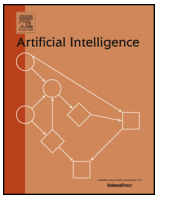
Artificial Intelligence 313 (2022) 103782



Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint



Risk verification of stochastic systems with neural network controllers ☆



Matthew Cleaveland [a,*], Lars Lindemann [a], Radoslav Ivanov [b], George J. Pappas [a]

[a] Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA 19104, USA
[b] Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180, USA

ARTICLE INFO

Article history:
Received 15 October 2021
Received in revised form 24 August 2022
Accepted 27 August 2022
Available online 1 September 2022

Keywords:
Formal verification
Risk-aware systems
Stochastic systems
Neural network controller verification

ABSTRACT

Motivated by the fragility of neural network (NN) controllers in safety-critical applications, we present a data-driven framework for verifying the risk of stochastic dynamical systems with NN controllers. Given a stochastic control system, an NN controller, and a specification equipped with a notion of trace robustness (e.g., constraint functions or signal temporal logic), we collect trajectories from the system that may or may not satisfy the specification. In particular, each of the trajectories produces a robustness value that indicates how well (severely) the specification is satisfied (violated). We then compute risk metrics over these robustness values to estimate the risk that the NN controller will not satisfy the specification. We are further interested in quantifying the difference in risk between two systems, and we show how the risk estimated from a nominal system can provide an upper bound the risk of a perturbed version of the system. In particular, the tightness of this bound depends on the closeness of the systems in terms of the closeness of their system trajectories. For Lipschitz continuous and incrementally input-to-state stable systems, we show how to exactly quantify system closeness with varying degrees of conservatism, while we estimate system closeness for more general systems from data in our experiments. We demonstrate our risk verification approach on two case studies, an underwater vehicle and an F1/10 autonomous car.

© 2022 Elsevier B.V. All rights reserved.

1. Introduction

As autonomous systems become increasingly complex, formally reasoning about system properties becomes more challenging. In particular, verifying systems with machine learning (ML) components (e.g., for perception and control) is difficult as these are often synthesized without any formal guarantees. It has also been shown, see e.g., [1], that ML components periodically fail which is especially concerning as these are increasingly being used in safety critical applications. Given this motivation, the goal of this work is: 1) to estimate the risk of stochastic systems with NN controllers in the loop from data, and 2) to provide conditions under which an NN controller has an acceptable risk for systems that are different from the data-generating system, e.g., due to changing environments that the system operates in. Risk metrics here allow for a trade off between the average and the worst possible case and can hence be used as a tuning knob towards risk-aware autonomy.

☆ This paper is part of the Special Issue: "Risk-aware Autonomous Systems: Theory and Practice".

* Corresponding author.
E-mail addresses: mcleav@seas.upenn.edu (M. Cleaveland), larsl@seas.upenn.edu (L. Lindemann), ivanor@rpi.edu (R. Ivanov), pappasg@seas.upenn.edu (G.J. Pappas).

https://doi.org/10.1016/j.artint.2022.103782
0004-3702/© 2022 Elsevier B.V. All rights reserved.

Depending on the application, there are several ways in which safety can be expressed. The simplest way is to express system safety as a constraint $c(x(t), t) \geq 0$ that has to hold for all $t \in \mathbb{T}$ for a time domain $\mathbb{T} \subseteq \mathbb{N}$, a constraint function $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$, and a system trajectory $x : \mathbb{T} \to \mathbb{R}^n$. Importantly, one can calculate a robustness value $\inf_{t \in \mathbb{T}} c(x(t), t)$ for the trajectory $x(t)$ that serves as a robustness measure. When building autonomous or robotic systems, one is often interested in having the system perform more complex tasks and satisfy spatial and temporal properties that cannot be expressed by a constraint $c(x(t), t) \geq 0$, e.g., a service robot performing a series of re-active tasks with deadlines, while always remaining within a safe zone. A popular way to specify such behaviors is with temporal logics such as the commonly used linear temporal logic (LTL) [2]. In LTL, a system trajectory either satisfies or violates an LTL specification, i.e., LTL has no notion of robustness. Recent effort has gone into adding a notion of robustness to LTL model checking [3] which, however, is quite coarse due to the discrete nature of LTL. As already argued above, it is important to quantify how robustly a trajectory satisfies a given specification. In the aforementioned example, one is interested in distinguishing between trajectories in which the robot remains comfortably within the safe region versus those in which the robot skirts the edge of the safe region. In contrast to LTL, signal temporal logic (STL) allows one to assign a robustness value to a trajectory indicating how robustly it satisfies the specification [4]. Trajectories that robustly satisfy the specification are much less likely to violate it when perturbations or noise affect the system. This notion of STL robustness has been extended to stochastic systems in [5] by considering the probability of satisfying the specification robustly. In our recent work [6], we instead propose to use risk metrics towards considering the risk of not satisfying the specification robustly. This in particular allows for a trade-off between average and worst case scenarios.

Contributions. In this paper, we consider the setup in which we are given a stochastic dynamical control system, an NN controller, and a system specification that the NN controller aims to satisfy. The specification is expressed either as a constraint $c(x(t), t) \geq 0$ for all $t \in \mathbb{T}$ or as an STL specification. An example is an autonomous robot that is controlled by an NN towards safely navigating an unknown environment. Our contributions are as follows:

1. We formulate, for the first time, the risk verification problem of a closed-loop stochastic system with an NN controller by leveraging risk metrics and robustness values associated with the specification at hand.
2. We estimate, from data, the risk of an NN controller not satisfying the specification robustly. The estimated risk has a clear interpretation that may be used to redesign the controller in the case that the verification returns an unacceptably small (or even negative) robustness.
3. We show how the risk estimated from a nominal system can provide an upper bound for the risk of a perturbed version of the system. We refer to the potential increase in risk as the risk verification gap. Particularly, the tightness of this risk verification gap depends on the closeness of the systems in terms of the closeness of their system trajectories. For Lipschitz continuous and incrementally input-to-state stable systems, we show how to exactly quantify system closeness with varying degrees of conservatism, while we estimate system closeness for more general systems from data in our experiments.
4. We present empirical evaluations for end-to-end perception-based reinforcement learning (RL) algorithms. We consider underwater vehicles as well as an autonomous racing car. We show how our risk verification approach can help to select the least risky controller from a set of controllers and empirically illustrate the risk verification gap as well as the effect of different risk metrics.

We claim that our framework is practical in the following ways: 1) we are able to verify high-dimensional systems, 2) we directly verify the closed-loop system of an NN controlled stochastic system, and 3) we are able to quantify the risk verification gap when transferring from a nominal to a perturbed system. We remark that NN controllers are complex and their closed-loop verification imposes significant computational costs, whereas our data-driven framework trades off this computational complexity with the ability to collect data. This framework differs from our previous work [7] because here we are verifying stochastic systems using a data-driven framework, whereas in [7] we had verified deterministic systems using a model-based approach.

The remainder of the paper is organized as follows. A literature review is given in Section 2. In Section 3, we present the two systems considered in this paper (an F1/10 car and an unmanned underwater vehicle) and define the risk verification problem of these systems under NN controllers for constraint functions $c(x(t), t)$. In Section 4, we discuss different risk formulations and show how the risk of an NN controller can be estimated from data. Section 5 defines the risk verification gap between a nominal and a perturbed system and shows under which conditions the risk verification gap is small. In Section 6, we show that our previous results extend in a similar manner when STL specifications are considered instead of constraint functions. We provide our two risk verification case studies in Section 7 and conclude the paper in Section 8.

2. Literature review

Recently, statistical model checking (SMC) has been used to formally reason about complex systems and specifications in a computationally tractable way [8]. The general idea behind SMC is that one can use tools developed from the field of statistics, see e.g., [9], to analyze system traces and obtain guarantees with respect to the overall system behavior. SMC is popular because it is computationally lightweight, especially when compared to computationally expensive model checking

and formal verification. This technique has been used to verify cyber-physical systems (CPS) under linear temporal logic (LTL) specifications [10], biological systems [11], and rare-event properties [12]. A major advantage of the SMC regime is that its guarantees are generally based on finite data by providing probably approximately correct (PAC) guarantees. In other words, verification guarantees are such that with high confidence a given system satisfies a specification with a user-defined probability. Note the appearance of two probabilities where particularly the first stems from the finite data regime, i.e., more data results in a higher confidence.

Verification and control under risk. One drawback of SMC is that one can fall into the trap of only looking at the average performance of a system, whereas reasoning about system safety may require looking at the worst case performance (i.e. the tail end of the distribution). Risk metrics, see e.g., [13] for an introduction, offer a principled way of reasoning about the tail behaviors of a system and provide a meaningful tuning knob in between the two extremes. Risk metrics have been used to synthesize controllers [14] for the common linear quadratic regulator [15], Markov decision processes (MDPs) [16], constrained MDPs [17] service robots in hospital settings [18], optimal energy grid management [19], and reinforcement learning [20]. In addition, risk metrics are increasingly being used to reason about system safety and verification [21,22]. As an alternative to risk metrics, previous works have directly bounded the probability of a system violating a constraint, which are known as chance constraints. They have been used for MDPs [23], partially observable Markov decision processes (POMDPs) [24] and RL [25]. However, these approaches do not account for how robustly a controller satisfies its constraints as we do in this paper.

Verification of ML components. There is a rich literature on the verification of ML components. Most work falls into the open-loop verification category, which seeks to analyze the ML component on its own. For analyzing deep neural networks, most works focus on checking some form of input-output reachability [26–29,10,30]. In addition, work has gone into estimating the Lipschitz constants of deep neural networks [31]. Closed-loop verification, on the other hand, considers a dynamical system that is controlled by an NN controller in a feedback loop [26]. For NN controllers, techniques for the closed-loop verification have appeared in [32–36]. Such formal reasoning approaches often assume full system knowledge, not allowing for stochastic components, and are conservative and computationally expensive.

There also exist works using statistical methods to compute formal guarantees for open-loop ML components. For example, the authors in [37] use Monte-Carlo methods for verifying recurrent neural networks (RNNs). In [38], confidence sets for neural networks are computed with PAC bounds and in [39] an expected-loss minimization framework is used to create PAC bounded classifiers. In addition, work has gone into using statistical methods for estimating the robustness of neural networks [40,41].

Verification under temporal logic specifications. There exists a rich and mature body of literature on model checking under temporal logic specifications [42]. In the context of this work, we are, however, dealing with statistical model checking of temporal properties. In [43] and [44], the authors used statistical testing to derive high confidence bounds on the probability of a CPS with ML components satisfying an STL specification. In [10], LTL hyper-properties of CPS were verified using statistical techniques. Statistical techniques have also been used to verify discrete abstractions of hybrid systems [45], the Toyota powertrain control benchmark [46], and conformance of CPS [47]. Statistical testing under STL specifications was considered in [48–51,5] where [48–50] combine data-driven and model-based verification techniques, while [51,5] present a purely data-driven verification technique to estimate probabilities over robustness distributions of the system.However, their approaches do not consider risk metrics and do not leverage the robustness values of STL.

3. Problem statement

In this section, we first present two case studies on which we then introduce the risk verification problem. The case studies considered in this paper are an F1/10 car in a hallway and a unmanned underwater vehicle (UUV) tracking a pipeline on the seafloor. After introducing these case studies, we introduce an abstract version of the risk verification problem.

3.1. Case studies

F1/10 car. The first case study concerns an F1/10 car [52] driving through a hallway with 90-degree turns, as shown in Fig. 1. The car is equipped with a LiDAR system for sensing the distances to the hallway. We use a standard bicycle model of the car and a ground truth LiDAR model to model the car in simulation. Given this ideal closed-loop simulator setup, we used RL to train NN controllers to safely navigate the car through the hallway. The first question we are interested in is which controller performs best in simulation, i.e., results in the lowest risk. In addition, it has been shown that NN controllers which are verified to be safe in simulation can still show poor performance when evaluated in the real world [7]. So if a controller performs well in simulation, what is the risk of it causing the real F1/10 car to crash into the walls of the hallway?

UUV. The second case study concerns a UUV that is tasked with following alongside a pipeline on the seafloor and using a side-scan sonar system to capture and inspect the pipeline, as shown in Fig. 2. We consider a high-fidelity UUV model based on the ROS-Gazebo real-time physics engine presented in [53]. As the real-time physics engine is too computationally complex to generate large amounts of data, we fit a partially linearized model of the UUV. Using this model, we use RL to



Fig. 1. The F1/10 operating environment. The grey dashed lines on the left represent LiDAR rays and the blue curve represents a safe path through the hallway. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

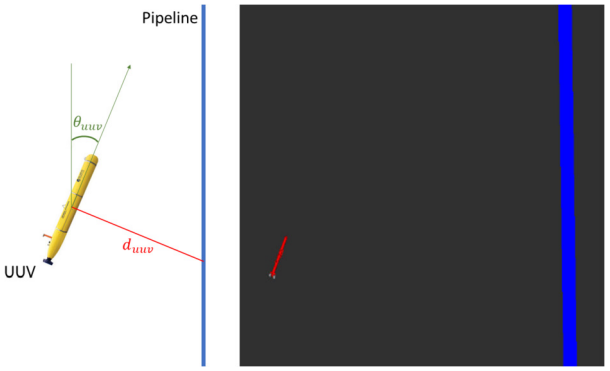


Fig. 2. The UUV operating environment, both in a schematic drawing and in the high fidelity physics engine. The pipelines are both shown in blue.

train NN controllers to guide the UUV along the pipe, while keeping an optimal distance for the sonar to accurately inspect the pipe. Once again, we can ask similar questions to the F1/10 case: which NN controller best guides the UUV along the pipe in the modeled environment? What are the risks of these NN controllers not properly guiding the UUV along the pipe in the real-time physics engine?

3.2. Nominal system model

The focus in this paper is on the verification of *stochastic dynamic control systems with NN controllers* in the loop. In particular, we are interested in verifying systems in terms of the risk of violating a given specification. As illustrated in the case studies, we distinguish between a (nominal) system model and the actual system (which could be a real system or a high-fidelity simulator with complex dynamics). The system model is useful as it permits theoretical analysis and allows one to use computationally-expensive RL techniques. Thus, we model the nominal system as a standard stochastic dynamic system:

$$\begin{aligned} X(t+1) &= f(X(t), u(Y(t)), V(t)),\ X(0) = X_0 \\ Y(t) &= g(X(t), W(t)), \end{aligned} \tag{1}$$

where $X(t) \in \mathbb{R}^n$ is the system state; $Y(t) \in \mathbb{R}^p$ are the observations; the functions $f : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^v \to \mathbb{R}^n$ and $g : \mathbb{R}^n \times \mathbb{R}^w \to \mathbb{R}^p$ describe the system dynamics and the output measurement map, respectively; the function $u : \mathbb{R}^p \to \mathbb{R}^m$ is an NN controller; and $V(t) \in D_v \subseteq \mathbb{R}^v$ and $W(t) \in D_w \subseteq \mathbb{R}^w$ describe process and measurement noise where the sets D_v and D_w are specified later. The initial position $X_0 \in \mathbb{R}^n$ as well as the process and measurement noise $V(t)$ and $W(t)$ are assumed to be random variables so that X defines a stochastic process.

3.3. Risk abstraction

For clarity, in what follows we provide an abstract notion of risk – the full formalization is presented in Section 4. We consider systems subject to real-time safety constraints (e.g., the F1/10 car needs to avoid colliding with walls at all times). One way to express real-time constraints is as follows:

$$c(x(t), t) \geq 0 \text{ for all } t \in \mathbb{T} \tag{2}$$

where $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$ is a measurable constraint function (e.g., hyper-planes representing walls in the F1/10 example) and $x(t)$ can be a realization of the system trajectory as in (1).[1] We quantify the robustness of a given trajectory $x(t)$ by measuring how much one would need to alter $x(t)$ to cause it to violate (or satisfy) the constraint c. For example, consider a trace of the F1/10 car where the car comes within 1 m of colliding with the wall. We would say that this trace has a robustness of 1.

Since the nominal system in (1) is stochastic, the robustness value in (2) is itself a random variable. This is why we are interested in computing risk metrics, e.g., as presented in [13], over the robustness value to quantify the risk of not satisfying the specification robustly, which we call the *robustness risk*. Commonly used risk metrics are the expected value, the variance, the value-at-risk, or the conditional value-at-risk [54]. We formally introduce risk metrics in Section 4.

3.4. Real system

Besides estimating the risk of the nominal system in (1), we are further interested in quantifying the difference in risk between the nominal system and a system that is different from (1), referred to as the real system and denoted as $\overline{X}$. We investigate two ways of describing the relationship between the nominal and the real system, which can then be used to bound the real system's risk with respect to the nominal system. In the first approach, which is standard in the control theory literature, we assume that the real system is captured by the model in (1), but for different distributions of V and W, i.e., a perturbed version of (1). However, we also note that this model may not be sufficient to describe a complex real system (or even a high-fidelity simulator). Thus, in the second approach, we analyze how the closeness between the nominal system in (1) and the real system, potentially very different from (1), affects the difference between their risks.

Our main insight is that the difference in risk between these two systems naturally depends on the similarity of the systems and can be bounded in terms of the difference of their system trajectories. Thus, as long as the nominal and real systems are not significantly different, we can provide bounds on the real system's risk with respect to the nominal system. In our experiments, we assume to have access to trajectories from both the nominal and the real system to validate our results.

3.5. Problem formulation

We first note that calculating the robustness risk for systems with NN controllers is itself a novel problem. Thus, we address that problem first.

Problem 1 *(Risk verification).* Assume the stochastic control system in (1) and a measurable constraint function $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$. Assume a set of NN controllers $\mathcal{U}$ is given. Calculate the robustness risks for the system in (1) under each NN controller $u_i \in \mathcal{U}$.

To solve Problem 1, we use data sampled from the system (1) under the NN controller u_i to obtain an *upper bound* of the robustness risk with high probability following ideas in our previous work [6], but here applied to a closed-loop setting. We emphasize that we are interested in an upper bound of the robustness risk to be risk averse as opposed to being risk neutral by estimating the mean of the robustness risk. Our main goal in Problem 1 is to analyze the effect of different risk metrics and different risk levels that can act as a tuning knob between risk averse and risk neutral decision making. We also argue that robustness should be used as a quality measure compared to existing verification techniques that rely on a binary verification answer.

The second problem that we address in this paper is to investigate how the robustness risk, calculated in Problem 1, can be used to determine how the NN controllers perform on systems that are different from (1), i.e., how the robustness risk can aid in determining if an NN controller is robust enough to be used on the real system. As mentioned above, we consider two versions of this problem: 1) the real system is a perturbed version of the model; 2) the real system dynamics are potentially very different from (1).

Problem 2 *(Risk verification gap for perturbed systems).* Consider the stochastic control system in (1) and a perturbed stochastic process $\overline{X}$ that also follows (1), but for different distributions of V and W. Two specific cases of (1) that we consider are Lipschitz continuous and incrementally input-to-state stable systems. We want to find a constant $\Delta \geq 0$ so that the difference in robustness risk between X and $\overline{X}$ is bounded by Δ.

As already emphasized, we note that the model in (1) may not be sufficient to describe a complex real system (or even a high-fidelity simulator) and we may not be able to find Δ as in Problem 2. We want to quantify the risk verification gap of such systems in terms of the closeness of the systems.

[1] In Section 6, we also consider signal temporal logic (STL) constraints as a more general way of expressing statements close to structured English.

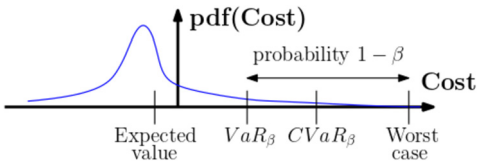


Fig. 3. Visualization of common risk metrics.

Problem 3 *(Expected risk verification gap for real system).* Consider the stochastic control system in (1) and a real system that may be different from (1). Using trajectories drawn from the two systems, find a constant $\Delta \geq 0$ so that the risk verification gap between the two systems is bounded by Δ with high confidence.

Remark 1. The insights from Problems 2 and 3 also provide a practical way to compare NN controllers on the nominal system. In particular, the controller that achieves the lowest risk on the nominal system is likely to perform best on the real system as well. The experimental evaluation confirms this observation.

4. Risk verification

We now present our risk verification framework for solving Problem 1.

4.1. Random variables, risk measures, and stochastic processes

Consider the *probability space* $(\Omega, \mathcal{F}, P)$ where Ω is the sample space, $\mathcal{F}$ is a σ-algebra of Ω, and $P : \mathcal{F} \to [0, 1]$ is a probability measure. More intuitively, an element in Ω is an *outcome* of an experiment, while an element in $\mathcal{F}$ is an *event* that consists of one or more outcomes whose probabilities can be measured by the probability measure P.

Let Z denote a real-valued *cost random variable*, i.e., a measurable function $Z : \Omega \to \mathbb{R}$. Let $\mathfrak{F}(A, B)$ denote the set of all functions which map input domain A to output domain B (e.g. $Z \in \mathfrak{F}(\Omega, \mathbb{R})$). A *risk metric* is a function $R : \mathfrak{F}(\Omega, \mathbb{R}) \to \mathbb{R}$ that maps from the set of real-valued random variables to the real numbers. The value of $R(Z)$ quantifies the risk associated with cost random variable Z. Risk metrics hence allow for a risk assessment in terms of such cost random variables. Commonly used risk metrics are the expected value, the variance, the value-at-risk, or the conditional value-at-risk [54], see Fig. 3 for a visualization of these risk metrics.

Particular properties of R that we need in this paper are *monotonicity*, *translational invariance*, and *subadditivity*:

- For two cost random variables $Z, Z' \in \mathfrak{F}(\Omega, \mathbb{R})$, the risk metric R is monotone if $Z(\omega) \leq Z'(\omega)$ for all $\omega \in \Omega$ implies that $R(Z) \leq R(Z')$.
- For a random variable $Z \in \mathfrak{F}(\Omega, \mathbb{R})$, the risk metric R is translationally invariant if, for any constant $c \in \mathbb{R}$, it holds that $R(Z + c) = R(Z) + c$.
- For two cost random variables $Z, Z' \in \mathfrak{F}(\Omega, \mathbb{R})$, the risk metric R is subadditive if $R(Z + Z') \leq R(Z) + R(Z')$.

Remark 2. In Appendix A, we summarize other desirable properties of R such as positive homogeneity, comotone additivity, and law invariance. We also provide a summary of existing risk metrics and emphasize that our presented method is compatible with any of these risk metrics as long as they are monotone, translationally invariant, and subadditive.

For a time domain $\mathbb{T} \subseteq \mathbb{N}$, a *stochastic process* is a function $X : \mathbb{T} \times \Omega \to \mathbb{R}^n$ where $X(t, \cdot)$ is a random vector for each fixed $t \in \mathbb{T}$. A stochastic process can be viewed as a collection of random vectors $\{X(t, \cdot) | t \in \mathbb{T}\}$ that are defined on a common probability space $(\Omega, \mathcal{F}, P)$. For a fixed $\omega \in \Omega$, the function $X(\cdot, \omega)$ is a *realization* of the stochastic process. Note that the stochastic dynamics system in (1) is a particular form of a stochastic process.

4.2. The robustness risk

Now we formally define constraint functions, constraint safe sets, trace robustness and robustness risk. In this section, we consider specifications as $c(x(t), t) \geq 0$ for all $t \in \mathbb{T}$.

Example 1. For the F1/10 case study where the car must avoid colliding into the walls of a hallway, we can choose

$$c(x(t), t) := d_w(x(t)) - \delta$$

where $d_w : \mathbb{R}^n \to \mathbb{R}$ returns the minimum distance from the current car position to the walls and $\delta \in \mathbb{R}_{\geq}$ is the minimum safe distance between the car and the walls. The constraint $c(x(t), t) \geq 0$ is, in this case, equivalent to $d_w(x(t)) \geq \delta$ for all $t \in \mathbb{R}_{\geq 0}$.

Importantly, we can quantify how robustly a signal x satisfies the constraint $c(x(t), t) \geq 0$. Therefore, let us define the sets

$$O^c(t) := \{x \in \mathbb{R}^n | c(x, t) \geq 0\}$$
$$O^{\neg c}(t) := \{x \in \mathbb{R}^n | c(x, t) < 0\}$$

that indicate the part of the state space where $x(t)$ satisfies and violates the constraint $c(x(t), t) \geq 0$, respectively. We can now define the signed distance $\text{Dist}^c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$ from $x(t)$ to the set $O^c(t)$ as

$$\text{Dist}^c(x(t), t) := \begin{cases} \inf_{x' \in \text{cl}(O^{\neg c}(t))} \|x(t) - x'\| & \text{if } c(x(t), t) \geq 0 \\ -\inf_{x' \in \text{cl}(O^c(t))} \|x(t) - x'\| & \text{otherwise} \end{cases}$$

where $\text{cl}(\cdot)$ denotes the closure operation of a set. In words, $\text{Dist}^c(x(t), t)$ denotes how much one would need to alter the value of signal x at time t to have it violate (satisfy) constraint $c(x(t), t)$, provided $x(t)$ already satisfies (violates) the constraint at time t. For instance, a value of $\text{Dist}^c(x(t), t) = 2$ means that $x(t)$ would need to be altered by at least 2 in order to violate c at time t (and vice versa if $\text{Dist}^c(x(t), t) = -2$).

Let us next define the *robustness function* $\rho^c : \mathfrak{F}(\mathbb{T}, \mathbb{R}^n) \to \mathbb{R}$, where $\mathfrak{F}(\mathbb{T}, \mathbb{R}^n)$ denotes the set of all measurable functions mapping from the domain $\mathbb{T}$ into the domain $\mathbb{R}^n$, as

$$\rho^c(x) := \inf_{t \in \mathbb{T}} \text{Dist}^c(x(t), t).$$

Naturally, notice that $\rho^c(x) \geq 0$ implies that the specification is satisfied by x, i.e., that $c(x(t), t) \geq 0$ for all $t \in \mathbb{T}$. If $\rho^c(x) > 0$, notice further that all signals $x' : \mathbb{T} \to \mathbb{R}^n$ that are such that $\|x(t) - x'(t)\| < \rho^c(x)$ for all $t \in \mathbb{T}$ also satisfy the specification, i.e., are such that $\rho^c(x') > 0$.

In the remainder, we will not evaluate the constraint function c over a deterministic signal x, but instead over a stochastic process X, i.e., we aim to use $c(X(t, \cdot), t)$ and the robustness of stochastic systems. Therefore, we compute risk metrics of the robustness random variable as $R(-\rho^c(X))$ where we consider $-\rho^c(X)$ as the cost random variable. Note that $-\rho^c(X)$ needs to be a random variable, i.e., $-\rho^c(X(\cdot, \omega))$ needs to be measurable in ω, which we show to hold in Appendix B.

4.3. Estimating an upper bound of the robustness risk

As the system in (1) defines a stochastic process, Problem 1 can be solved by estimating upper bounds of the risk $R(-\rho^c(X))$ using frameworks as presented in [55–57]. In this paper, we particularly use the commonly studied risk metrics of the value-at-risk and the conditional value-at-risk. We estimate an upper bound of the robustness risk $R(-\rho^c(X))$ from N observed realizations $X^1, \ldots, X^N$ of the stochastic process X. Let us, for convenience, first define the random variable

$$Z := -\rho^c(X).$$

For further convenience, let us define the tuple

$$\mathcal{Z} := (Z^1, \ldots, Z^N)$$

where $Z^i := -\rho^c(X^i, t)$.

Value-at-Risk (VaR): For a risk level of $\beta \in (0, 1)$, the VaR of Z is given by

$$VaR_\beta(Z) := \inf\{\alpha \in \mathbb{R} | F_Z(\alpha) \geq \beta\}$$

where $F_Z(\alpha)$ denotes the CDF of Z. To estimate $F_Z(\alpha)$, define the empirical CDF

$$\widehat{F}(\alpha, \mathcal{Z}) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(Z^i \leq \alpha)$$

where $\mathbb{I}$ denotes the indicator function. From the empirical CDF, the empirical $VaR_\beta(Z)$ is obtained as follows:

$$\widehat{VaR}_\beta(\mathcal{Z}) = \inf\left\{\alpha \in \mathbb{R} | \widehat{F}(\alpha, \mathcal{Z}) \geq \beta\right\}$$

Now we can compute an upper bound of $VaR_\beta(Z)$ as follows:

Proposition 1 *([55]). Let $\delta \in (0, 1)$ be a probability threshold and $\beta \in (0, 1)$ be a risk level. Let $Z^1, \ldots, Z^N$ be N independent copies of the random variable Z. Then, with probability at least $1 - \delta$ it holds that*

$$VaR_\beta(Z) \leq \widehat{VaR}_{\beta + c_N(\delta)}(\mathcal{Z})$$

where

$$c_N(\delta) := \sqrt{\frac{1}{2N} log\Big(\frac{\pi^2 N^2}{3\delta}\Big)}.$$

Conditional Value-at-Risk (CVaR): For a risk level of $\beta \in (0,1)$, the CVaR of Z is given by

$$CVaR_\beta(Z) := \inf_{\alpha \in \mathbb{R}} \alpha + (1-\beta)^{-1} E([Z-\alpha]^+)$$

where $[Z-\alpha]^+ := \max(Z-\alpha, 0)$. For estimating $CVaR_\beta(Z)$ from data $\mathcal{Z}$, we focus here on the case where the random variable $Z = -\rho^\phi(X,t)$ has bounded support (for a fixed t) and leverage results from [57]. In particular, we assume that $P(Z \leq b) = 1$ for a known constant b. Note that Z has bounded support when $\rho^\phi(X,t)$ is bounded, which can be achieved either by construction of ϕ or by clipping off $\rho^\phi(X,t)$ outside the interval $(-\infty, b]$. We can then compute an upper bound of $CVaR_\beta(Z)$ as follows:

Proposition 2 *([57]). Let $\delta \in (0, 0.5]$ be a probability threshold and $\beta \in (0,1)$ be a risk level. Assume that $P(Z \leq b) = 1$. Let $Z^1, \ldots, Z^N$ be N independent copies of the random variable Z. Then, with probability at least $1-\delta$ it holds that*

$$CVaR_\beta(Z) \leq O_{N+1} - \frac{1}{\beta} \sum_{i=1}^{N} (O_{i+1} - O_i) \left[\frac{i}{N} - \sqrt{\frac{\ln(1/\delta)}{2N}} - (1-\beta) \right]^+$$

where $x^+ := max\{x, 0\}$, $O_1, \ldots, O_N$ are the order statistics (i.e. $Z^1, \ldots, Z^N$ sorted in ascending order) and $O_{N+1} := b$.

5. The risk verification gap between nominal and perturbed systems

In this section, we are interested in quantifying the difference in risk between two systems and understanding their relation. We want to find an upper bound on the robustness risk of the real system $R(-\rho^c(\overline{X}))$ in terms of the robustness risk of the nominal system $R(-\rho^c(X))$. We would hence like to find a bound on the *risk verification gap* $R(-\rho^c(\overline{X})) - R(-\rho^c(X))$. We will show that the risk verification gap can be bounded by the maximal error between the trajectories of the nominal and the perturbed system $\|\overline{X}(t,\omega) - X(t,\omega)\|$. Let us, for now, assume that the trajectory error can be bounded by Δ, i.e., assume that

$$\|\overline{X}(t,\omega) - X(t,\omega)\| \leq \Delta \tag{3}$$

for all realizations $\omega \in \Omega$ and all times $t \in \mathbb{T}$. Later in this section, we present three ways of how one may obtain Δ. Our first goal is to quantify how the bound Δ affects the difference in robustness, as measured by the function ρ^c, by which system trajectories of (1) and $\overline{X}$ satisfy the specification.

Lemma 1. *Consider the stochastic control systems in (1) under a controller u and $\overline{X}$, and assume that the trajectory error Δ in (3) is known. Let $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$ be a measurable constraint function. Then it holds that*

$$\rho^c(X(\cdot,\omega)) - \Delta \leq \rho^c(\overline{X}(\cdot,\omega)) \tag{4}$$

for each realization $\omega \in \Omega$.

Proof. See Appendix C. □

Now that we have quantified in Lemma 1 how close the robustness of the systems in (1) and $\overline{X}$ will be, we can directly use this result to bound the risk verification gap under the assumption that the risk metric R is monotone and translationally invariant.

Theorem 1. *Let the same conditions as in Lemma 1 hold and let the risk metric R be monotone and translationally invariant. Then it holds that*

$$R(-\rho^c(\overline{X})) \leq R(-\rho^c(X)) + \Delta.$$

Proof. See Appendix D. □

We can further use the risk verification gap to compare controllers in terms of their robustness risk. Specifically, we are interested in determining if a controller that has a lower robustness risk on the nominal system than another controller will also have a lower risk on the perturbed system. From the proofs of Lemma 1 and Theorem 1 one can, under the same assumption, immediately see that also the inequality $R(-\rho^c(X)) \leq R(-\rho^c(\overline{X})) + \Delta$ holds. The next result is a straightforward consequence of these results and hence provided without proof.

Corollary 1. *Let the same conditions as in Theorem 1 hold. Assume that $R(-\rho^c(X_{u_1})) \leq R(-\rho^c(X_{u_2})) - c$ for two controllers u_1 and u_2. Then it holds that $R(-\rho^c(\overline{X}_{u_1})) \leq R(-\rho^c(\overline{X}_{u_2}))$ if $2\Delta \leq c$.*

Recall that the bound on the risk verification gap hinges on knowing the bound Δ. We next present three ways to calculate or estimate Δ.

Lipschitz constants. A first, but potentially conservative approach, is to quantify Δ in terms of Lipschitz constants of the system. Let us therefore consider that $\overline{X}$ is produced by the perturbed system

$$\begin{aligned} \overline{X}(t+1) &= f(\overline{X}(t), u(\overline{Y}(t)), \overline{V}(t)),\ \overline{X}(0) = X_0 \\ \overline{Y}(t) &= g(\overline{X}(t), \overline{W}(t)), \end{aligned} \tag{5}$$

where f, g, and u are as in (1), but where the process and measurement noise is different from V and W and instead described by the random variables $\overline{V}(t) \in D_v \subseteq \mathbb{R}^n$ and $\overline{W}(t) \in D_w \subseteq \mathbb{R}^p$. Additionally, assume that f, g, and u are Lipschitz continuous with known Lipschitz constants and that the sets D_v and D_w are compact. Note that Lipschitz constants of NN controller u can be calculated as, for instance, described in [31].

At time t, and without any further assumption on the controller u, we can upper bound the error $\|\overline{X}(t,\omega) - X(t,\omega)\|$ between the system trajectories of the nominal system (1) and the perturbed system (5) for each realization $\omega \in \Omega$ by the function

$$\Delta(t+1) := L_{f,1}\Delta(t) + L_{f,2}L_u(L_{g,1}\Delta(t) + L_{g,2}w^*) + L_{f,3}v^* \tag{6}$$

where $\Delta(0) := 0$ and where $L_{f,1}$, $L_{f,2}$, and $L_{f,3}$ are the Lipschitz constants of the first, second, and third arguments of f, L_u is the Lipschitz constant of u, $L_{g,1}$ and $L_{g,2}$ are the Lipschitz constants of g, and v^* and w^* are calculated as

$$v^* := 2 \max_{v \in D_v} \|v\| \quad \text{and} \quad w^* := 2 \max_{w \in D_w} \|w\|.$$

In other words, it holds that

$$\|\overline{X}(t,\omega) - X(t,\omega)\| \leq \Delta(t).$$

The derivation of the bound $\Delta(t)$ is shown in Appendix F. Note now that the bound $\Delta(t)$ depends on time, and in fact grows with time, so that Theorem 1 can not be applied directly. However, when we only consider a bounded time interval over which the constraint function c is evaluated, we can derive a corresponding result. The next result hence follows by a minor modification of Lemma 1 and Theorem 1 and is stated without proof.

Corollary 2. *Consider the stochastic control systems in (1) and (5) under a controller u and assume that the trajectory error $\Delta(t)$ is as in (6). Let $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$ be a measurable constraint function with $T := \max\{\mathbb{T}\}$ being bounded. Let the risk metric R be monotone and translationally invariant. Then it holds that*

$$R(-\rho^c(\overline{X})) \leq R(-\rho^c(X)) + \Delta(T).$$

As remarked previously, such Lipschitz bounds for Δ may potentially be conservative. Similar bounds, in a different context, can for instance be obtained by considering Lipschitz continuous risk metrics, see [58, Definition 4.2].

Incrementally input-to-state stable controller. While still considering the perturbed system in (5), we may avoid such conservatism and instead use tools from robust control. Particularly, we can assume that the controller u is such that the system in (1) is incrementally input-to-state stable (see e.g., [59]). Let $D := (V, W)$ and $D_D := D_v \times D_w$, then the system (1) is incrementally input-to-state stable if there exists a function $\gamma \in \mathcal{K}_\infty$[2] such that

$$\|X(t,\omega; D_1) - X(t,\omega; D_2)\| \leq \gamma\Big(\sup_{d_1, d_2 \in D} \|d_1 - d_2\|\Big)$$

[2] A function $\gamma : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is said to be of class $\mathcal{K}$ if it is continuous, strictly increasing, and $\gamma(0) = 0$. It is of class $\mathcal{K}_\infty$ if furthermore $\lim_{s\to\infty} \gamma(s) = \infty$.

for every realization $\omega \in \Omega$ where $X(t, \omega; D_i)$ is the solution to (1) but where D is replaced with any stochastic process $D_i : \mathbb{N} \times \Omega \to D_D$. While this may seem like a stringent assumption on the system, learning controllers that exhibit iISS or similar stability properties is an active area of research [60–62].

It follows in this case that the error bound between the system trajectories of the nominal system (1) and the perturbed system in (5) are independent of t, i.e., that

$$\|\overline{X}(t, \omega) - X(t, \omega)\| \leq \gamma(\sup_{d_1, d_2 \in D} \|d_1 - d_2\|)$$

so that we can set $\Delta := \gamma(\sup_{d_1, d_2 \in D} \|d_1 - d_2\|)$ and simply apply Theorem 1.

Remark 3. Note that any stable linear system is iISS. This can be straightforwardly shown by plugging in the system dynamics into the iISS definition.

Stochastic bounds. Let us now return back to the generic perturbed system $\overline{X}$ from Section 3.4. Note that we have so far assumed a worst case upper bound Δ as in (3). Let us now assume that the trajectory error is bounded by a random variable, i.e., there exists $\Gamma : \Omega \to \mathbb{R}$ such that

$$\|\overline{X}(t, \omega) - X(t, \omega)\| \leq \Gamma(\omega) \tag{7}$$

for each realization $\omega \in \Omega$ and all times $t \in \mathbb{T}$. In this case, it can be easily seen that Lemma 1 still holds while using $\Gamma(\omega)$ instead of Δ. Based on this observation, we can bound the risk verification gap by $R(\Gamma)$ if we assume that the risk metric is subadditive instead of being translationally invariant.

Theorem 2. *Consider the stochastic control systems in (1) and $\overline{X}$ under a controller u and assume that the trajectory error $\Gamma(\omega)$ in (7) is known. Let $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$ be a measurable constraint function and let the risk metric R be monotone and subadditive invariant. Then it holds that*

$$R(-\rho^c(\overline{X})) \leq R(-\rho^c(X)) + R(\Gamma).$$

Proof. See Appendix E. □

We note that the bound in Theorem 2 is tighter than the bound in Theorem 1. However, the downside is that one will have to estimate $R(\Gamma)$, which will require having knowledge of trajectories of $\overline{X}$ which one could use to directly calculate $R(-\rho^c(\overline{X}))$. An empirical estimation of $R(\Gamma)$ also requires generating sample pairs from the nominal and real systems that feature the same corresponding realizations ω across all random variables, which may be possible in a sim2sim but not necessarily possible in a sim2real setting. Nonetheless, this approach offers additional insight and a way of quantifying the risk gap between two systems.

6. Signal Temporal Logic (STL)

Signal temporal logic [4] specifications allow for more expressivity than constraint functions $c(x, t)$ that were discussed so far. We now extend the previously derived results to such specifications.

6.1. Risk verification under signal temporal logic specifications

STL specifications are constructed from predicates $\mu : \mathbb{R}^n \to \{\top, \bot\}$ where $\top := \infty$ and $\bot := -\infty$ encode logical true and false, respectively. Let us associate an observation map $O^\mu \subseteq \mathbb{R}^n$ with μ as

$$O^\mu := \mu^{-1}(\top)$$

where $\mu^{-1}(\top)$ denotes the inverse image of $\top$ under μ. The set O^μ hence denotes the parts of the state space where the predicate μ is true. Often, predicates are defined via a predicate function $h : \mathbb{R}^n \to \mathbb{R}$ as

$$\mu(\zeta) := \begin{cases} \top & \text{if } h(\zeta) \geq 0 \\ \bot & \text{otherwise} \end{cases}$$

for $\zeta \in \mathbb{R}^n$ so that $O^\mu = \{\zeta \in \mathbb{R}^n | h(\zeta) \geq 0\}$. The syntax of STL is defined as

$$\phi ::= \top \mid \mu \mid \neg\phi \mid \phi' \wedge \phi'' \mid \phi' U_I \phi'' \tag{8}$$

where ϕ' and ϕ'' are STL formulas and where U_I is the until operator with $I \subseteq \mathbb{R}_{\geq 0}$. The until operator encodes that ϕ' always has to hold until ϕ'' holds at some point within the interval I. The meanings of the negation and conjunction

operators $\neg$ and $\wedge$ are 'not' and 'and'. One can further define the disjunction, eventually, and always operators as $\phi' \vee \phi'' := \neg(\neg\phi' \wedge \neg\phi'')$, $F_I\phi := \top U_I \phi$, and $G_I\phi := \neg F_I \neg\phi$, respectively. We provide the semantics of STL [4] that determine whether or not a signal x satisfies an STL formula ϕ at time t in Appendix G. An STL formula ϕ is bounded if all time intervals I in ϕ are bounded. Each bounded STL formula ϕ has a formula length L^ϕ that defines the minimum length a signal must be in order to be able to say whether or not it satisfies ϕ, see Appendix G for a formal statement.

Robustness. One may be especially interested in the robustness by which a signal x satisfies the STL formula ϕ at time t. For this purpose, the authors in [63] introduce the robustness degree and in particular the function $\text{dist}^{\neg\phi} : \mathfrak{F}(\mathbb{N}, \mathbb{R}^n) \times \mathbb{N} \to \mathbb{R}_{\geq 0}$. Intuitively, $\text{dist}^{\neg\phi}(x,t)$ measures the distance of the signal x to the set of signals that violate ϕ at time t (see Appendix H for a formal definition). It holds that $\text{dist}^{\neg\phi}(x,t) = 0$ indicates that x violates or marginally satisfies ϕ, while $\text{dist}^{\neg\phi}(x,t) > 0$ indicates that x satisfies ϕ. In fact, larger values of $\text{dist}^{\neg\phi}(x,t)$ indicate that ϕ is satisfied more robustly. Unfortunately, $\text{dist}^{\neg\phi}(x,t)$ is in general difficult to calculate. The authors in [63] introduce the *robust semantics* $\rho^\phi : \mathfrak{F}(\mathbb{N}, \mathbb{R}^n) \times \mathbb{N} \to \mathbb{R}$ as a tractable but more conservative robustness estimate, i.e., it holds that $\rho^\phi(x,t) \leq \text{dist}^{\neg\phi}(x,t)$ [63, Theorem 28]. A formal definition is given in Appendix H.

Instead of computing the STL robustness of deterministic traces, we are interested in evaluating the STL robustness over a stochastic process X. We use random variable $\text{dist}^\phi(X,t)$ to denote the robustness values of X for specification ϕ. The risk of the stochastic process X not satisfying ϕ robustly, called the "STL robustness risk" [6], is defined to be $R(-\text{dist}^{\neg\phi}(X,t))$. As previously remarked, the function $\text{dist}^{\neg\phi}$ is in general hard to calculate and work with. In [6, Thm. 3], we show that

$$R(-\text{dist}^{\neg\phi}(X,t)) \leq R(-\rho^\phi(X,t))$$

if R is a monotone risk metric. In other words, $R(-\rho^\phi(X,t))$ is more risk-averse than $R(-\text{dist}^{\neg\phi}(X,t))$ and may instead be used. Now, by setting $Z := -\rho^\phi(X,t)$, we can estimate an upper bound to the STL robustness risk in the same way as discussed in Section 4.3.

6.2. The risk verification gap for signal temporal logic

For STL specifications ϕ we can quantify the risk verification gap similarly to Section 5 where we considered specifications defined by $c(x,t)$. The next Lemma directly resembles Lemma 1.

Lemma 2. *Consider the stochastic control systems in (1) under a controller u and $\overline{X}$. Let ϕ be an STL formula in positive normal form[3] and assume that the trajectory error Δ in (3) is known. Then it holds that*

$$\rho^\phi(X(\cdot,\omega),t) - \Delta \leq \rho^\phi(\overline{X}(\cdot,\omega),t)) \tag{9}$$

for each realization $\omega \in \Omega$ and time $t \in \mathbb{N}$. If additionally ϕ is a bounded STL formula in positive normal form with formula length L^ϕ and the trajectory error $\Delta(t)$ is as in (6), then it holds that

$$\rho^\phi(X(\cdot,\omega),t) - \Delta(t + L^\phi) \leq \rho^\phi(\overline{X}(\cdot,\omega),t)) \tag{10}$$

for each realization $\omega \in \Omega$ and time $t \in \mathbb{N}$.

Proof. See Appendix I. □

Based on this lemma, we can state similar results as in Theorem 1 and Corollary 2 when considering STL specifications ϕ.

Theorem 3. *Let the conditions of Lemma 2 hold and let the risk metric R be monotone and translationally invariant. If ϕ is an STL formula in positive normal form and the trajectory error Δ in (3) is known, it holds that*

$$R(-\rho^\phi(\overline{X},t)) \leq R(-\rho^\phi(X,t)) + \Delta.$$

If additionally ϕ is a bounded STL formula in positive normal form with formula length L^ϕ and the trajectory error $\Delta(t)$ is as in (6), then it holds that

$$R(-\rho^\phi(\overline{X},t)) \leq R(-\rho^\phi(X,t)) + \Delta(t + L^\phi).$$

Proof. See Appendix J. □

[3] A formula ϕ is in positive normal form if ϕ contains no negations, or negations only appear in front of predicates so that they can be absorbed into them. This is without loss of generality as it is known that every STL formula can be re-written in positive normal form [64].

7. Case studies

7.1. F1/10

Recall the F1/10 system described in Section 3, where an F1/10 car must travel through a hallway using LiDAR and an NN controller. The car must travel through the hallway without colliding into any walls, which is captured with the following constraint function:

$$c(x(t), t) := d(x(t)) > d_w \tag{11}$$

7.1.1. Nominal system

We used a standard bicycle model as the model of the car, denoted as f. The LiDAR model, denoted as g, returns the ground truth value of each of the 21 LiDAR rays (see the grey dashed lines in Fig. 1 for what each LiDAR ray looks like). There is uniform noise $W(t)$ applied to each LiDAR ray's distance. The controller, denoted as NN, takes arrays of LiDAR rays as input and returns steering commands as output. Equation (12) shows the F1-10th with simulated LiDAR written in the form of (1). The state is $X = \begin{bmatrix} x & y & v & \theta \end{bmatrix}^T$, where x and y are the car's position, v is the car's velocity, and θ is the car's heading. For the full model description, see [7].

$$X(t+1) = f(X(t), \text{NN}(Y(t))), \; X(0) = X_0 \tag{12a}$$

$$Y(t) = g(X(t), W(t)), \tag{12b}$$

7.1.2. Perturbed system

A common problem with LiDAR is that the rays can bounce off reflective surfaces, causing the sensor to give spuriously large distance readings. To address this issue, we created two more LiDAR models, one with randomly dropped LiDAR rays (denoted by g_d) and one where we learn the observation model from data using a generative adversarial network (GAN) (denoted by g_{gan}). For the dropped ray model, at the start of each new trace we randomly select 5 LiDAR rays to get dropped (thus modeling random reflections) for the entire trace, as opposed to the noisy ground truth distance. This is a crude model of how rays are reflected, since in the real world each LiDAR ray can be reflected and recovered over the course of a trace and the environmental conditions affect when rays get dropped.

Alternatively, the GAN model aims to capture LiDAR artifacts that are observed in a specific environment. To train the GAN model, we used real data collected in our prior work [7]. Specifically, the GAN takes the car's (estimated) state as input and outputs a distribution of the LiDAR rays observed from that position. We used the Wasserstein GAN training algorithm [65,66] with a deconvoluational NN architecture [67]. In Section 7.1.4 we perform a quantitative analysis of the GAN.

7.1.3. Controller training

Next, we used the system in (12) to train 6 different NN controllers for navigating the car through the hallway, taking LiDAR rays as input and giving steering commands as output. To explore the effect of different RL algorithms and different architectures, three of the NNs were trained using the Deep Deterministic Policy Gradient (DDPG) algorithm [68] and the other three were trained using the Twin Delayed DDPG (TD3) algorithm [69]. The reward function is borrowed from [7]; it includes a large penalty for the car hitting the wall to enforce learning safe controllers. Each NN consists of two layers with tanh activations with either 64 neurons each (denoted by 64×64) or 128 neurons each (denoted by 128×128). In summation, we used DDPG to train two 64×64 NNs and one 128×128 NN and we used TD3 to train two 64×64 NNs and one 128×128 NN. Note that using the same architecture and training algorithm leads to different NNs due to randomness in the training procedure.

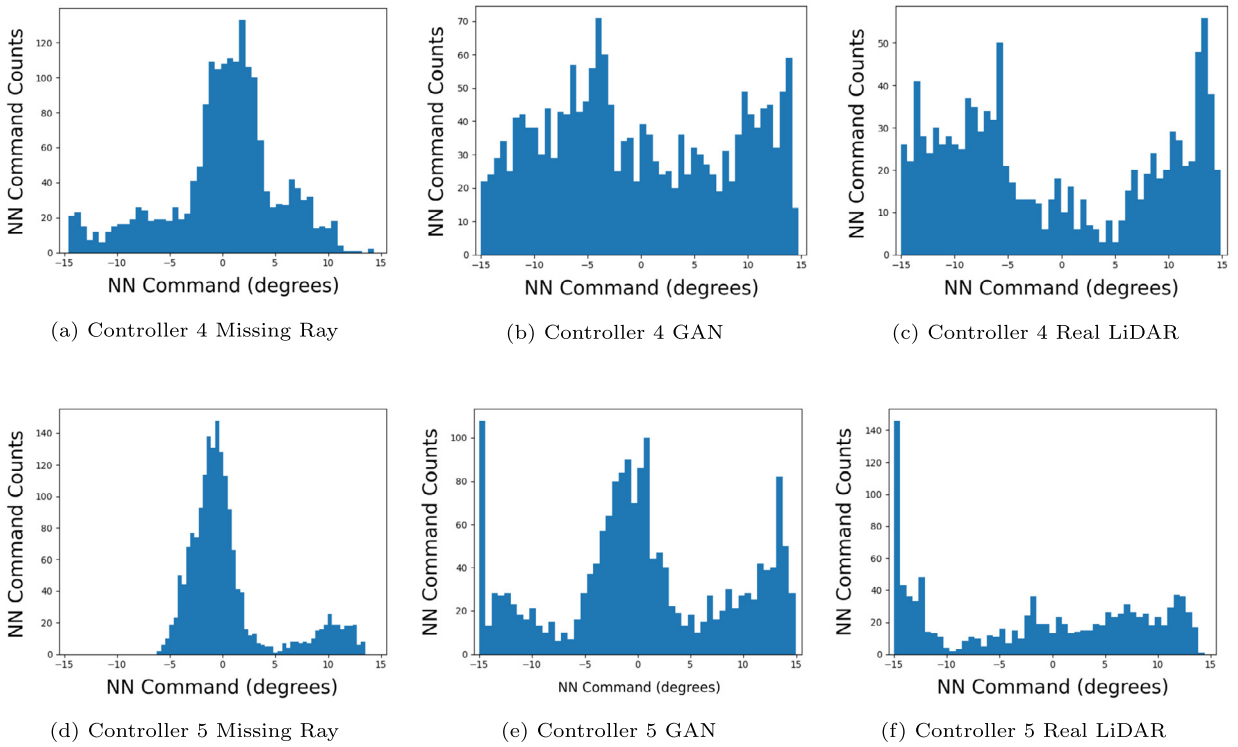
7.1.4. GAN evaluation

To determine how realistic the simulated F1/10 car with the GAN LiDAR model is, we are interested in experimentally determining the quality of the GAN that we trained to emulate real LiDAR readings. A common method for assessing the quality of GANs is through inspecting the behavior of whatever component takes the GAN output as input. For example, the inception score (IS) metric grades image generating GANs by running the GAN images through an image classifier and examining the difference between the classifier's accuracy on real versus generated images [70]. These indirect methods are popular because it is difficult to directly quantify what a good (high-dimensional) GAN output should look like. Thus, we compare the control commands issued by the controllers under the three different LiDAR models with the control commands issued when using real LiDAR data as inputs to the controllers.

An additional challenge in our case is that we would like the GAN to produce realistic data for each system state. Since we do not have ground truth states for the real F1/10 data, we compare the overall distributions of the controller commands in the four different settings (simulated, missing ray, GAN, and real Lidar data). Specifically, we use the initial straight portion of the hallway, since that part of the hallway should give similar LiDAR readings and requires similar control commands to navigate. Although we do not have ground truth positions for the real F1/10 runs, the first half of each (seven-second)

Table 1
Wasserstein distances of control commands between different LiDAR models and real LiDAR data. The best (lowest) value in each row has been bolded.

Controller name	Controller type	Simulated LiDAR to real LiDAR	Missing ray LiDAR to real LiDAR	GAN LiDAR to real LiDAR
1	DDPG 64×64	3.5173	3.4613	**1.0903**
2	TD3 64×64	7.1850	7.4398	**2.2854**
3	DDPG 128×128	2.0057	1.9783	**0.9140**
4	DDPG 64×64	5.6847	5.4686	**1.7944**
5	TD3 128×128	5.7271	6.1513	**2.3520**
6	TD3 64×64	7.5859	7.8669	**1.2940**



(a) Controller 4 Missing Ray (b) Controller 4 GAN (c) Controller 4 Real LiDAR

(d) Controller 5 Missing Ray (e) Controller 5 GAN (f) Controller 5 Real LiDAR

Fig. 4. Histograms showing control command distributions for the dropped ray LiDAR model, GAN LiDAR model, and real LiDAR data for various controllers.

run roughly corresponds to the straight segment before the turn. Thus, we conservatively use only the first three seconds of LiDAR readings for each run of the real F1/10 car, in order to compute the control input distributions in the straight segment.

First, we compute the Wasserstein distance between the distributions of control commands for the three types of LiDAR models with the distribution of the control commands on the real LiDAR data, which is shown in Table 1. The Wasserstein distance is a standard metric for the distance between distributions – intuitively, it measures the probability mass that needs to be moved in order to convert one distribution into a another [71]. Across all six controllers, the GAN LiDAR model controls have a considerably smaller Wasserstein distance to the real LiDAR controls than either of the other two LiDAR models do. This is evidence that the GAN does indeed do a reasonable job of approximating the real LiDAR data. Next, we plot histograms of the control commands for the GAN LiDAR and real LiDARs for all six controllers. The histograms of controllers 4 and 5 are in Fig. 4 and the histograms of the other 4 controllers are in Appendix K. The histograms show similar patterns of control commands for most of the controllers. For example, controller 4 shows high levels of actions on $[10, 15]$ and $[-10, -15]$ and few actions from $[-3, 5]$ for both the GAN and real LiDAR. However, the GAN and real LiDAR histograms for controller 5 are quite different from one another, which demonstrates that the GAN is not a perfect approximation of the real LiDAR.

These results demonstrate that the GAN results in a closer approximation of the real LiDAR than either of the other two LiDAR models we considered. This opens up future research directions of using GANs to make simulations more realistic.

Table 2
F1/10 VaR values for different LiDAR models under the specification given in (11).

Controller name	Controller type	Simulated LiDAR	Dropped ray LiDAR	GAN LiDAR
1	DDPG 64 × 64	−**0.0907**	0.1113	0.1222
2	TD3 64 × 64	−0.0589	0.0840	0.0853
3	DDPG 128 × 128	−0.0366	0.1649	0.1188
4	DDPG 64 × 64	−0.0905	**0.0263**	0.1060
5	TD3 128 × 128	−0.0668	0.0413	0.0801
6	TD3 64 × 64	−0.0498	0.0300	**0.0545**

Table 3
F1/10 CVaR values for different LiDAR models under the specification given in (11).

Controller name	Controller type	Simulated LiDAR	Dropped ray LiDAR	GAN LiDAR
1	DDPG 64 × 64	−**0.0802**	0.1518	0.1666
2	TD3 64 × 64	−0.0436	0.1259	0.1213
3	DDPG 128 × 128	−0.0303	0.2008	0.1840
4	DDPG 64 × 64	−0.0791	0.1064	0.1521
5	TD3 128 × 128	−0.0562	0.0945	0.1059
6	TD3 64 × 64	−0.0435	**0.0782**	**0.1032**

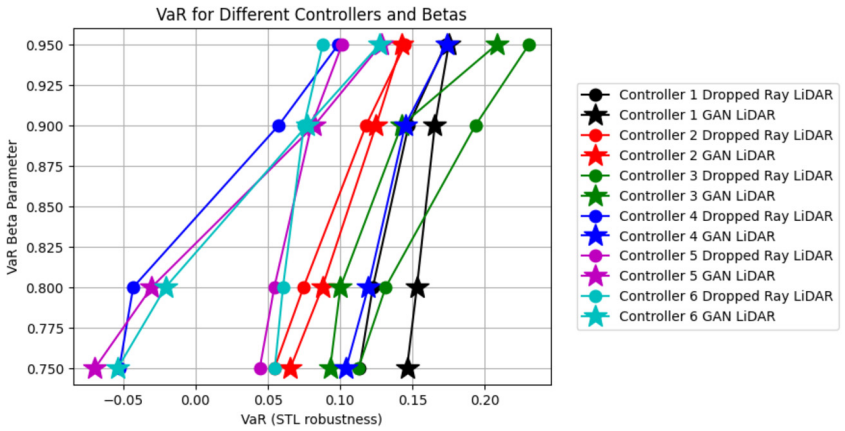


Fig. 5. Value at Risks (VaRs) for the F1/10 car under the specification given in (11) for the dropped ray and GAN LiDAR models and various controllers.

7.1.5. Experiments

We are now interested in computing the robustness risk of the various NN controlled F1/10 systems satisfying the specification in (11). To do so, we first generate 50000 traces for each NN and for each of the three LiDAR models. We simulate the right-hand turn scenario used in prior work [7]. From these data, we compute the VaR and CVaR risks for each of the controllers using a single risk level of $\beta := 0.9$ and confidence of $1 - \delta := 0.95$, which are shown in Tables 2 and 3. The least risky controller is indicated in bold font.

First, note that all controllers perform best on the simulated LiDAR model. In fact, the risk values are all negative on the simulated LiDAR model. This indicates that the controllers are all generally safe for the simulated LiDAR model, as negative risk robustness values indicate that the system is safe, while positive ones indicate that the system is unsafe. This makes sense as that is the model on which the controllers were trained and is consistent with the risk verification gap theory. We also see that the controllers perform better on the dropped ray LiDAR model than the GAN LiDAR model. That is because the dropped ray LiDAR model is more similar to the simulated LiDAR model than the GAN model is. Note that this is consistent with the risk verification gap theory presented in Section 5.

Note next that there is little correlation between which controllers perform well on the simulated LiDAR model and which perform well on the dropped ray and GAN LiDAR models. This highlights the potential conservatism of the risk verification gap theory as per Corollary 1, which is that if the nominal model is quite different from the perturbed model, then the bound on the risk gap Δ may be large (this is analyzed in more detail in the next section). We remark that this discrepancy is smaller for the second case study. However, controllers that perform well on the dropped ray model appear to also perform well on the GAN LiDAR model.

Next, we varied the β values for the VaR and CVaR metrics. These β values control how much or little of the tail gets considered in the risk. Larger values only consider the extremes of the tails. Figs. 5 and 6 show the different risk values for

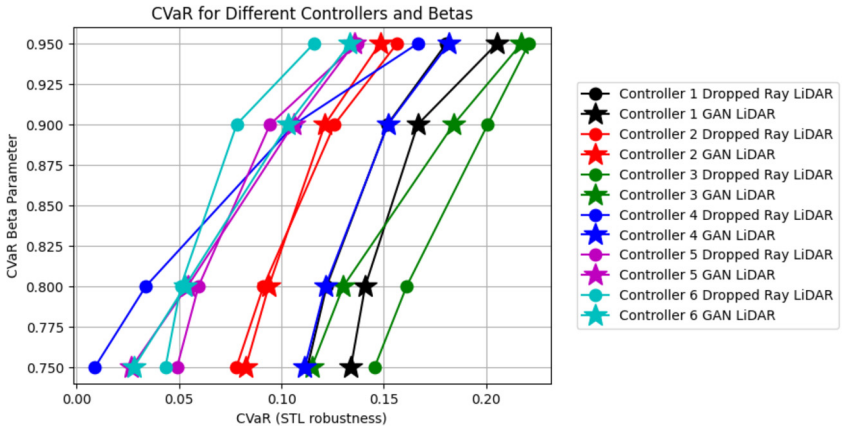
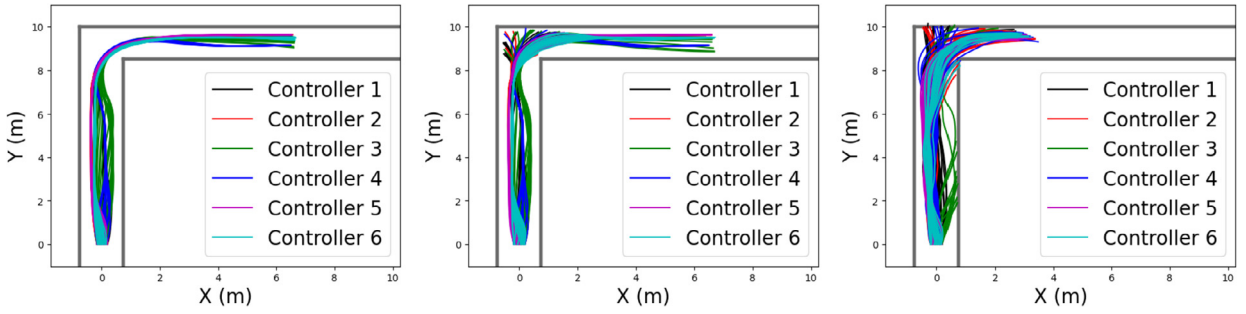


Fig. 6. Conditional Value at Risks (CVaRs) for the F1/10 car under the specification given in (11) for the dropped ray and GAN LiDAR models and various controllers.



(a) Simulated LiDAR traces (b) Missing ray LiDAR traces (c) GAN LiDAR traces

Fig. 7. Traces for different LiDAR models and controllers.

various controllers for both the dropped ray and GAN LiDAR models. Note that we still observe that the controllers which perform better on the dropped ray LiDAR also perform well on the GAN LiDAR. In particular, for the dropped ray LiDAR Controller 4 shows very good CVaR values for smaller β values, but as β increases the performances becomes dominated by other controllers. Controller 4 then performs much worse than those controllers on the GAN data. This illustrates the benefit of using risk metrics analyzing systems in simulation. The poor performance of Controller 4 at the extreme tails of its robustness distribution properly indicates that it would not generalize well to the GAN LiDAR model.

Finally, we plot 20 traces at random for each controller for both the missing ray and GAN LiDAR models. These traces can be seen in Fig. 7. As expected, controller 6 crashes in the walls the least. However, note that the controllers are in general unsafe, especially for the GAN LiDAR model. This is reflected in the robustness risks being positive in Tables 4 and 5, i.e., using robustness risk can indicate the quality of safety.

7.1.6. Trace difference risk bounds

In this section, we use the results from Theorems 1 and 2 to bound the robustness risks of the GAN LiDAR model based on the robustness risks of the dropped ray LiDAR models and the trace differences between the two models. We can define the trace difference bound $\Delta(\omega)$ as

$$\Delta(\omega) := \sup_{t \in \mathbb{T}} \|\overline{X}(t, \omega) - X(t, \omega)\| \tag{13}$$

where X denotes the dropped ray LiDAR model and $\overline{X}$ denotes the GAN LiDAR model. The goal of these experiments is to experimentally determine bounds on (13) to use and verify Theorems 1 and 2.

To do this, we run both the dropped LiDAR and GAN LiDAR models from the same random initial conditions for 5000 trials to get a distribution of trace differences, which are shown as histograms in Fig. 8. Then we compute the supremum, VaR, and CVaR of these trace differences, which we denote as $\sup_{\omega \in \Omega} \Delta(\omega)$, $VaR_\beta(\Delta(\omega))$, and $CVaR_\beta(\Delta(\omega))$. We refer to these values as the trace difference risks. Finally, we use Theorems 1 and 2 to combine these trace difference risks with the dropped ray LiDAR model robustness risks to determine upper bounds on the GAN LiDAR model robustness risks. These bounds, along with the trace difference risks, are shown in Tables 4 and 5. The GAN risk bounds are all larger than the empirical GAN robustness risks we computed in Tables 2 and 3, which shows that the bounds are sound. In addition, the

Table 4
GAN LiDAR VaR bounds estimated from trace bounds and the dropped ray LiDAR VaR bounds.

Controller name	Dropped ray LiDAR VaR	$\sup_{\omega\in\Omega}(\Delta(\omega))$	$VaR(\Delta(\omega))$	GAN LiDAR VaR bound Theorem 1	GAN LiDAR VaR bound Theorem 2
1	0.1113	**2.5671**	1.9074	**2.6784**	2.0187
2	0.0840	2.8430	1.7495	2.927	1.8335
3	0.1649	2.7444	1.4020	2.9093	1.5669
4	**0.0263**	2.9291	1.5601	2.9554	1.5864
5	0.0413	2.7183	0.8942	2.7596	0.9355
6	0.0300	2.8128	**0.7815**	2.8428	**0.8115**

Table 5
GAN LiDAR CVaR bounds estimated from trace bounds and the dropped ray LiDAR CVaR bounds.

Controller name	Dropped ray LiDAR CVaR	$\sup_{\omega\in\Omega}(\Delta(\omega))$	$CVaR(\Delta(\omega))$	GAN LiDAR CVaR bound Theorem 1	GAN LiDAR CVaR bound Theorem 2
1	0.1518	**2.5671**	2.2008	**2.7189**	2.3526
2	0.1259	2.8430	2.4045	2.9689	2.5304
3	0.2008	2.7444	1.8608	2.9452	2.0616
4	0.1064	2.9291	2.1220	3.0355	2.2284
5	0.0945	2.7183	1.6597	2.8128	1.7542
6	**0.0782**	2.8128	**1.5475**	2.8910	**1.6257**

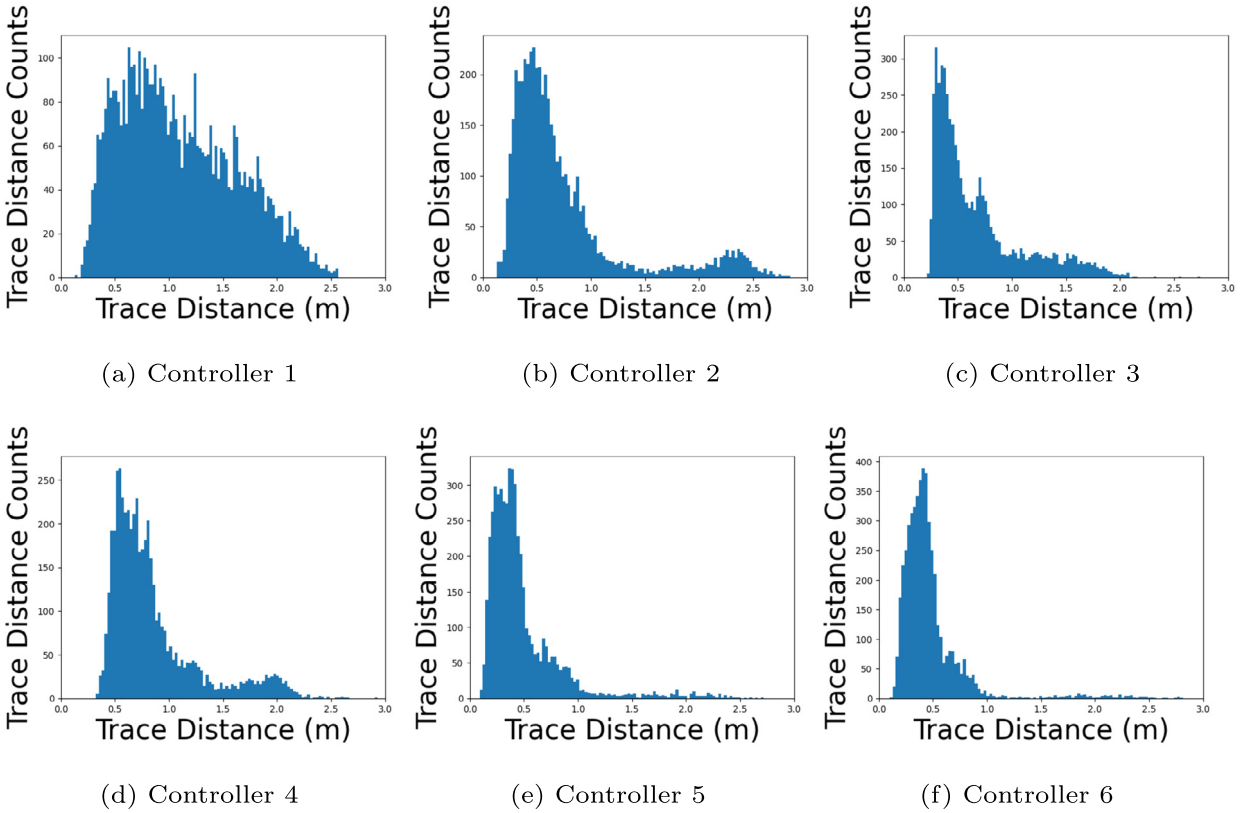


Fig. 8. Trace Difference histograms between the dropped ray and GAN LiDAR models for various controllers.

relative values of the Theorem 2 bounds match those of the empirical GAN robustness risks from Tables 2 and 3. In fact, the two controllers with the best estimated GAN robustness risks, controllers 5 and 6 (see Tables 2 and 3), also have the best robustness risks bounds when using Theorem 2. This indicates that the bound in Theorem 2 is useful in practice. One drawback is that the bounds are a bit larger than the empirical robustness risks, which indicates that the bounds can be conservative. On the other hand, the bounds from using Theorem 1 have less of a correlation with the relative robustness

risks of the controllers. This highlights the benefit of treating the trace differences as random variables, as opposed to simply considering their upper bounds.

7.2. UUV

Recall the UUV system described in Section 3, where a UUV is tasked with tracking a pipeline on the seafloor using a side scan sonar and an NN controller. As the sonar works best in a certain range of distances from the pipeline, we consider the following requirement in STL that expresses expeditiousness:

$$\begin{aligned}\Phi_{UUV} :=& G\left[d_{uuv} < d_l \implies F_{[0,t_{uuv}]}\left(d_{uuv} \geq d_l\right)\right] \wedge \\ & G\left[d_{uuv} > d_u \implies F_{[0,t_{uuv}]}\left(d_{uuv} \leq d_u\right)\right]\end{aligned} \tag{14}$$

where d is the UUV's distance to the pipeline in the plane, d_u and d_l are the upper and lower distances to the pipe which the UUV must lie within and t_{uuv} is the amount of time in which the UUV is allowed to leave this distance range. Intuitively, this STL formula states that if the UUV leaves the ideal range of distances to the pipeline, $[d_l, d_u]$, then it must return to that ideal range within t_{uuv} seconds. This ensures that the side scan sonar will be able to clearly track the pipeline while allowing the UUV to briefly lose the pipeline if needed, e.g. to avoid an obstacle.

7.2.1. Nominal system (linearized model)

Recall that we consider a high-fidelity UUV simulator based on the ROS-Gazebo real-time physics engine presented in [53]. As mentioned in Section 3, executing the simulator requires significant computation resources, which makes it challenging to train a controller using RL. To overcome this issue, we use a linear model from the simulator and treat that model as the nominal system.

The linearized model of the UUV is available online at https://github.com/Verisig/verisig. In summary the linearization process is as follows. Using data from the real-time physics engine, one can identify a 4-th order linearized model of the UUV, which takes as input heading, speed, and depth commands and outputs the actual heading, speed, and depth of the UUV. Note that the full UUV model is still not linear since the linearized model does not capture global position. One can use a kinematic model, based on the speed and heading, in order to model the UUV position dynamics.

The equations in (15) show the dynamics of the linearized model of the UUV, where f_{lin} represents the linearized dynamics. The model states are: $X = \begin{bmatrix} x_{uuv} & y_{uuv} & \theta_{uuv} & v_{uuv} & d_{uuv} \end{bmatrix}^T$, which contains the 2-D position, heading, speed and depth of the UUV, respectively. The measurements are the UUV's heading and distance with respect to the pipeline, i.e., $Y = \begin{bmatrix} \theta_{uuv} & d_{uuv} \end{bmatrix}^T$, with the addition of additive Gaussian noise $W(t)$. Finally, the NN controller takes the measurements as input and outputs a desired heading

$$X(t+1) = f_{lin}(X(t), \mathrm{NN}(Y(t))),\ X(0) = X_0 \tag{15a}$$

$$Y(t) = g(X(t), W(t)). \tag{15b}$$

7.2.2. Perturbed system (high-fidelity simulator)

As mentioned above, we consider the high-fidelity UUV physics simulator to be the (real) perturbed system in this case study. In this case, the dynamics f and state $\overline{X}(t)$ are internal to the real-time physics engine and thus can be considered unknown, as shown in (16). Moreover, $\overline{Y}(t)$ and NN are the same as their counterparts in (15).

$$\overline{X}(t+1) = f(\overline{X}(t), \mathrm{NN}(\overline{Y}(t))),\ \overline{X}(0) = X_0 \tag{16a}$$

$$\overline{Y}(t) = g(\overline{X}(t)). \tag{16b}$$

7.2.3. Controller training

Next, we used the linearized system in (15) to train NN controllers to satisfy the objective in (14) via RL. In particular, we used the Twin Delayed Deep Deterministic Policy Gradients (TD3) algorithm [69] to train 4 NN controllers for guiding the UUV alongside a pipeline on the seafloor. Each NN consists of 2 tanh layers, each of width 32. The reward function r is as follows:

$$r(X(t), \mathrm{NN}(Y(t))) = 0.5 - 100\, \mathbb{1}_{d_{uuv} \notin [d_1, d_2]} - 0.5\, \mathrm{NN}(Y(t)) - 0.1 \left| d_{uuv} - \frac{d_2 - d_1}{2} \right| \tag{17}$$

where $[d_1, d_2]$ is the safe interval which gets altered when training each of the NNs and $X(t)$, $Y(t)$, $\mathrm{NN}(\cdot)$ and d_{uuv} are as in Section 7.2.1.

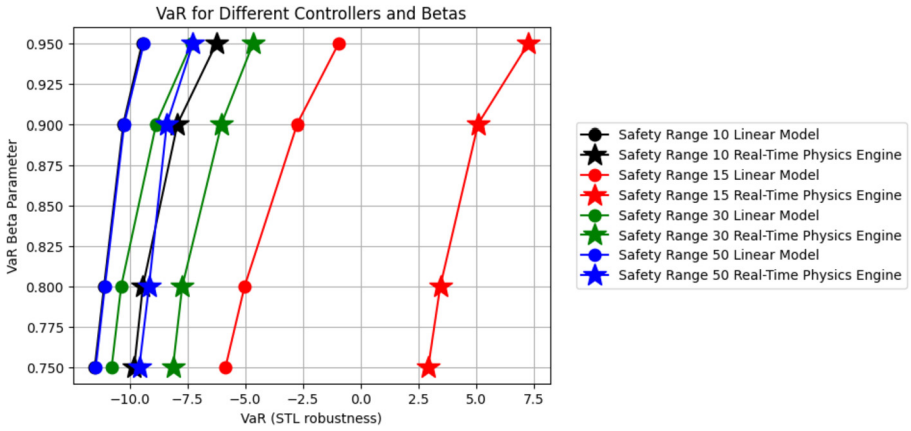


Fig. 9. Value-at-Risks (VaRs) for the STL robustness under the specification in (14) of the 4 different controllers on both UUV models for varied β parameters.

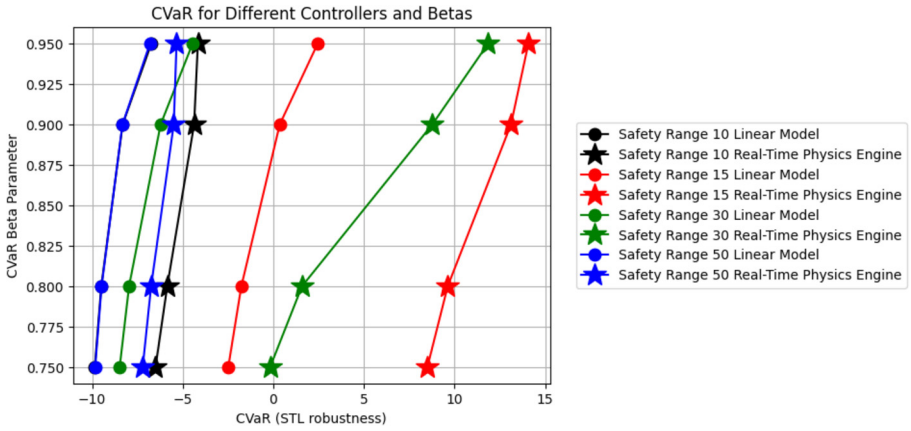


Fig. 10. Conditional Value-at-Risks (CVaRs) for the STL robustness under the specification in (14) of the 4 different controllers on both UUV models for varied β parameters.

7.2.4. UUV experiments

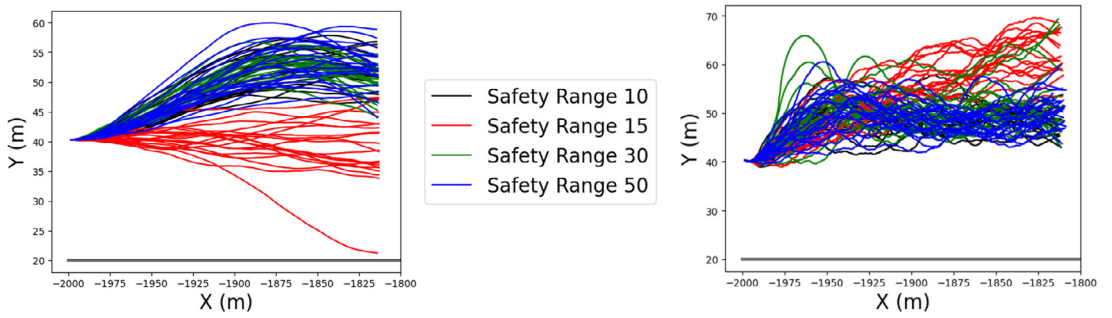
To verify the risk of each of these NN controllers, each of them was used to generate 5000 runs of the linearized model and 300 runs of the real-time physics engine, each corresponding to 3 minutes of data per run. Finally, the STL robustness risk of the STL specification in (14) is estimated as explained in Section 4.1 for the VaR and the CVaR and for a risk level of $\beta := 0.9$ and a confidence of $1 - \delta := 0.95$. The results are shown in Table 6. The controllers which exhibit higher risk on the linearized model also exhibit higher risk on the real-time physics engine for both VaR and CVaR, supporting the risk verification gap theory from Section 5. In particular, the safety range 10 and 50 controllers have the lowest risks on the linearized model and on the real-time physics engine, while the safety range 15 controller has high risk both the linearized model and the real-time physics engine. This offers further empirical evidence that systems which exhibit lower robustness risk in simulation will also exhibit lower robustness risk in the real world.

Next, we varied the risk level β for the VaR and CVaR metrics. These β values control how much or little of the tail gets considered in the risk. Larger values only consider the extremes of the tails. Figs. 9 and 10 show the VaR and CVaR values, respectively, for both the linearized model and real-time physics engine and different controllers for the specification given in (14). These figures both show that controllers which exhibit low risk values on the linearized model also exhibit low risk values on the real-time physics engine and vice versa, and are hence in line with the results in Section 5.

Finally we plot 20 traces at random for each controller for both the linear simulator and real-time physics engine, which can be seen in Fig. 11. As expected, the safety range 15 controller paths get the farthest from the $[d_l, d_u]$ interval specified in the STL specification (14), whereas the safety range 10 and 50 controllers stay most closely within the prescribed range. This matches the risks from Table 6.

7.2.5. Trace difference risk bounds

In this section, we use the results from Theorems 1 and 2 to bound the robustness risks of the real-time physics engine based on the robustness risks of the linearized model and the trace differences between the two models. We can define the trace difference bound $\Delta(\omega)$ as



(a) UUV linear simulator traces

(b) UUV real-time physics engine traces

Fig. 11. Traces for different UUV models and controllers.

Table 6
UUV risks for a real-time physics engine and a linearized model for specification (14).

Safety range	Linearized model VaR	Real-time physics engine VaR	Linearized model CVaR	Real-time physics engine CVaR
10	**−10.1215**	−7.9380	−7.0261	−4.3691
15	−1.8944	5.0788	2.3087	13.1208
30	−8.2857	−6.0316	−5.2708	8.7808
50	−10.0697	**−8.3995**	**−7.3678**	**−5.4930**

$$\Delta(\omega) := \sup_{t \in \mathbb{T}} \|\overline{X}(t,\omega) - X(t,\omega)\| \tag{18}$$

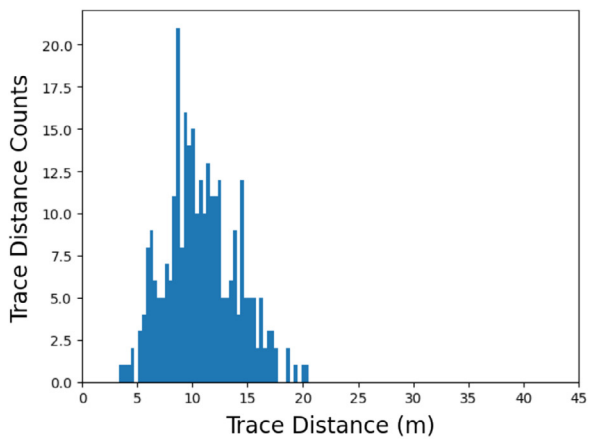
where X denotes the linearized model and $\overline{X}$ denotes the real-time physics engine. The goal of these experiments is to experimentally determine bounds on (18) to use and verify Theorems 1 and 2.

To do this, we run both the linearized model and real-time physics engine from the same random initial conditions for 300 trials to get a distribution of trace differences, which are shown as histograms in Fig. 12. Then we compute the supremum, VaR, and CVaR of these trace differences, which we denote as $\sup_{\omega\in\Omega} \Delta(\omega)$, $VaR_\beta(\Delta(\omega))$, and $CVaR_\beta(\Delta(\omega))$. We refer to these values as the trace difference risks. Finally, we use Theorems 1 and 2 to combine these trace difference risks with the linearized model robustness risks to determine upper bounds on the real-time physics engine robustness risks. These bounds, along with the trace difference risks, are shown in Tables 7 and 8. The real-time physics engine risk bounds are all larger than the empirical real-time physics engine risks we computed in Table 6, which shows that the bounds are sound. In addition, the relative values of the real-time physics engine risk bounds match those of the empirical real-time physics engine risks. Specifically, the safety range 10 and 50 controllers offer the best robustness risk values and bounds for both Theorems 1 and 2, while the safety range 15 controllers have a much larger robustness risk value and bound than the other controllers do. This indicates that our risk bounds in Theorems 1 and 2 can aid in determining which controllers will best transfer from a nominal to a perturbed system. One drawback is that the bounds are a bit larger than the empirical robustness risks, which indicates that the bounds can be conservative. Note that the application of (7) and Theorem 2 requires generating samples from the linearized model and real-time physics engine at the same initial conditions, which may not always be possible in the sim2real setting.

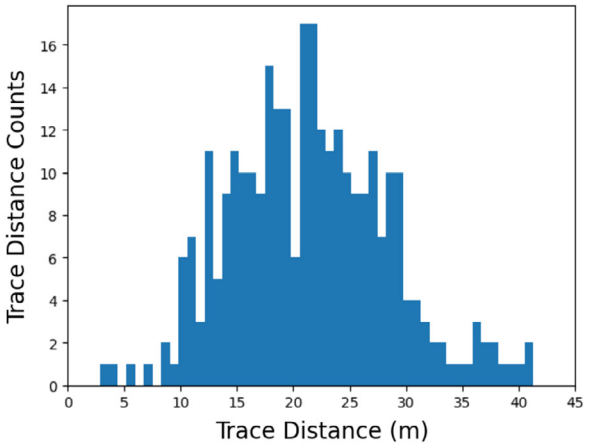
8. Conclusions

In this paper, we have presented a framework for the closed-loop risk verification of stochastic systems with NN controllers. Specifically, our framework is based on risk metrics and distributions of trace robustness that are defined with respect to the violation of the system specification. We particularly allow for the use of formal specification languages such as signal temporal logic specifications. It was shown how such risk can be reliably estimated from finite data. Additionally, we bound the changes in risk when the system experiences disturbances for both general and iISS systems. We note that iISS systems have much tighter risk bounds. These bounds are conservative for the general system or require the strong iISS property, so we also provide an empirical method to determine the change in risk based on differences of sampled traces. Finally, our framework was applied to the risk verification and empirical risk gap estimation of an F1/10 race car and an underwater vehicle controlled by end-to-end NN controllers. Future research directions include applying the risk verification framework to NNs in an open loop or standalone setting, using the framework to guide the learning process of NN controllers, and extending the risk gap quantification to the sim2real gap.

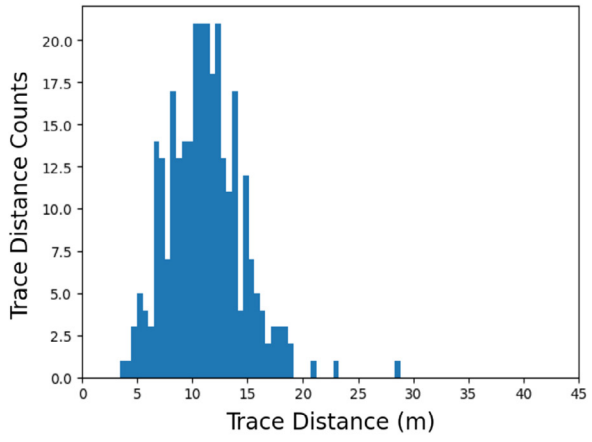
We believe that this work makes three important contributions for the analysis of artificial intelligent systems. First, risk estimation is an essential tool for the verification of safety-critical systems with NN components. Notably, our method can



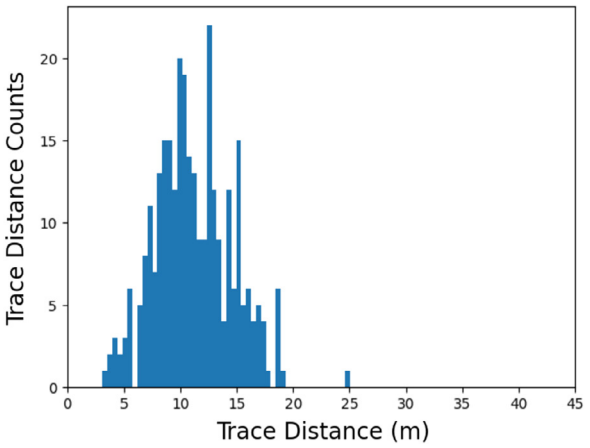
(a) Safety Range 10



(b) Safety Range 15



(c) Safety Range 30



(d) Safety Range 50

Fig. 12. Trace Difference histograms between the linearized model and real-time physics engine for various controllers.

Table 7
Real-time physics engine VaR bounds estimated from trace bounds and the linearized model VaR bounds.

Safety range	Linearized model VaR	$\sup_{\omega\in\Omega}(\Delta(\omega))$	$VaR(\Delta(\omega))$	Real-time physics engine VaR bound Theorem 1	Real-time physics engine VaR bound Theorem 2
10	$-$**10.1215**	**20.5451**	**15.3278**	**10.4236**	**5.2063**
15	$-$1.8944	41.2857	29.9825	39.3913	28.0881
30	$-$8.2857	28.8160	15.6049	20.5303	7.3192
50	$-$10.0697	25.0174	15.7795	14.9477	5.7098

Table 8
Real-time physics engine CVaR bounds estimated from trace bounds and the linearized model CVaR bounds.

Safety range	Linearized model VaR	$\sup_{\omega\in\Omega}(\Delta(\omega))$	$CVaR(\Delta(\omega))$	Real-time physics engine CVaR bound Theorem 1	Real-time physics engine CVaR bound Theorem 2
10	$-$7.0261	**20.5451**	**20.0697**	**13.5190**	**13.0436**
15	2.3087	41.2857	40.7459	43.5944	43.0546
30	$-$5.2708	28.8160	26.7425	23.5452	21.4717
50	$-$**7.3678**	25.0174	23.5814	17.6496	16.2136

directly verify closed-loop stochastic systems with NN controllers, whereas current model-based verification techniques cannot. Second, NN controllers are complex and their closed-loop verification imposes significant computational costs, whereas our data-driven method trades off this computational complexity with the ability to collect data. Lastly, the quantification of the risk verification gap allows us to quantify the worst case risk of a perturbed version of a nominal system. We further believe that the information gained from risk verification may in the future guide the design process and close the loop between training and verification of NN controllers.

CRediT authorship contribution statement

Matthew Cleaveland: Conceptualization, Formal analysis, Methodology, Software, Writing – original draft. **Lars Lindemann:** Conceptualization, Methodology, Writing – original draft. **Radoslav Ivanov:** Software, Writing – review & editing. **George J. Pappas:** Funding acquisition, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research was generously supported by AFRL grant FA8750-18-C-0090, DARPA grant FA8750-18-C-0090, AFOSR grant FA9550-19-1-0265, and NSF award CPS-2038873.

Appendix A. Risk metrics

We next summarize some desirable properties of a risk metric. Let therefore $Z, Z' \in \mathfrak{F}(\Omega, \mathbb{R})$ be cost random variables. A risk metric is *coherent* if the following four properties are satisfied.

1. Monotonicity: If $Z(\omega) \leq Z'(\omega)$ for all $\omega \in \Omega$, it holds that $R(Z) \leq R(Z')$.

2. Translation invariance: Let $c \in \mathbb{R}$. It holds that $R(Z + c) = R(Z) + c$.

3. Positive homogeneity: Let $c \in \mathbb{R}_{\geq 0}$. It holds that $R(cZ) = R(Z)$.

4. Subadditivity: It holds that $R(Z + Z') \leq R(Z) + R(Z')$.

If the risk metric additionally satisfies the following two properties, then it is called a distortion risk metric.

5. Comonotone additivity: If $(Z(\omega) - Z(\omega'))(Z'(\omega) - Z'(\omega')) \geq 0$ for all $\omega, \omega' \in \Omega$ (namely, Z and Z' are commotone), it holds that $R(Z + Z') = R(Z) + R(Z')$.

6. Law invariance: If Z and Z' are identically distributed, then $R(Z) = R(Z')$.

Common examples of popular risk metrics are the expected value $\mathrm{E}(Z)$ (risk neutral) and the worst-case $\operatorname{ess\,sup}_{\omega\in\Omega} Z(\omega)$ as well as:

- Mean-Variance: $\mathrm{E}(Z) + \lambda \mathrm{Var}(Z)$ where $c > 0$.
- Value at Risk (VaR) at level $\beta \in (0, 1)$: $VaR_\beta(Z) := \inf\{\alpha \in \mathbb{R} | F_Z(\alpha) \geq \beta\}$.
- Conditional Value at Risk (CVaR) at level $\beta \in (0, 1)$: $CVaR_\beta(Z) := E(Z | Z > VaR_\beta(Z))$.

Many risk metrics are not coherent and can lead to a misjudgement of risk, e.g., the mean-variance is not monotone and the value at risk (which is closely related to chance constraints as often used in optimization) does not satisfy the subadditivity property.

Appendix B. Measurability of $-\rho^c(X)$

For a constraint function $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$, we want to show that the robustness function $\rho^c(X(\cdot, \omega))$ is measureable in ω. In this case, $\rho^c(X)$ and hence $-\rho^c(X)$ are random variables so that $R(-\rho^c(X))$ is well defined. Recall that $\rho^c(X(\cdot, \omega)) = \inf_{t\in\mathbb{T}} \mathrm{Dist}^c(X(t, \omega), t)$ where the signed distance is

$$\mathrm{Dist}^c(X(t,\omega),t) = \begin{cases} \inf_{x' \in \mathrm{cl}(O^{\neg c}(t))} \|X(t,\omega) - x'\| & \text{if } c(X(t,\omega),t) \geq 0 \\ -\inf_{x' \in \mathrm{cl}(O^{c}(t))} \|X(t,\omega) - x'\| & \text{otherwise.} \end{cases}$$

We first show that $\mathrm{Dist}^c(X(t, \omega))$ is measurable in ω for a fixed $t \in \mathbb{T}$. Note therefore that: 1) the function $c(X(t, \omega))$ is measurable in ω for a fixed $t \in \mathbb{T}$, 2) the indicator function of a measurable set is measurable, and 3) the distance function to a non-empty set, i.e., $\inf_{x' \in \mathrm{cl}(O^{\neg c}(t))} \|X(t, \omega) - x'\|$, is continuous (see e.g., [72, Chapter 3]) and hence measurable. Consequently, $\mathrm{Dist}^c(X(t, \omega))$ is measurable. To now conclude measurability of $\rho^c(X(\cdot, \omega))$ in ω, note that the infimum over a countable number of measurable functions is again measurable, see e.g., [73, Theorem 4.27].

Appendix C. Proof of Lemma 1

For a constraint function $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$, we want to show that $\rho^c(X(\cdot,\omega)) - \Delta \leq \rho^c(\overline{X}(\cdot,\omega))$ for each realization $\omega \in \Omega$. For convenience, let $\overline{x} := \overline{X}(\cdot,\omega)$ and $x := X(\cdot,\omega)$ and recall that the robustness function $\rho^c(x)$ for the constraint function c is defined as $\rho^c(x) = \inf_{t\in\mathbb{T}} \mathrm{Dist}^c(x(t),t)$ where the signed distance is

$$\mathrm{Dist}^c(x(t),t) = \begin{cases} \inf_{x' \in \mathrm{cl}(O^{\neg c}(t))} \|x(t) - x'\| & \text{if } c(x(t),t) \geq 0 \\ -\inf_{x' \in \mathrm{cl}(O^{c}(t))} \|x(t) - x'\| & \text{otherwise.} \end{cases}$$

Note next that it holds that $\inf_{x' \in \mathrm{cl}(O^{\neg c}(t))} \|x(t) - x'\|$ and $\inf_{x' \in \mathrm{cl}(O^{c}(t))} \|x(t) - x'\|$ are Lipschitz continuous functions with Lipschitz constant one, see for instance [72, Chapter 3]. Consequently, it holds that $\mathrm{Dist}^c(x(t),t)$ is Lipschitz continuous in its first argument with Lipschitz constant one. From this, we have that $|\mathrm{Dist}^c(x(t),t) - \mathrm{Dist}^c(\overline{x}(t),t)| \leq \|x(t) - \overline{x}(t)\| \leq \Delta$ for all times $t \in \mathbb{T}$. By the definition of ρ^c, we consequently have that $|\rho^c(x) - \rho^c(\overline{x})| = |\inf_{t\in\mathbb{T}} \mathrm{Dist}^c(x(t),t) - \inf_{t\in\mathbb{T}} \mathrm{Dist}^c(\overline{x}(t),t)| \leq \Delta$ so that $\rho^c(\overline{x}) \geq \rho^c(x) - \Delta$.

Appendix D. Proof of Theorem 1

For a constraint function $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$, we want to show that $R(-\rho^c(\overline{X})) \leq R(-\rho^c(X)) + \Delta$. By Lemma 1 and since the risk metric R is monotone (see Appendix A), it follows that $R(-\rho^c(\overline{X})) \leq R(-\rho^c(X) + \Delta)$. Since R is additionally translationally invariant (see Appendix A), it further holds that $R(-\rho^c(X) + \Delta) = R(-\rho^c(X)) + \Delta$ so that $R(-\rho^c(\overline{X})) \leq R(-\rho^c(X)) + \Delta$, which concludes the proof.

Appendix E. Proof of Theorem 2

For a constraint function $c : \mathbb{R}^n \times \mathbb{T} \to \mathbb{R}$, we want to show that $R(-\rho^c(\overline{X})) \leq R(-\rho^c(X)) + R(\Gamma)$. By minor modification of Lemma 1, one can show that $\rho^c(\overline{X}(\cdot,\omega)) \geq \rho^c(X(\cdot,\omega)) - \Gamma(\omega)$ for each realization $\omega \in \Omega$. Since the risk metric R is monotone (see Appendix A), it follows that $R(-\rho^c(\overline{X})) \leq R(-\rho^c(X) + \Gamma)$. Since R is now assumed to be subadditive (see Appendix A), it further holds that $R(-\rho^c(X) + \Gamma) \leq R(-\rho^c(X)) + R(\Gamma)$ so that $R(-\rho^c(\overline{X})) \leq R(-\rho^c(X)) + R(\Gamma)$, which concludes the proof.

Appendix F. Bound $\Delta(t)$ on $\|\overline{X}(t,\omega) - X(t,\omega)\|$

First, recall that the nominal system has dynamics

$$X(t+1) = f(X(t), u(Y(t)), V(t)),\ X(0) = X_0$$
$$Y(t) = g(X(t), W(t)),$$

while the perturbed system has dynamics

$$\overline{X}(t+1) = f(\overline{X}(t), u(\overline{Y}(t)), \overline{V}(t)),\ \overline{X}(0) = X_0$$
$$\overline{Y}(t) = g(\overline{X}(t), \overline{W}(t)).$$

Additionally, recall that $L_{f,1}$, $L_{f,2}$, and $L_{f,3}$ are the Lipschitz constants of the first, second, and third arguments of f, respectively, $L_{g,1}$ and $L_{g,2}$ are the Lipschitz constants of g, and L_u is the Lipschitz constant of u. Finally, note that we have disturbance bounds $v^* := 2\max_{v\in D_v} \|v\|$ and $w^* := 2\max_{w\in D_w} \|w\|$. Note that initially $\|\overline{X}(0,\omega) - X(0,\omega)\| = 0$ as $\overline{X}(0,\omega) = X_0(\omega)$ and $X(0,\omega) = X_0(\omega)$. We can then upper bound $\|\overline{X}(t+1,\omega) - X(t+1,\omega)\|$ recursively as follows.

$$\begin{aligned}
\|\overline{X}(t+1,\omega) - X(t+1,\omega)\| &\overset{(a)}{=} \|f(\overline{X}(t,\omega), u\left(\overline{Y}(t)\right), \overline{V}(t)) - f(X(t,\omega), u(Y(t)), V(t))\| \\
&\overset{(b)}{=} \|f(\overline{X}(t,\omega), u\left(g(\overline{X}(t,\omega), \overline{W}(t))\right), \overline{V}(t)) \\
&\quad - f(X(t,\omega), u\left(g(X(t,\omega), W(t))\right), V(t))\| \\
&\overset{(c)}{\leq} L_{f,1}\|\overline{X}(t,\omega) - X(t,\omega)\| + L_{f,2}\|u\left(g(\overline{X}(t,\omega), \overline{W}(t))\right) \\
&\quad - u(g(X(t,\omega), W(t)))\| + L_{f,3}v^* \\
&\overset{(d)}{\leq} L_{f,1}\Delta(t) + L_{f,2}L_u\|g(\overline{X}(t,\omega), \overline{W}(t)) \\
&\quad - g(X(t,\omega), W(t))\| + L_{f,3}v^*
\end{aligned}$$

$$\overset{(e)}{\leq} L_{f,1}\Delta(t) + L_{f,2}L_u(L_{g,1}\|\overline{X}(t,\omega) - X(t,\omega)\| + L_{g,2}w^*) + L_{f,3}v^*$$
$$\overset{(f)}{\leq} L_{f,1}\Delta(t) + L_{f,2}L_u(L_{g,1}\Delta(t) + L_{g,2}w^*) + L_{f,3}v^* = \Delta(t+1).$$

Equalities (a) and (b) follow from plugging in the dynamics of the nominal and perturbed systems. Inequality (c) follows by plugging in the Lipschitz constants of f and applying the bounds on V and $\overline{V}$. Inequality (d) follows from plugging in the Lipschitz constant of u, while ineqaulity (e) follows by the Lipschitz constants of g and the bounds on W and $\overline{W}$. Finally, inequality (f) follows from plugging in the bound from the previous time step $\Delta(t)$.

Appendix G. Semantics of STL

We now define the semantics of an STL formula ϕ as in (8) for a signal x and thereby give ϕ a meaning. To do so, we define the satisfaction function $\beta^\phi : \mathfrak{F}(T,\mathbb{R}^n) \times T \to \mathbb{B}$ where $\beta^\phi(x,t) = \top$ indicates that the signal x satisfies the formula ϕ at time t, while $\beta^\phi(x,t) = \bot$ indicates that x does not satisfy ϕ at time t. For a signal $x : \mathbb{N} \to \mathbb{R}^n$, the semantics $\beta^\phi(x,t)$ of an STL formula ϕ are inductively defined as

$$\begin{aligned}
\beta^\top(x,t) &:= \top,\\
\beta^\mu(x,t) &:= \begin{cases} \top & \text{if } x(t) \in O^\mu \\ \bot & \text{otherwise,} \end{cases}\\
\beta^{\neg\phi}(x,t) &:= \neg\beta^\phi(x,t),\\
\beta^{\phi'\wedge\phi''}(x,t) &:= \min(\beta^{\phi'}(x,t), \beta^{\phi''}(x,t)),\\
\beta^{\phi' U_I \phi''}(x,t) &:= \sup_{t'' \in t \oplus (I \cap \mathbb{N})} \Big(\min\big(\beta^{\phi''}(x,t''), \inf_{t' \in (t,t'') \cap \mathbb{N}} \beta^{\phi'}(x,t')\big)\Big).
\end{aligned}$$

In the above definition, the symbols $\oplus$ and $\ominus$ denote the Minkowski sum and the Minkowski difference, respectively. Finally, an STL formula ϕ is said to be satisfiable if $\exists x \in \mathfrak{F}(T,\mathbb{R}^n)$ such that $\beta^\phi(x,0) = \top$.

We say that an STL formula ϕ is bounded if all time intervals I in ϕ are bounded. A bounded STL formula ϕ has a formula length L^ϕ that can be calculated, similarly to [64], as

$$\begin{aligned}
L^\top = L^\mu &:= 0\\
L^{\neg\phi} &:= L^\phi\\
L^{\phi'\wedge\phi''} &:= \max(L^{\phi'}, L^{\phi''})\\
L^{\phi' U_I \phi''} &:= \max\{I \cap \mathbb{N}\} + \max(L^{\phi'}, L^{\phi''})
\end{aligned}$$

A finite signal of length L^ϕ is now sufficient to determine if ϕ is satisfied at time t. In particular, a signal $x : T_L \to \mathbb{R}^n$, where $T_L := \{t, \dots, t, \dots, t + L^\phi\}$, contains sufficient information to determine if ϕ is satisfied by the signal x at time t.

Appendix H. Robustness degree and robust semantics of STL

Let us define the set of signals that satisfy ϕ at time t as

$$\mathcal{L}^\phi(t) := \{x \in \mathfrak{F}(T,\mathbb{R}^n) | \beta^\phi(x,t) = \top\}.$$

For a signal $x \in \mathcal{L}^\phi(t)$, the robustness degree $\text{dist}^\phi(x,t)$ then tells us how much the signal x can be perturbed by additive noise before changing the Boolean truth value of the specification ϕ. Towards a formal definition, let us define the signal metric

$$\kappa(x,x^*) := \sup_{t \in T} \|x(t) - x^*(t)\|.$$

Note that $\kappa(x,x^*)$ is the L_∞ norm of the signal $x - x^*$. The distance of x to the set $\mathcal{L}^\phi(t)$ is then defined via the metric κ as

$$\text{dist}^\phi(x,t) = \bar{\kappa}\big(x, \text{cl}(\mathcal{L}^\phi(t))\big) := \inf_{x^* \in \text{cl}(\mathcal{L}^\phi(t))} \kappa(x,x^*),$$

where $\text{cl}(\mathcal{L}^\phi(t))$ denotes the closure of $\mathcal{L}^\phi(t)$.

As remarked the function $\text{dist}^\phi(x,t)$ is hard to evaluate. Instead, the robust semantics that are associated with an STL formula ϕ can be used. For a signal $x : \mathbb{N} \to \mathbb{R}^n$, the robust semantics $\rho^\phi(x,t)$ of an STL formula ϕ are inductively defined as

$$\rho^{\top}(x,t) := \top,$$
$$\rho^{\mu}(x,t) := \begin{cases} \mathrm{dist}^{\neg\mu}(x,t) & \text{if } x \in \mathcal{L}^{\mu}(t) \\ -\mathrm{dist}^{\mu}(x,t) & \text{otherwise,} \end{cases}$$
$$\rho^{\neg\phi}(x,t) := -\rho^{\phi}(x,t),$$
$$\rho^{\phi'\wedge\phi''}(x,t) := \min(\rho^{\phi'}(x,t), \rho^{\phi''}(x,t)),$$
$$\rho^{\phi' U_I \phi''}(x,t) := \sup_{t'' \in t \oplus (I \cap \mathbb{N})} \Big(\min\big(\rho^{\phi''}(x,t''), \inf_{t' \in (t,t'') \cap \mathbb{N}} \rho^{\phi'}(x,t')\big)\Big).$$

Note that the difference between the non-robust and robust STL semantics lies solely in the way these operators are defined for predicates μ.

Appendix I. Proof of Lemma 2

Our goal is to show that the inequalities in (9) and (10) hold. We present the proof for (10) first, i.e., the case of a bounded STL formula ϕ with formula length L^{ϕ} and for the trajectory error $\Delta(t)$ as in (6). The result for (9) then follows the same way with minor modifications and is omitted. For an STL formula ϕ as in (8), we hence want to show that $\rho^{\phi}(X(\cdot,\omega),t) - \Delta(t+L^{\phi}) \leq \rho^{\phi}(\overline{X}(\cdot,\omega),t))$ for each realization $\omega \in \Omega$ and time $t \in \mathbb{T}$.

For convenience, let again $\overline{x} := \overline{X}(\cdot,\omega)$ and $x := X(\cdot,\omega)$. The proof follows inductively on the structure of ϕ as in (8) and without considering the case of negations as ϕ is assumed to be in positive normal form.

Predicates. First recall the STL robust semantics for predicates defined in Appendix H as

$$\rho^{\mu}(x,t) := \begin{cases} \mathrm{dist}^{\neg\mu}(x,t) & \text{if } x \in \mathcal{L}^{\mu}(t) \\ -\mathrm{dist}^{\mu}(x,t) & \text{otherwise,} \end{cases}$$

and note that

$$\mathrm{dist}^{\mu}(x,t) = \bar{d}\big(x(t), \mathrm{cl}(O^{\mu})\big) := \inf_{x' \in \mathrm{cl}(O^{\mu})} \|x(t) - x'\|,$$

according to [63, Lemma 57]. It further holds that $\bar{d}$ is a Lipschitz continuous function with Lipschitz constant one, see for instance [72, Chapter 3]. Accordingly, it holds that $\rho^{\mu}(x,t)$ is Lipschitz continuous in its first argument with Lipschitz constant one. For each predicate μ_i and for each time $t' \leq t + L^{\phi}$, it hence follows that $|\rho^{\mu_i}(x,t') - \rho^{\mu_i}(\overline{x},t')| \leq \|x(t') - \overline{x}(t')\| \leq \Delta(t') \leq \Delta(t+L^{\phi})$ where the last inequality follows by since $\Delta(t') \leq \Delta(t+L^{\phi})$ for $t' \leq t+L^{\phi}$. Consequently, we have that $\rho^{\mu_i}(\overline{x},t') \geq \rho^{\mu_i}(x,t) - \Delta(t+L^{\phi})$ for $t' \leq t+L^{\phi}$.

Conjunctions. For conjunctions $\phi' \wedge \phi''$ and by the induction assumption, it holds that $\rho^{\phi'}(\overline{x},t') \geq \rho^{\phi'}(x,t') - \Delta(t+L^{\phi})$ and $\rho^{\phi''}(\overline{x},t') \geq \rho^{\phi''}(x,t') - \Delta(t+L^{\phi})$ for $t' \leq t + L^{\phi} - L^{\phi'\wedge\phi''}$. Now, for $t' \leq t + L^{\phi} - L^{\phi'\wedge\phi''}$, it follows that

$$\begin{aligned} \rho^{\phi'\wedge\phi''}(\overline{x},t') &= \min(\rho^{\phi'}(\overline{x},t'), \rho^{\phi'}(\overline{x},t')) \\ &\geq \min(\rho^{\phi'}(x,t'), \rho^{\phi''}(x,t')) - \Delta(t+L^{\phi}) \\ &= \rho^{\phi'\wedge\phi''}(x,t') - \Delta(t+L^{\phi}). \end{aligned}$$

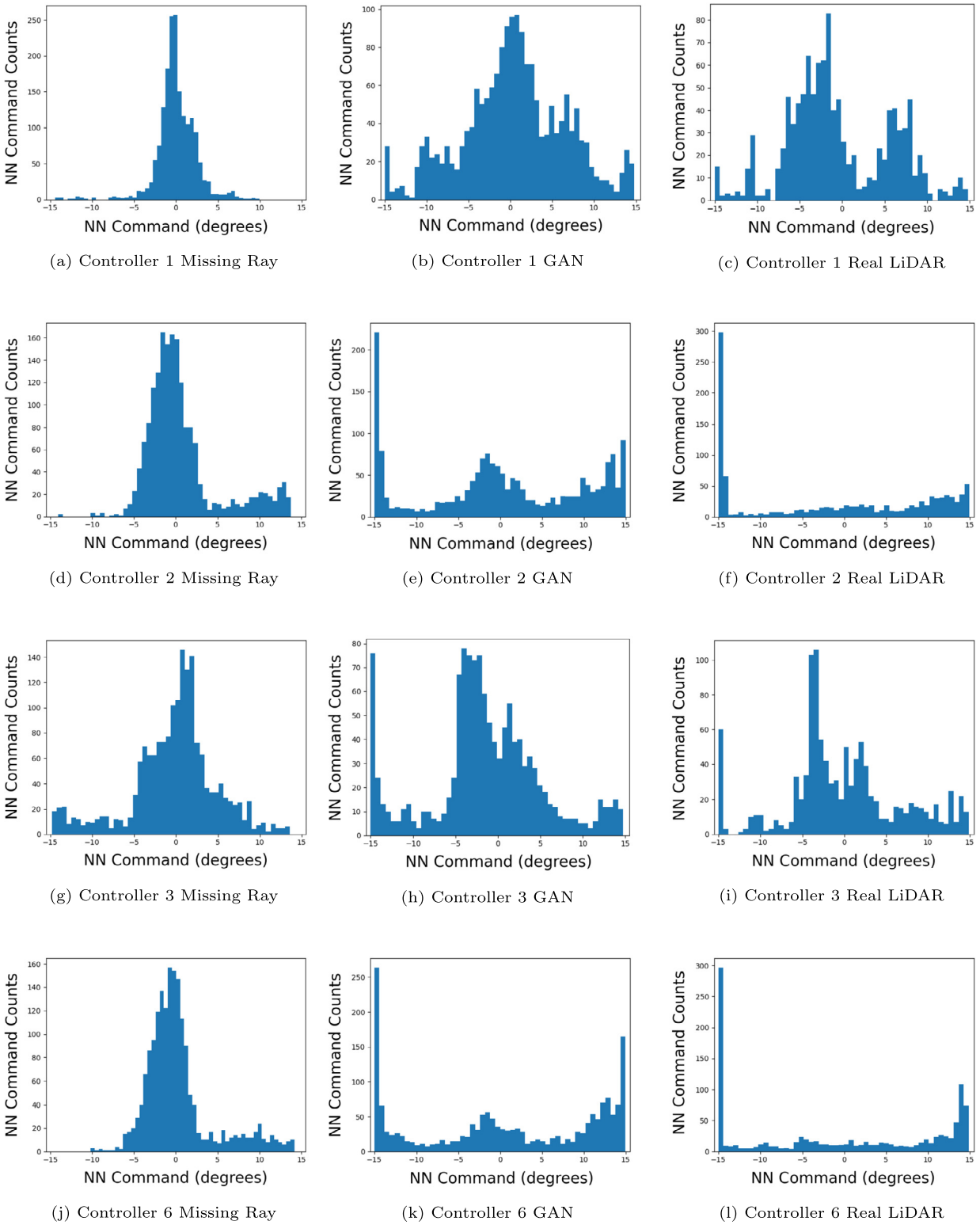
Until. Using the induction assumption and the same reasoning as for conjunctions, it follows that $\rho^{\phi' U_I \phi''}(\overline{x},t') \geq \rho^{\phi' U_I \phi''}(x,t') - \Delta(t+L^{\phi})$ for $t' \leq t + L^{\phi} - L^{\phi' U_I \phi''}$, and it can hence be concluded that (9) holds.

Appendix J. Proof of Theorem 3

We present the proof for a bounded STL formula ϕ with formula length L^{ϕ} and for the trajectory error $\Delta(t)$ as in (6) first, i.e., we first show that $R(-\rho^{\phi}(\overline{X},t)) \leq R(-\rho^{\phi}(X,t)) + \Delta(t+L^{\phi})$. The proof for $R(-\rho^{\phi}(\overline{X},t)) \leq R(-\rho^{\phi}(X,t)) + \Delta$ then follows the same way, i.e., for unbounded STL formulas ϕ and for the trajectory error Δ. By (9) and since the risk metric R is monotone (see Appendix A), it follows that $R(-\rho^{\phi}(\overline{X},t)) \leq R(-\rho^{\phi}(X,t) + \Delta(t+L^{\phi}))$. Since R is additionally translationally invariant (see Appendix A), it further holds that $R(-\rho^{\phi}(X,t) + \Delta(t+L^{\phi})) = R(-\rho^{\phi}(X,t)) + \Delta(t+L^{\phi})$ so that $R(-\rho^{\phi}(\overline{X},t)) \leq R(-\rho^{\phi}(X,t)) + \Delta(t+L^{\phi})$, which concludes the proof.

Appendix K. F1/10 GAN figures

In this section, we report in Fig. K.13 the remaining histograms for the GAN evaluation from Section 7.1.4.



(a) Controller 1 Missing Ray (b) Controller 1 GAN (c) Controller 1 Real LiDAR

(d) Controller 2 Missing Ray (e) Controller 2 GAN (f) Controller 2 Real LiDAR

(g) Controller 3 Missing Ray (h) Controller 3 GAN (i) Controller 3 Real LiDAR

(j) Controller 6 Missing Ray (k) Controller 6 GAN (l) Controller 6 Real LiDAR

Fig. K.13. Histograms showing control command distributions for the GAN LiDAR model and real LiDAR data for various controllers.

References

[1] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, R. Fergus, Intriguing properties of neural networks, preprint, arXiv:1312.6199.
[2] A. Pnueli, The temporal logic of programs, in: Proceedings of the Symposium on Foundations of Computer Science, Washington, DC, 1977, pp. 46–57.
[3] T. Anevlavis, M. Philippe, D. Neider, P. Tabuada, Being correct is not enough: efficient verification using robust linear temporal logic, ACM Trans. Comput. Log. 23 (2) (2022) 1–39.
[4] O. Maler, D. Nickovic, Monitoring temporal properties of continuous signals, in: Proceedings of the Conference on Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems, Grenoble, France, 2004, pp. 152–166.
[5] E. Bartocci, L. Bortolussi, L. Nenzi, G. Sanguinetti, System design of stochastic models using robustness of temporal properties, Theor. Comput. Sci. 587 (2015) 3–25.
[6] L. Lindemann, N. Matni, G.J. Pappas, STL robustness risk over discrete-time stochastic processes, in: Proceedings of the Conference on Decision and Control, Austin, TX, 2021, pp. 1329–1335.
[7] R. Ivanov, T.J. Carpenter, J. Weimer, R. Alur, G.J. Pappas, I. Lee, Case study: verifying the safety of an autonomous racing car with a neural network controller, in: Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control, Sydney, New South Wales, 2020.
[8] A. Legay, A. Lukina, L.M. Traonouez, J. Yang, S.A. Smolka, R. Grosu, Statistical model checking, in: Computing and Software Science, Springer, 2019, pp. 478–504.
[9] A. Wald, Sequential tests of statistical hypotheses, Ann. Math. Stat. 16 (2) (1945) 117–186.
[10] Y. Wang, M. Zarei, B. Bonakdarpour, M. Pajic, Statistical verification of hyperproperties for cyber-physical systems, ACM Trans. Embed. Comput. Syst. 18 (5s) (2019) 1–23.
[11] A. David, K.G. Larsen, A. Legay, M. Mikučionis, D.B. Poulsen, S. Sedwards, Statistical model checking for biological systems, Int. J. Softw. Tools Technol. Transf. 17 (3) (2015) 351–367.
[12] C. Jegourel, A. Legay, S. Sedwards, Importance splitting for statistical model checking rare properties, in: Proceedings of the International Conference on Computer Aided Verification, Springer, Saint Petersburg, Russia, 2013, pp. 576–591.
[13] A. Majumdar, M. Pavone, How should a robot assess risk? Towards an axiomatic theory of risk in robotics, in: Robotics Research, Springer, 2020, pp. 75–84.
[14] Y. Wang, M.P. Chapman, Risk-averse autonomous systems: a brief history and recent developments from the perspective of optimal control, Artif. Intell. (2022) 103743.
[15] A. Tsiamis, D.S. Kalogerias, L.F. Chamon, A. Ribeiro, G.J. Pappas, Risk-constrained linear-quadratic regulators, in: Proceedings of the Conference on Decision and Control, Jeju Island, Republic of Korea, 2020, pp. 3040–3047.
[16] Y. Chow, A. Tamar, S. Mannor, M. Pavone, Risk-sensitive and robust decision-making: a CVaR optimization approach, in: Proceeding of the Conference on Advances in Neural Information Processing Systems, Montreal, Canada, 2015.
[17] A. Wachi, Y. Sui, Safe reinforcement learning in constrained Markov decision processes, in: Proceedings of the International Conference on Machine Learning, 2020, pp. 9797–9806.
[18] R.S. Novin, A. Yazdani, A. Merryweather, T. Hermans, Risk-aware decision making for service robots to minimize risk of patient falls in hospitals, in: Proceedings of the International Conference on Robotics and Automation, Xi'an, China, 2021, pp. 3299–3305.
[19] R. Khodabakhsh, S. Sirouspour, Optimal control of energy storage in a microgrid by minimizing conditional value-at-risk, IEEE Trans. Sustain. Energy 7 (3) (2016) 1264–1273, https://doi.org/10.1109/TSTE.2016.2543024.
[20] Y. Chow, M. Ghavamzadeh, L. Janson, M. Pavone, Risk-constrained reinforcement learning with percentile risk criteria, J. Mach. Learn. Res. 18 (1) (2017) 6070–6120.
[21] M.P. Chapman, J. Lacotte, A. Tamar, D. Lee, K.M. Smith, V. Cheng, J.F. Fisac, S. Jha, M. Pavone, C.J. Tomlin, A risk-sensitive finite-time reachability approach for safety of stochastic dynamic systems, in: Proceedings of the American Control Conference, Philadelphia, PA, 2019, pp. 2958–2963.
[22] M.P. Chapman, R. Bonalli, K.M. Smith, I. Yang, M. Pavone, C.J. Tomlin, Risk-sensitive safety analysis using conditional value-at-risk, IEEE Trans. Autom. Control (2021).
[23] M. Ono, M. Pavone, Y. Kuwata, J. Balaram, Chance-constrained dynamic programming with application to risk-aware robotic space exploration, Auton. Robots 39 (4) (2015) 555–571.
[24] S. Thiébaux, B. Williams, et al., RAO*: an algorithm for chance-constrained POMDP's, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30, Phoenix, Arizona, 2016.
[25] B. Peng, Y. Mu, J. Duan, Y. Guan, S.E. Li, J. Chen, Separated proportional-integral lagrangian for chance constrained reinforcement learning, in: Proceedings of the Intelligent Vehicles Symposium, Nagoya, Japan, 2021, pp. 193–199.
[26] G. Katz, C. Barrett, D.L. Dill, K. Julian, M.J. Kochenderfer, Reluplex: an efficient smt solver for verifying deep neural networks, in: Proceedings of the International Conference on Computer Aided Verification, Heidelberg, Germany, 2017, pp. 97–117.
[27] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, M. Vechev, Ai2: safety and robustness certification of neural networks with abstract interpretation, in: Proceedings of the Symposium on Security and Privacy, San Francisco, CA, 2018, pp. 3–18.
[28] S. Dutta, S. Jha, S. Sankaranarayanan, A. Tiwari, Output range analysis for deep feedforward neural networks, in: Proceedings of NASA Formal Methods Symposium, Newport News, VA, 2018, pp. 121–138.
[29] M. Fazlyab, M. Morari, G.J. Pappas, Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming, IEEE Trans. Autom. Control (2020).
[30] A. Xue, L. Lindemann, R. Alur, Parametric chordal sparsity for sdp-based neural network verification, preprint, arXiv:2206.03482.
[31] M. Fazlyab, A. Robey, H. Hassani, M. Morari, G. Pappas, Efficient and accurate estimation of Lipschitz constants for deep neural networks, in: Proceedings of the Conference on Advances in Neural Information Processing Systems, vol. 32, 2019, pp. 11427–11438.
[32] R. Ivanov, J. Weimer, R. Alur, G.J. Pappas, I. Lee, Verisig: verifying safety properties of hybrid systems with neural network controllers, in: Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control, Montreal, Quebec, 2019, pp. 169–178.
[33] S. Dutta, X. Chen, S. Jha, S. Sankaranarayanan, A. Tiwari, Sherlock - a tool for verification of neural network feedback systems: demo abstract, in: Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control, Montreal, Quebec, 2019, pp. 262–263.
[34] J. Fan, C. Huang, X. Chen, W. Li, Q. Zhu, Reachnn*: a tool for reachability analysis of neural-network controlled systems, in: Proceedings of the International Symposium on Automated Technology for Verification and Analysis, Hanoi, Vietnam, 2020, pp. 537–542.
[35] H.-D. Tran, D.M. Lopez, X. Yang, P. Musau, L.V. Nguyen, W. Xiang, S. Bak, T.T. Johnson, The neural network verification (NNV) tool, in: Proceedings of the Workshop on Design Automation for CPS and IoT, 2020, pp. 21–22.
[36] C. Huang, J. Fan, X. Chen, W. Li, Q. Zhu, Polar: a polynomial arithmetic framework for verifying neural-network controlled systems, arXiv:2106.13867, 2021.
[37] D. Vengertsev, E. Sherman, Recurrent neural network properties and their verification with Monte Carlo techniques, in: SafeAI@AAAI, 2020.
[38] S. Park, O. Bastani, N. Matni, I. Lee, Pac confidence sets for deep neural networks via calibrated prediction, in: Proceedings of the International Conference on Learning Representations, 2020.
[39] S. Bates, A. Angelopoulos, L. Lei, J. Malik, M. Jordan, Distribution-free, risk-controlling prediction sets, J. ACM 68 (6) (2021) 1–34.

[40] S. Webb, T. Rainforth, Y.W. Teh, M.P. Kumar, A statistical approach to assessing neural network robustness, in: International Conference on Learning Representations, 2018.
[41] R. Mangal, A.V. Nori, A. Orso, Robustness of neural networks: a probabilistic and practical approach, in: Proceedings of the International Conference on Software Engineering: New Ideas and Emerging Results, Montreal, Canada, 2019, pp. 93–96.
[42] C. Baier, J.-P. Katoen, Principles of Model Checking, 1st edition, The MIT Press, Cambridge, MA, 2008.
[43] M. Zarei, Y. Wang, M. Pajic, Statistical verification of learning-based cyber-physical systems, in: Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control, Sydney, New South Wales, 2020, pp. 1–7.
[44] X. Qin, Y. Xian, A. Zutshi, C. Fan, J.V. Deshmukh, Statistical verification of cyber-physical systems using surrogate models and conformal inference, in: Proceedings of the International Conference on Cyber-Physical Systems, Milan, Italy, 2022, pp. 116–126.
[45] Y. Wang, N. Roohi, M. West, M. Viswanathan, G.E. Dullerud, Statistical verification of dynamical systems using set oriented methods, in: Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control, Seattle, Washington, 2015, pp. 169–178.
[46] N. Roohi, Y. Wang, M. West, G.E. Dullerud, M. Viswanathan, Statistical verification of the Toyota powertrain control verification benchmark, in: Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control, Pittsburgh, Pennsylvania, 2017, pp. 65–70.
[47] Y. Wang, M. Zarei, B. Bonakdarpoor, M. Pajic, Probabilistic conformance for cyber-physical systems, in: Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems, Nashville, Tennessee, 2021, pp. 55–66.
[48] A. Salamati, S. Soudjani, M. Zamani, Data-driven verification under signal temporal logic constraints, IFAC-PapersOnLine 53 (2) (2020) 69–74.
[49] A. Salamati, S. Soudjani, M. Zamani, Data-driven verification of stochastic linear systems with signal temporal logic constraints, Automatica 131 (2021) 109781.
[50] J. Jackson, L. Laurenti, E. Frew, M. Lahijanian, Formal verification of unknown dynamical systems via gaussian process regression, preprint, arXiv: 2201.00655.
[51] E. Bartocci, L. Bortolussi, L. Nenzi, G. Sanguinetti, On the robustness of temporal properties for stochastic models, preprint, arXiv:1309.0866.
[52] F1/10 autonomous racing competition, https://f1tenth.org/.
[53] M.M.M. Manhães, S.A. Scherer, M. Voss, L.R. Douat, T. Rauschenbach, UUV simulator: a gazebo-based package for underwater intervention and multi-robot simulation, in: Proceedings of OCEANS 2016 MTS/IEEE Monterey, Monterey, California, 2016, pp. 1–8.
[54] R.T. Rockafellar, S. Uryasev, et al., Optimization of conditional value-at-risk, J. Risk 2 (2000) 21–42.
[55] B. Szorenyi, R. Busa-Fekete, P. Weng, E. Hüllermeier, Qualitative multi-armed bandits: a quantile-based approach, in: Proceedings of the International Conference on Machine Learning, PMLR, 2015, pp. 1660–1668.
[56] K.E. Nikolakakis, D.S. Kalogerias, O. Sheffet, A.D. Sarwate, Quantile multi-armed bandits: optimal best-arm identification and a differentially private scheme, IEEE J. Sel. Areas Inf. Theory 2 (2) (2021) 534–548.
[57] P. Thomas, E. Learned-Miller, Concentration inequalities for conditional value at risk, in: Proceedings of the International Conference on Machine Learning, Long Beach, CA, 2019, pp. 6225–6233.
[58] A. Huang, L. Leqi, Z. Lipton, K. Azizzadenesheli, Off-policy risk assessment in contextual bandits, in: Proceedings of the Conference on Advances in Neural Information Processing Systems, vol. 34, 2021.
[59] D.N. Tran, B.S. Rüffer, C.M. Kellett, Incremental stability properties for discrete-time systems, in: Proceedings of the Conference on Decision and Control, Las Vegas, NV, 2016, pp. 477–482.
[60] S. Tu, A. Robey, T. Zhang, N. Matni, On the sample complexity of stability constrained imitation learning, in: Proceedings of the Conference on Learning for Dynamics and Control Conference, Palo Alto, CA, 2022, pp. 180–191.
[61] A.J. Havens, B. Hu, On imitation learning of linear control policies: enforcing stability and robustness constraints via lmi conditions, in: Proceedings of the American Control Conference, 2021, pp. 882–887.
[62] N.M. Boffi, S. Tu, J.-J.E. Slotine, Regret bounds for adaptive nonlinear control, in: Learning for Dynamics and Control, PMLR, 2021, pp. 471–483.
[63] G.E. Fainekos, G.J. Pappas, Robustness of temporal logic specifications for continuous-time signals, Theor. Comput. Sci. 410 (42) (2009) 4262–4291.
[64] S. Sadraddini, C. Belta, Robust temporal logic model predictive control, in: Proceedings of the 53rd Annual Allerton Conference on Communication, Control, and Computing, Monticello, IL, 2015, pp. 772–779.
[65] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, in: Proceedings of the International Conference on Machine Learning, Sydney, Australia, 2017, pp. 214–223.
[66] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, A.C. Courville, Improved training of Wasserstein gans, in: Proceedings of the Conference on Advances in Neural Information Processing Systems, Long Beach, CA, vol. 30, 2017.
[67] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, preprint, arXiv:1511.06434.
[68] T.P. Lillicrap, J.J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous control with deep reinforcement learning, preprint, arXiv:1509.02971.
[69] S. Fujimoto, H. Hoof, D. Meger, Addressing function approximation error in actor-critic methods, in: Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 2018, pp. 1587–1596.
[70] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, Improved techniques for training gans, in: Proceedings of the Conference on Advances in Neural Information Processing Systems, Barcelona, Spain, vol. 29, 2016.
[71] S. Vallender, Calculation of the Wasserstein distance between probability distributions on the line, Theory Probab. Appl. 18 (4) (1974) 784–786.
[72] J.R. Munkres, Topology, 2nd edition, Prentice Hall, 2000.
[73] A.H. Guide, Infinite Dimensional Analysis, Springer, 2006.