability of forcing the system towards unsafe. The BB has the highest average distance indicating a weak attack performance in four SVA scenarios which address the same result as observation 1.

Observation 4: We verified the stealthiness of our SVA and show the results in Fig. 3 using PointGoal and CarCircle benchmarks. We show four different observation histories in each figure in different attack scenarios. We assume the victim system utilizes the manipulated observation to compute the robustness of the goal ϕ_g. The dotted line represents the robustness value after the attack, consistently equal to or greater than the solid line. This confirms that all four SVA scenarios maintain their stealthiness.

VI. Discussion

In this paper, our proposed SVA framework targets the attack on STL-guided RL. A pertinent question arises: Does the SVA compromise the safety of heuristic RL employing a hand-engineered reward function? We assert that, for any safe reinforcement learning problem, the SVA framework has the potential to lead the system into an unsafe region if the adversary possesses knowledge of both safety constraints and the control policy. However, if the adversary lacks access to the control policy, training a surrogate control policy becomes impractical, given the unknown reward function.

Limitation of the SVA. In the white-box attack of the SVA framework, we consider the adversary using the transition function p to obtain the malicious action for the next time step. However, such step-by-step action is not optimal even though it achieves the optimal of the specific step. One of the possible improvements of the solution is to leverage the transition function to compute a sequence of actions $u'_0, u'_1, ..., u'_t$. The sequence of malicious actions to achieve a particular objective in this context can be viewed as a reachability problem, a topic that has been explored in prior works [5]. However, these works compute the attack sequence using a precise system model and model-based controller with bounded noise, this is not the primary focus of our paper.

Defense. While we show the SVA framework induces malicious sensor attacks to force the CPS to take potentially hazardous actions, it is important to note that the impact of SVA can be mitigated or defended through some efforts. We explore some possible solutions to defend against the Safety Violation Attack in cyber-physical systems.

Robust training has been proven to be efficient in mitigating adversarial perturbation and improving policy robustness. Many robust training frameworks rely on adversarial techniques to manually generate attacks, enabling policies to be trained against such adversaries. We consider that robustly training a policy with the SVA adversary, which directly targets sensor attacks that address safety, significantly enhances both the controller's robustness and the overall safety of the system.

Instead of ensuring safety in the training phase, using the prior model to identify the unsafe is also a solution for the SVA adversary. The utilization of formal verification establishes the correctness of system behavior and identifies the unsafe state and behavior, enabling the system to halt operations before an adversary can compel it into an unsafe region. However, it's important to note that the efficacy of this method relies on having an accurate system model. In situations where the environment changes, the system may remain vulnerable to threats if the identification of unsafe states is incomplete or outdated.

VII. Conclusion

In this study, we introduced the Safety Violation Attack (SVA) framework, which provides a novel approach to assess the vulnerability of systems with an STL-guided RL controller. We conduct different attack strategies based on the different levels of the adversary knowledge. We analyze that the existing adversarial attack on RL can not efficiently violate safety. We evaluated the effectiveness of the SVA framework on various benchmarks, including the OpenAI Safety Gym platform. Our results demonstrate the potential risks of deploying STL-guided RL controllers, especially in safety-critical applications. We observe that the SVA framework effectively identified vulnerabilities, highlighting the need for enhanced security measures in such systems.

Acknowledgement

This work was supported in part by NSF CNS-2333980. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the National Science Foundation (NSF).

References

[1] Diego GS Pivoto, Luiz FF de Almeida, Rodrigo da Rosa Righi, Joel JPC Rodrigues, Alexandre Baratella Lugli, and Antonio M Alberti. Cyber-physical systems architectures for industrial internet of things applications in industry 4.0: A literature review. *Journal of manufacturing systems*, 2021.

[2] Mengyu Liu, Lin Zhang, Vir V Phoha, and Fanxin Kong. Learn-to-respond: Sequence-predictive recovery from sensor attacks in cyber-physical systems. In *IEEE Real-Time Systems Symposium*. IEEE, 2023.

[3] Mengyu Liu, Lin Zhang, Pengyuan Lu, Kaustubh Sridhar, Fanxin Kong, Oleg Sokolsky, and Insup Lee. Fail-safe: Securing cyber-physical systems against hidden sensor attacks. In *IEEE Real-Time Systems Symposium*. IEEE, 2022.

[4] Fanxin Kong, Meng Xu, James Weimer, Oleg Sokolsky, and Insup Lee. Cyber-physical system checkpointing and recovery. In *ACM/IEEE 9th International Conference on Cyber-Physical Systems*. IEEE, 2018.

[5] Lin Zhang, Xin Chen, Fanxin Kong, and Alvaro A Cardenas. Real-time attack-recovery for cyber-physical systems using linear approximations. In *IEEE Real-Time Systems Symposium*. IEEE, 2020.

[6] Lin Zhang, Mengyu Liu, and Fanxin Kong. Ai-enabled real-time sensor attack detection for cyber-physical systems. In *AI Embedded Assurance for Cyber Systems*, pages 91–120. Springer, 2023.

[7] Lukas Brunke, Melissa Greeff, Adam W Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 2022.

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

[9] Yixuan Wang, Simon Zhan, Zhilu Wang, Chao Huang, Zhaoran Wang, Zhuoran Yang, and Qi Zhu. Joint differentiable optimization and verification for certified reinforcement learning. In *Proceedings of the ACM/IEEE 14th International Conference on Cyber-Physical Systems (with CPS-IoT Week 2023)*, pages 132–141, 2023.

[10] Simon Sinong Zhan, Yixuan Wang, Qingyuan Wu, Ruochen Jiao, Chao Huang, and Qi Zhu. State-wise safe reinforcement learning with pixel observations. *arXiv preprint arXiv:2311.02227*, 2023.

[11] Alberto Camacho, Rodrigo Toro Icarte, Toryn Q Klassen, Richard Anthony Valenzano, and Sheila A McIlraith. Ltl and beyond: Formal languages for reward function specification in reinforcement learning. In *IJCAI*, volume 19, pages 6065–6073, 2019.

[12] Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*. Springer Berlin Heidelberg, 2004.

[13] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 2019.

[14] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 2019.

[15] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

[16] Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. *Advances in Neural Information Processing Systems*, 2020.

[17] Anay Pattanaik, Zhenyi Tang, Shuijing Liu, Gautham Bommannan, and Girish Chowdhary. Robust deep reinforcement learning with adversarial attacks, 2017.

[18] Mengyu Liu, Pengyuan Lu, Xin Chen, Fanxin Kong, Oleg Sokolsky, and Insup Lee. Fulfilling formal specifications asap by model-free reinforcement learning. *arXiv preprint arXiv:2304.12508*, 2023.

[19] Aniruddh G. Puranic, Jyotirmoy V. Deshmukh, and Stefanos Nikolaidis. Learning from demonstrations using signal temporal logic in stochastic and continuous domains. *IEEE Robotics and Automation Letters*, 2021.

[20] Nathan Fulton and André Platzer. Safe reinforcement learning via formal methods: Toward safe control through proof and learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

[21] Nikhil Kumar Singh and Indranil Saha. Stl-based synthesis of feedback controllers using reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[22] Xiao Li, Yao Ma, and Calin Belta. A policy search method for temporal logic specified reinforcement learning tasks. In *Annual American Control Conference*. IEEE, 2018.

[23] Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In *International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*. Springer, 2004.

[24] Alexandre Donzé and Oded Maler. Robust satisfaction of temporal logic over real-valued signals. In *International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, 2010.

[25] Xiao Li, Cristian-Ioan Vasile, and Calin Belta. Reinforcement learning with temporal logic rewards. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2017.

[26] Yanchao Sun, Ruijie Zheng, Yongyuan Liang, and Furong Huang. Who is the strongest enemy? towards optimal and efficient evasion attacks in deep rl. *arXiv preprint arXiv:2106.05087*, 2021.

[27] Mengdi Huai, Jianhui Sun, Renqin Cai, Liuyi Yao, and Aidong Zhang. Malicious attacks against deep reinforcement learning interpretations. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

[28] Panagiota Kiourti, Kacper Wardega, Susmit Jha, and Wenchao Li. Trojdrl: Evaluation of backdoor attacks on deep reinforcement learning. IEEE, 2020.

[29] Zuxin Liu, Zijian Guo, Zhepeng Cen, Huan Zhang, Jie Tan, Bo Li, and Ding Zhao. On the robustness of safe reinforcement learning under observational perturbations. *arXiv preprint arXiv:2205.14691*, 2022.

[30] Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma, Yan Zheng, Kangjie Chen, and Yang Liu. Stealthy and efficient adversarial attacks against deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[31] Hongkai Chen, Scott A. Smolka, Nicola Paoletti, and Shan Lin. An stl-based approach to resilient control fornbsp;cyber-physicalnbsp;systems. In *Proceedings of the 26th ACM International Conference on Hybrid Systems: Computation and Control*, HSCC '23. ACM, 2023.

[32] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[33] Amir Khazraei, Spencer Hallyburton, Qitong Gao, Yu Wang, and Miroslav Pajic. Learning-based vulnerability analysis of cyber-physical systems. In *ACM/IEEE 13th International Conference on Cyber-Physical Systems*. IEEE, 2022.

[34] Paul Griffioen, Sean Weerakkody, Bruno Sinopoli, Omur Ozel, and Yilin Mo. A tutorial on detecting security attacks on cyber-physical systems. In *18th European Control Conference (ECC)*, 2019.

[35] Amin Ghafouri, Yevgeniy Vorobeychik, and Xenofon Koutsoukos. Adversarial regression for detecting attacks in cyber-physical systems. *arXiv preprint arXiv:1804.11022*, 2018.

[36] Yize Chen, Yuanyuan Shi, and Baosen Zhang. Optimal control via neural networks: A convex approach. *arXiv preprint arXiv:1805.11835*, 2018.

[37] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[38] Yunxiao Qin, Yuanhao Xiong, Jinfeng Yi, and Cho-Jui Hsieh. Training meta-surrogate model for transferable adversarial attack. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[39] Huan Zhang, Hongge Chen, Duane Boning, and Cho-Jui Hsieh. Robust reinforcement learning on state observations with learned optimal adversary. *arXiv preprint arXiv:2101.08452*, 2021.

[40] Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. In *Advances in Neural Information Processing Systems*, 2020.

[41] Joshua Achiam and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. 2019.

[42] Nathaniel Hamilton, Preston K Robinette, and Taylor T Johnson. Training agents tonbsp;satisfy timed andnbsp;untimed signal temporal logic specifications withnbsp;reinforcement learning. Springer-Verlag, 2022.

[43] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[44] Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. Kinematic and dynamic vehicle models for autonomous driving control design. *IEEE Intelligent Vehicles Symposium*, 2015.