Conservative Perception Models for Probabilistic Model Checking

1st Matthew Cleaveland
MIT Lincoln Laboratory
Lexington, MA, USA
matthew.cleaveland@ll.mit.edu

2nd Pengyuan Lu
University of Pennsylvania
Philadelphia, PA, USA
pelu@seas.upenn.edu

3rd Oleg Sokolsky
University of Pennsylvania
Philadelphia, PA, USA
sokolsky@seas.upenn.edu

4th Insup Lee
University of Pennsylvania
Philadelphia, PA, USA
lee@seas.upenn.edu

5th Ivan Ruchkin
University of Florida
Gainesville, FL, USA
iruchkin@ece.ufl.edu

***Abstract*—Verifying the behaviors of autonomous systems with learned perception components is a challenging problem due to the complexity of the perception and the uncertainty of operating environments. Probabilistic model checking is a powerful tool for providing guarantees on stochastic models of systems. However, constructing model-checkable models of black-box perception components for system-level mathematical guarantees has been an enduring challenge. In this paper, we propose a method for constructing provably conservative Interval Markov Decision Process (IMDP) models of closed-loop systems with perception components. We prove that our technique results in conservative abstractions with a user-specified probability. We evaluate our approach in an automatic braking case study using the object detector YOLO11 in the CARLA driving simulator.**

***Index Terms*—neural perception, probabilistic model checking, interval Markov decision process**

I. Introduction

Advances in machine learning have led to systems with complex learning-enabled components (LECs). These LECs tend to exhibit unpredictable, difficult-to-model behaviors. The uncertainties of LECs and real-world environments are frequently handled with probabilistic model checking (PMC) [25], which can be used to compute the probability that a safety property holds. PMC is particularly suitable for analyzing highly uncertain systems since it explicitly models both stochastic and non-deterministic behaviors.

Using PMC to verify closed-loop systems requires building a stochastic model of the system, usually as a probabilistic automaton (PA) or a Markov decision process (MDP). This model is often abstracted from a more detailed description, such as differential or difference equations. Then model checking tools, such as PRISM [24], are used to analyze that abstraction, such as computing the probability of the car colliding with an obstacle. If the abstraction overapproximates the true underlying system (i.e., contains all of its possible behaviors), PMC provides safety guarantees about the true system. Such abstractions are referred to as *conservative abstractions*.

DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited.

Alas, constructing conservative abstractions of realistic systems is difficult. Consider a closed-loop autonomous system consisting of a controller, sensor, perception, and physical dynamics. Discretizing and conservatively modeling the LEC-free control and dynamics components as an MDP with non-determinism is well-understood (see, e.g., [8]). However, constructing provably conservative MDP abstractions of high-dimensional sensing (e.g., a camera) and LEC-based perception is an open question in the literature, due to the uncertainty of the environment (e.g., the weather conditions affecting the camera) and the complexity of the perception algorithms (e.g., a large object detection neural network).

Two major uncertainties obstruct the creation of provably conservative probabilistic abstractions of sensing and perception LECs, even given a sizeable validation dataset. First, one must account for the inherent uncertainty in using data-driven methods to fit MDP parameters, e.g., by considering the confidence intervals of the estimated parameters of the model. Second, it is difficult to account for the mismatch between discrete MDP models and sensing/perception components that vary continuously with respect to the (continuous) state of the true system. Existing perception abstraction methods either only consider the first challenge [28], or consider neither of these challenges [5], [9], instead just assuming that their perception abstractions are conservative.

This paper develops a data-driven approach to constructing provably conservative MDP models of sensing/perception for verification of LEC-based autonomous systems. As system models, we employ *interval Markov decision processes (IMDPs)* — probabilistic models that allow for transition probabilities to take any value within an interval. Our approach designs specialized *confidence intervals (CIs)* to account for the inherent uncertainty in fitting a model using data (the first uncertainty above) and *logistic regression* to estimate how the perception behaviors change within the discrete abstract states (the second uncertainty above). We prove that our abstractions are conservative with high probability.

We evaluate our approach on an automatic emergency

braking system (AEBS) for an autonomous car with a YOLO11 perception component in the self-driving car simulator CARLA [13]. Our experiments show that our abstractions are indeed empirically conservative, unlike simpler baselines. We also investigate how different hyperparameters of our approach affect the resulting safety estimates of the AEBS.

The contributions of this paper are three-fold:

- A method for constructing conservative models of perception components with high probability.
- A proof that our perception abstractions, when composed with conservative abstractions of controller/plant models, produce conservative system models.
- An empirical evaluation of our abstractions on an AEBS case study using a learning-based perception model.

The rest of the paper is organized as follows. We survey related work in Section II and give the necessary formal background in Section III. Section IV introduces our problem statement, and Section V introduces our perception abstraction approach. We state our conservatism guarantee in Section VI. We describe the results of our two case studies in Section VII. The paper ends with a conclusion in Section VIII.

II. Related Work

A. Safe Perception

Autonomous safety depends on the accuracy of perception. Therefore, researchers in safety-critical systems aim to improve vision quality. First, fail-safe perceptions can be constructed by improved training quality. For instance, training quality may be measured by adversarial robustness [26]. An alternative approach is to employ online monitoring and intervention so that inaccurate predictions are replaced. Such approaches include detectors for unfamiliar inputs (which are more likely to be mis-predicted) [30] and diagnostic graphs that online analyze the cause of mis-predictions through module dependency [3]. Notice that neither of the two approaches guarantees successful predictions on sets of inputs. Moreover, researchers have assisted the above two solutions methods to identify the vulnerable inputs to perception models [22]. Specifically, they inject manually designed disturbance into the perception's inputs to mimic different adversarial scenarios, to find the inputs with a high likelihood of failure. Additionally, researchers have developed techniques for incorporating system-level safety requirements into perception loss functions [11] and those same safety requirements into accuracy requirements of the perception models [21], providing useful information in perception construction and fine-tuning.

B. Model Checking

Model checking is an established approach to verify whether a system satisfies a given formal specification. Verification literature first focuses on deterministic overapproximate reachability, such as Verisig [16], which proves specifications for neural networks with nonlinear activation functions. Recent research emphasizes probabilistic verification, where the system is represented as a stochastic process. These representations include discrete and continuous-time Markov chains (DMTCs and CMTCs), Markov decision processes (MDPs), probabilistic timed automata (PTAs) [24], [2], as well as interval Markov chains and Markov decision processes (IMC and IMDP) [4]. Existing works in the literature have also investigated controller synthesis on IMCs and IMDPs [18], [33]. We build on this literature to verify a system model that provably accounts for black-box perception uncertainty.

C. Perception Abstractions

Another studied problem is how to abstractly represent the perception process. Many approaches using probabilistic models like Markov Decision Processes (MDPs) or similar to abstract perception components for model checking [28], [5], [9]. Other uses of perception models include run-time monitoring [31], [27] and falsification [14]. Another line of research constructs conservative abstractions of full system behaviors using stochastic models like MDPs [1] or Generalized Polynomial Chaos [20]. These approaches do not (and often cannot) prove that their model of a real-world object detector is conservative with respect to physical reality.

Alternatively, for specific vision systems, researchers have built detailed parameterized models (e.g., of a pinhole camera), primarily for road-following autonomous vehicle cameras [20], [28] but also aircraft landing cameras [32] and racing car LiDARs [17]. However, such approaches are difficult to generalize to visually complex phenomena in a wide range of systems.

The most similar works to ours are [5], [28]. In [5], the authors model the perception component of an autonomous system as a distance-parameterized confusion matrix. They then fit the parameters of the model using perception data from the system. However, the authors do not account for either of the two uncertainties mentioned in the introduction. Therefore, their abstraction may not be conservative, which could lead to a false positive verification verdict. In [28], the authors use uncertainty quantification methods to compute confidence intervals for perception LECs. However, their approach does not consider the mismatch between discrete PA models and continuous sensing/perception components.

III. Background: Probabilistic Model Checking

Definition 1 (MDP)**.** *A* Markov Decision Process *(MDP) is a tuple* $\mathsf{M}^{mdp} = (S, s_0, \alpha, \delta, L)$*, where* S *is a finite set of states,* $s_0 \in S$ *is the initial state,* α *is an alphabet of action labels,* $\delta : S \times \alpha \times S \to [0,1]$ *such that* $\forall s \in S, a \in \alpha$, $\sum_{s' \in S} \delta(s,a,s') = 1$ *is a probabilistic transition function, and* $L : S \to 2^{AP}$ *is a state labeling function with atomic propositions* AP*.*

If $\delta(s,a,s') = p$, then M^{mdp} can make a transition in state s with action a to state s' with probability p. A state s is *terminal* if $\forall a \in \alpha : \delta(s,a,s) = 1$. A *path* in M^{mdp} is a (potentially infinite) sequence of transitions $\pi = s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} \dots$ with $\delta(s_i, a_i, s_{i+1}) \neq 0$. A *set of paths* is denoted as Π. We use $\mathsf{M}^{\text{mdp}}(s)$ to denote the MDP M^{mdp} with initial state s.

Definition 2 (IMDP)**.** *An* Interval Markov Decision Process *(IMDP) is a tuple* $\mathsf{M}^{imdp} = (S, s_0, \alpha, \Delta, L)$*, where* S *is a finite*

set of states, $s_0 \in S$ is the initial state, α is an alphabet of action labels, $\Delta : S \times \alpha \times S \to \{[b,c] \mid 0 < b \leq c \leq 1\} \cup \{[0,0]\}$ is an interval transition probability function, and $L : S \to 2^{AP}$ is a state labeling function with atomic propositions AP. We further decompose Δ into its lower and upper components, denoted Δ^l and Δ^u, respectively: $\Delta(s,a,s') = [\Delta^l(s,a,s'), \Delta^u(s,a,s')]$.

Remark 1. *The requirement that $b > 0$ in the definition of Δ ensures that the graph structure of the IMDP stays the same for all the values of $\Delta(s,a,s')$. This is necessary for IMDP model checking algorithms on which we rely.*

Now we define the usual parallel composition for MDPs and IMDPs. In this paper, the perception component will be represented by an MDP/IMDP and the dynamics/control will be represented by an MDP.

Definition 3 (MDP composition)**.** *The* parallel composition *of MDPs $\mathsf{M}_1^{mdp} = \{S_1, s_0^1, \alpha_1, \delta_1, L_1\}$ and $\mathsf{M}_2^{mdp} = \{S_2, s_0^2, \alpha_2, \delta_2, L_2\}$ is given by the MDP $\mathsf{M}_1^{mdp} || \mathsf{M}_2^{mdp} = \{S_1 \times S_2, (s_0^1, s_0^2), \alpha_1 \cup \alpha_2, \delta, L\}$, where $L(s_1,s_2) = L_1(s_1) \cup L_2(s_2)$ and*

$$\delta((s_1,s_2),a,(s_1',s_2')) = \begin{cases} \delta_1(s_1,a,s_1') \cdot \delta_2(s_2,a,s_2') & \text{if } a \in \alpha_1 \bigcap \alpha_2 \\ \delta_1(s_1,a,s_1') & \text{if } a \in \alpha_1 \setminus \alpha_2 \wedge s_2 = s_2' \\ \delta_2(s_2,a,s_2') & \text{if } a \in \alpha_2 \setminus \alpha_1 \wedge s_1 = s_1' \\ 0 & \text{otherwise} \end{cases}$$

Definition 4 (MDP-IMDP composition)**.** *The* parallel composition *of MDP $\mathsf{M}^{mdp} = \{S_1, s_0^1, \alpha_1, \delta_1, L_1\}$ with IMDP $\mathsf{M}^{imdp} = \{S_2, s_0^2, \alpha_2, \Delta_2, L_2\}$ is given by the IMDP $\mathsf{M}^{mdp} || \mathsf{M}^{imdp} = \{S_1 \times S_2, (s_0^1, s_0^2), \alpha_1 \cup \alpha_2, \Delta, L\}$, where $L(s_1,s_2) = L_1(s_1) \cup L_2(s_2)$ and*

$$\delta((s_1,s_2),a,(s_1',s_2')) = \begin{cases} [\delta_1(s_1,a,s_1') \cdot \Delta_2^l(s_2,a,s_2'), \delta_1(s_1,a,s_1') \cdot \Delta_2^u(s_2,a,s_2')] \\ \qquad \text{if } a \in \alpha_1 \bigcap \alpha_2 \\ [\delta_1(s_1,a,s_1'), \delta_1(s_1,a,s_1')] \\ \qquad \text{if } a \in \alpha_1 \setminus \alpha_2 \wedge s_2 = s_2' \\ [\Delta_2^l(s_2,a,s_2'), \Delta_2^u(s_2,a,s_2')] \\ \qquad \text{if } a \in \alpha_2 \setminus \alpha_1 \wedge s_1 = s_1' \\ [0,0] \quad \text{otherwise} \end{cases}$$

We also need a way of determining if the stochastic behaviors of an MDP M^{mdp} are contained within the set of stochastic behaviors defined by an IMDP M^{imdp}. To do that, we start with the known general notion of satisfaction relations [19].

Definition 5 (MDP-IMDP satisfaction)**.** *[Definition 2 in [12]] MDP $\mathsf{M}^{mdp} = \{S_1, s_0^1, \alpha_1, \delta_1, L_1\}$* implements *$IMDP$ $\mathsf{M}^{imdp} = \{S_2, s_0^2, \alpha_2, \Delta_2, L_2\}$ if there exists a* satisfaction relation *$R \subseteq S_1 \times S_2$ such that whenever $s_1 R s_2$, then:*

- *$L_1(s_1) = L_2(s_2)$*
- *There exists a* correspondence function *$\varphi : S_1 \to (S_2 \to [0,1])$ such that*
  - *For all $s_1' \in S_1$ and $a_1 \in \alpha_1$, if $\delta_1(s_1,a_1,s_1') > 0$ then $\varphi(s_1')$ defines a distribution on S_2,*
  - *For all $a_1 \in \alpha_1$, there exists $a_2 \in \alpha_2$ such that $\sum_{s_1' \in S_1} \delta_1(s_1,a_1,s_1')\varphi(s_1')(s_2') \in \Delta_2(s_2,a_2,s_2')$ for all $s_2' \in S_2$, and*
  - *For all $s_1' \in S_1$ and $s_2' \in S_2$, if $\varphi(s_1')(s_2') > 0$, then $s_1' R s_2'$*

Note that the correspondence function φ maps a state from M^{mdp} to a distribution of states in M^{imdp}.

Reasoning about MDPs and IMDPs also requires the notion of *schedulers*, which resolve non-determinism to execute the models. For our purposes, a scheduler σ maps each state of the MDP/IMDP to an available action label in that state. We use $\Pi^{\sigma}_{\mathsf{M}^{\text{imdp}}}$ for the set of all paths through M^{imdp} when controlled by scheduler σ and $Sch_{\mathsf{M}^{\text{imdp}}}$ for the set of all schedulers for M^{imdp}. Similar notions apply to MDPs as well.

This paper is concerned with the probabilities of safety properties in the aforementioned stochastic models, which we state using *Safety* Linear Temporal Logic (LTL) formulas over state labels [23].

Definition 6 (Probability of LTL formula)**.** *For LTL formula ψ, IMDP M^{imdp} (or MDP M^{mdp}), and scheduler $\sigma \in Sch_{\mathsf{M}^{imdp}}$ (or $\sigma \in Sch_{\mathsf{M}^{mdp}}$), the* upper and lower probabilities of ψ holding on M^{imdp} *are:*

$$\begin{aligned} Pr^{\sigma,u}_{\mathsf{M}^{imdp}}(\psi) &\coloneqq Pr^{\sigma,u}_{\mathsf{M}^{imdp}}\{\pi \in \Pi^{\sigma}_{\mathsf{M}^{imdp}} \mid \pi \models \psi\} \\ Pr^{\sigma,l}_{\mathsf{M}^{imdp}}(\psi) &\coloneqq Pr^{\sigma,l}_{\mathsf{M}^{imdp}}\{\pi \in \Pi^{\sigma}_{\mathsf{M}^{imdp}} \mid \pi \models \psi\}, \end{aligned} \tag{1}$$

and the probability of ψ holding on M^{mdp} is

$$Pr^{\sigma}_{\mathsf{M}^{mdp}}(\psi) \coloneqq Pr^{\sigma}_{\mathsf{M}^{mdp}}\{\pi \in \Pi^{\sigma}_{\mathsf{M}^{mdp}} \mid \pi \models \psi\}, \tag{2}$$

where $\pi \models \psi$ indicates that the path π satisfies ψ in the standard LTL semantics [29]. We limit our scope to LTL safety properties, which are LTL specifications that can be falsified by a finite trace.

Probabilistically verifying an LTL formula ψ against M^{imdp} requires checking that the probability of satisfying ψ meets the desired probability bounds for all schedulers. This involves computing the minimum or maximum probability of satisfying ψ over all schedulers:

$$\begin{aligned} Pr^{min}_{\mathsf{M}^{\text{imdp}}}(\psi) &\coloneqq \inf_{\sigma \in Sch_{\mathsf{M}^{\text{imdp}}}} Pr^{\sigma,l}_{\mathsf{M}^{\text{imdp}}}(\psi) \\ Pr^{max}_{\mathsf{M}^{\text{imdp}}}(\psi) &\coloneqq \sup_{\sigma \in Sch_{\mathsf{M}^{\text{imdp}}}} Pr^{\sigma,u}_{\mathsf{M}^{\text{imdp}}}(\psi) \end{aligned} \tag{3}$$

The corresponding definition of an LTL formula ψ holding on an MDP M^{mdp} is:

$$\begin{aligned} Pr^{min}_{\mathsf{M}^{\text{mdp}}}(\psi) &\coloneqq \inf_{\sigma \in Sch_{\mathsf{M}^{\text{mdp}}}} Pr^{\sigma}_{\mathsf{M}^{\text{mdp}}}(\psi) \\ Pr^{max}_{\mathsf{M}^{\text{mdp}}}(\psi) &\coloneqq \sup_{\sigma \in Sch_{\mathsf{M}^{\text{mdp}}}} Pr^{\sigma}_{\mathsf{M}^{\text{mdp}}}(\psi) \end{aligned} \tag{4}$$

IV. Problem: Conservative Verification

This paper addresses the problem of constructing perception models from data to enable conservative system verification. We call such perception models "conservative perception models". An IMDP is a *conservative perception model* if it is

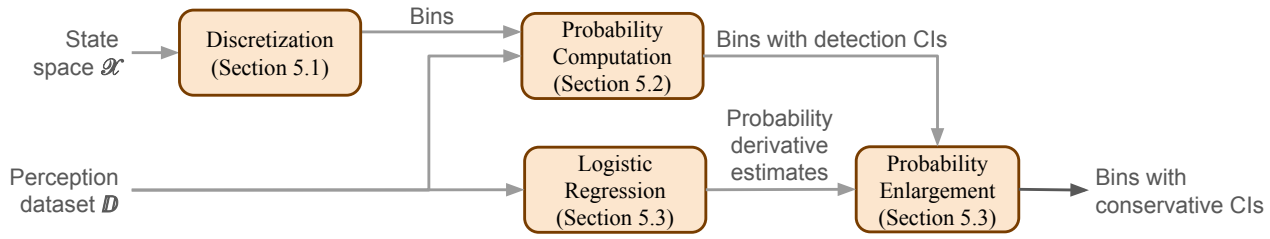

Fig. 1: Overview of our approach to conservative perception modeling.

implemented by the MDP of the true perception, per Definition 5. Consider the following discrete-time dynamical system M_{sys} with a binary classifier or object detector for perception:

$$\mathsf{M}_{\text{sys}} \begin{cases} x(t+1) = f(x(t), u(t))), \\ y(t) = g(x(t), v(t)), \\ z(t) = h(y(t)), \\ u(t) = c(z(t)), \end{cases} \tag{5}$$

where $x(t) \in \mathcal{X} \subset \mathbb{R}^n$ is the system state (with bounded $\mathcal{X}$); $y(t) \in \mathbb{R}^p$ are the observations; $z(t) \in \{0,1\}$ is the output of a perception system[1], $u(t) \in \mathcal{U} \subset \mathbb{R}^m$ is the control output, the functions $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $g : \mathbb{R}^n \times \mathbb{R}^v \to \mathbb{R}^p$, $h : \mathbb{R}^p \to \{0,1\}$ describe the system dynamics, sensor maps, and perception maps; the function $c : \{0,1\} \to \mathbb{R}^m$ is a stateless controller; and $v(t) \in D_v \subseteq \mathbb{R}^v$ describes (potentially unbounded) perception noise. The $v(t)$ noise represents inexact sensing, such as a LiDAR scanner giving noisy distance estimates. Due to the noise, Equation (5) is a stochastic dynamical system.

Along with M_{sys}, we are given an LTL safety property ψ to verify. The verification's goal is to compute a conservative estimate (i.e., a lower bound) of the probability that M_{sys} satisfies ψ. To do this, our objective is to construct an abstract and analyzable model of M_{sys}, ideally as an MDP or IMDP, which we denote as M_{abs}. It also must be a *conservative system model*, defined as follows.

Definition 7 (Conservative system model)**.** *Given system M_{sys} as in Equation* (5)*, a model MDP or IMDP M_{abs}, and safety property ψ, we say the M_{abs} is a* conservative system model *with respect to M_{sys} if*

$$P^{min}_{\mathsf{M}_{abs}}(\psi) \leq P_{\mathsf{M}_{sys}}(\psi) \tag{6}$$

Model structure. When constructing M_{abs}, it is typical to split it into two components: a *controller-plant model* M_{cpl} (of f and c in Eq. 5) and a *perception model* M_{per} (of g and h in Eq. 5). This decomposition is used because the behaviors in M_{cpl} are often (non-)deterministic and/or straightforward to model using first principles (e.g., using the controller equations and system dynamics). On the other hand, the behaviors in M_{per} are typically modeled probabilistically due to their complexity (e.g., large neural networks) and dependency on uncertain physical environments.

Therefore, the controller-plant model M_{cpl} can be built as a conservative non-probabilistic MDP (i.e., one whose behavior contains that of true system M_{sys}) with a standard discrete abstraction technique (for example, in [8]). However, constructing a conservative M_{per}, either as an MDP or IMDP, is an open problem. Therefore, this paper focuses on synthesizing M_{per} in a conservative manner using labeled perception data (i.e., the perception outputs y paired with the true underlying states x) from the system M_{sys}. Section VI will show that our M_{abs} is a conservative system model, as per Definition 7.

With M_{cpl} and M_{per} in hand, we can form M_{abs} via parallel composition that synchronizes perception actions (e.g., detection/non-detection):

$$\mathsf{M}_{\text{abs}} = \mathsf{M}_{\text{cpl}} \mid\mid \mathsf{M}_{\text{per}} \tag{7}$$

Before moving on, we need to define the general structure of M_{cpl} and give a formal definition for what it means for M_{cpl} to be a *conservative controller/plant abstraction*:

Definition 8 (Controller-Plant Model)**.** *Our* controller-plant model M_{cpl} *is a non-probabilistic MDP with* $\mathsf{M}_{cpl} = \{S_{\mathsf{M}_{cpl}}, s^0_{\mathsf{M}_{cpl}}, \alpha_{\mathsf{M}_{cpl}}, \delta_{\mathsf{M}_{cpl}}, L_{\mathsf{M}_{cpl}}\}$*. The (discrete) state space S is formed by partitioning the (continuous) state space $\mathcal{X}$ of M_{sys} into a finite set of mutually exclusive regions, so every $s \in S_{\mathsf{M}_{cpl}}$ has a corresponding region $X \subseteq \mathcal{X}$ (we use $x \in s$ to denote this correspondence). The initial state $s^0_{\mathsf{M}_{cpl}}$ is the state in $S_{\mathsf{M}_{cpl}}$ which contains the initial state x_0 of M_{sys} in its set of corresponding states. The actions $\alpha_{\mathsf{M}_{cpl}} = \alpha_{per} \times \alpha_{reach}$, where $\alpha_{per} = \{0,1\}$ (which we refer to as "perception actions") are the outputs of the perception component h and α_{reach} (which we refer to as "reachability actions") are the actions that correspond to the non-determinism required to conservatively model the continuous system dynamics f and controller u with a discrete abstraction.[2] The transition relation $\delta_{\mathsf{M}_{cpl}}$ maps states and a pair (perception action, reachability actions) to the next state in $S_{\mathsf{M}_{cpl}}$. Finally, $L_{\mathsf{M}_{cpl}} : S \to 2^{AP}$, were AP is a set of atomic propositions.*

Definition 9 (Conservative Controller-Plant Model)**.** $\mathsf{M}_{cpl} = \{S_{\mathsf{M}_{cpl}}, s_0, \alpha_{\mathsf{M}_{cpl}}, \delta_{\mathsf{M}_{cpl}}, L_{\mathsf{M}_{cpl}}\}$ *is a* conservative controller/plant abstraction *if for every state $s \in S_{\mathsf{M}_{cpl}}$ and perception action $a_{per} \in \alpha_{per}$, the set of next states $S' = \{s' \in S \mid \exists a_{reach} \in \alpha_{reach}$ s.t. $\delta_{\mathsf{M}_{cpl}}(s, \{a_{per}, a_{reach}\}, s') = 1\}$ in M_{cpl} contains*

[1]For brevity, we assume the perception system detects the presence of a single obstacle in the environment. However, our approach can handle systems with multiple types of obstacles and $z(t)$ from a fixed finite set.

[2]For a visualization of this phenomenon, see Figure 2 in [8].

every state in M_{sys} *that can be mapped to by the dynamics of* M_{sys}*. Formally:*

$$\forall s \in S_{\mathsf{M}_{cpl}}, \forall a_{per} \in \alpha_{per}, \forall x \in s, \exists s' \in S, \exists a_{reach} \in \alpha_{reach}$$
such that $\delta_{\mathsf{M}_{cpl}}(s, \{a_{per}, a_{reach}\}, s') = 1$ *and* $f(x, u(a)) \in s'$

Remark 2. *The mismatch between the discrete abstractions and continuous system behaviors is a challenge that existing approaches fail to consider when making* M_{per} *abstractions.*

With the above in mind, this paper solves this problem:

Problem 1 (Perception Abstraction Construction for Conservative Verification)**.** *Given the closed-loop dynamical system* M_{sys} *defined in Equation* (5)*, a safety property* ψ*, perception data* $\mathcal{D}_{sys} = \{x_i, z_i\}_{i=1,\dots,N}$ *where* $x_i \in \mathbb{R}^n$ *is the state of* M_{sys} *and* $z_i \in \{0, 1\}$ *is the output of the perception component, and a conservative model* M_{cpl} *of the control and dynamics portion of* M_{sys}*, synthesize a conservative model* M_{per} *of the perception component(s) of* M_{sys} *such that*

$$P^{min}_{\mathsf{M}_{cpl}||\mathsf{M}_{per}}(\psi) \leq P_{\mathsf{M}_{sys}}(\psi) \tag{8}$$

V. Conservative Approach to Perception Modeling

We now construct conservative models of M_{per} using IMDPs to solve Problem 1. Recall that we are given a dataset of system states and perception outputs, denoted as $\mathcal{D} = \{x_i, z_i\}_{i=1,\dots,N}$. With this dataset in hand, our modeling approach consists of three steps, as overviewed in Figure 1:

1. Discretize the continuous state space $\mathcal{X}$ to get state set S, since MDP and IMDP models require discrete states.
2. For each discrete state, compute an interval of detection probabilities using $\mathcal{D}$ and binomial confidence intervals [15]. This is a novel step where we differ from existing perception models [5], [9].
3. Enlarge the detection probability ranges from Step 2 to account for the change in detection probability as the system's (continuous) state $x \in \mathcal{X}$ changes. This is required to fully bridge the gap between the continuous parts of Equation (5) and our discrete models.

A. Step 1: State Space Discretization

The first step is to partition the state space $\mathcal{X}$ into a finite, discrete set of bins, which we denote as S. Each bin $s \in S$ corresponds to region $\mathcal{X}_i \subseteq \mathcal{X}$. With a slight abuse of notation, we say that $x \in s$ when $x \in \mathcal{X}_i$. These bins are intended for building M_{per} and do **not** have to match those of M_{cpl}: we can use non-deterministic transitions to account for states in M_{cpl} that overlap multiple states in M_{per}, and vice versa.

In terms of performing the binning in practice, we see two main ways of doing this. First, one can divide the state space into a set of equally spaced hyper-rectangles (or intervals if the state space is 1D, rectangles in 2D, ...)[3]. In this case, the size of the hyper-rectangle serves as a hyper-parameter of the approach. The second way is to first select the desired number of bins for each dimension of the state space $\mathcal{X}$, which we denote as $\mathcal{b}_1, \dots, \mathcal{b}_n$, and then divide the state space such that each bin has an equal number of datapoints in it. For example, if one has a 1D system and l datapoints (so $|\mathcal{D}| = l$), then one would sort the points $\{x_i, z_i\} \in \mathcal{D}$ by the x_i values and assign each chunk of $\frac{l}{n}$ to its own bin.

The choice of a binning method is subject to important trade-offs. The well-known one is between the granularity of binning and the accuracy/cost of verification. The main hyper-parameter is the number of bins used to partition $\mathcal{X}$. Using a lot of bins to create a fine partition will lead to an M_{per} with a larger state space, which is more expensive to model check. On the plus side, a model with more bins bridges the discrete-continuous gap better, since the behaviors of the perception component change continuously over the state space.

We point out two other, subtler trade-offs at play that our discretization-based approach identifies. The existing perception abstraction approaches [5], [9] fail to account for these trade-offs, since they ignore the inherent uncertainty present in M_{per} — and so they fail to make it conservative. The first tradeoff is that using more bins means that there will be fewer datapoints per bin. This will lead to larger probability intervals for each bin in Step 2 since there are fewer datapoints fed into the binomial confidence intervals. These larger detection probability intervals in turn will lead to a larger interval on the estimated safety probability $\hat{P}(\psi)$. This trade-off captures the intuition that having fewer datapoints to compute the transition probabilities of M_{per} generally results in a less precise model.

Another subtle trade-off relates the bin sizes to the detection probability enlargement in Step 3. Using fewer, larger bins means that the detection probability intervals computed in Step 2 will have to be inflated by a larger amount to account for the perception system's performance changing over the continuous state space $\mathcal{X}$. This trade-off captures the intuition that a model with larger (discrete) bins will generally fail to capture the changes in the performance of the perception component over the continuous state space $\mathcal{X}$, which leads to a less accurate model.

B. Step 2: Bin Probability Computation

The second step of our approach uses the datapoints in $\mathcal{D}$ to estimate the detection probabilities of the perception component h within each bin $s \in S$. First, we denote the datapoints from $\mathcal{D}$ that lie in s as

$$\mathcal{D}_s = \{(x, z) \in \mathcal{D} \mid x \in s\} \tag{9}$$

Just as in prior works [5], [9], we can then compute the empirical detection probability of the datapoints in $\mathcal{D}_s$:

$$\hat{p}_s = \Big(\sum_{(x,z)\in\mathcal{D}_s} \mathbb{1}_{z=1}\Big)/|\mathcal{D}_s| \tag{10}$$

At this step, the previous approaches in the literature would put these estimates as transition probabilities in M_{per}. However, these point estimates of the detection probability are not reliable: they are themselves random values subject to data collection noise, not equal to the true underlying probabilities. Thus, the conservatism of M_{per} is at risk.

[3]This is the approach taken by the previous works [5], [9].

To alleviate this issue, we replace the point estimate $\hat{p}_s$ with a *confidence interval (CI)*, denoted as $[\underline{\hat{p}}_s, \bar{\hat{p}}_s]$ for bin s.[4] There are several methods to obtain a confidence interval for the detection probability of the perception model within the bin, also known as a binomial confidence interval. We employ the Clopper-Pearson confidence interval because its coverage level is never less than the desired nominal coverage [10]. When constructing a confidence interval, one needs to select a confidence level $1-\alpha$. Using a larger value of $1-\alpha$ leads to larger probability intervals, but the proportion of bins in which the confidence interval contains the true detection probability will, in turn, be larger.

We argue that the choice of $1-\alpha$ should be determined by the desired confidence in the model checking result for the overall system M_{sys}. If one wants a confidence of $1-\alpha_{MC}$ that the model checking results are conservative (i.e. Definition 7 holds with probability $1-\alpha_{MC}$) and there are $|S|$ states in M_{per}, then one should set $1-\alpha = 1-\alpha_{MC}/|S|$. Intuitively, this formula captures the fact that each bin's confidence interval has an $\alpha_{MC}/|S|$ probability of not containing the true detection probability, so by applying a union bound [6], the probability of every bin's confidence interval containing the true detection probability is lower bounded by

$$1 - \sum_{s \in S} \frac{\alpha_{MC}}{|S|} = 1 - \alpha_{MC} \tag{11}$$

We will use this union-bound argument when proving the main theoretical result of this paper in Theorem 1.

C. Step 3: Probability Enlargement

As the third and final step, we expand the confidence intervals from Step 2 to account for the fact that the perception model's detection probability can change within each bin. Intuitively, the confidence intervals we computed in the previous step only account for the *average* detection probability within each bin, so we cannot say if it covers *all* the detection probabilities for every system state in the bin. Thus, the argument for the conservatism of M_{per} would be incomplete.

To account for these potential changes in intra-bin detection probabilities, we need a way of inflating the bin confidence intervals based on how much the underlying detection probability changes within each bin. We perform this inflation by estimating the derivative of the detection probability with respect to the system state using a *surrogate model*.

As the surrogate model, we fit a logistic regression model to the entire dataset $\mathcal{D}$. This model takes as input the system state x and outputs the estimated probability that the perception component will detect the object, or $\hat{P}(z = 1 \mid x)$. We denote the function as $LR : \mathcal{X} \to [0, 1]$. Then, we compute how much the estimated detection probability changes within each bin. For bin s, we compute this *probability enlargement factor*:

$$\delta_s^p := \left(\max_{x \in s} LR(x)\right) - \left(\min_{x \in s} LR(x)\right) \tag{12}$$

[4]Using CIs to construct IMDPs has been explored in the areas of synthesis [18] and active learning [2] (which inspires our approach), but not verification.

Now, the updated detection probability interval for bin s is

$$[\underline{\hat{p}}_s - \delta_s^p, \bar{\hat{p}}_s + \delta_s^p] \tag{13}$$

Remark 3. *If we fit a surrogate model to detection probabilities, why not use it to construct M_{per}? We attempted to directly use the logistic regression, along with its associated confidence interval, to estimate the range of true detection probabilities within the bins. However, we found that the logistic regression confidence interval did not empirically achieve the coverage that we wanted. That is because the logistic regression CIs can be viewed as the CIs for the parameters of how to map data onto the space of logistic models — not for the data itself. We demonstrate this in the experiments in Section VII-B and particularly Figure 2.*

VI. Conservative Verification Guarantee

In this section, we state the main theoretical guarantee of this work. Specifically, we show that given the inputs to Problem 1 and an M_{per} constructed using Steps 1–3 from Section V, verifying the resulting composed model M_{abs} will give a conservative estimate of the safety of the true system M_{sys}:

$$P_{\mathsf{M}_{\text{sys}}}(\psi) \in [Pr^{min}_{\mathsf{M}_{\text{abs}}}(\psi), Pr^{max}_{\mathsf{M}_{\text{abs}}}(\psi)] \tag{14}$$

Before the theorem and its proof, we define some notation and a lemma. The proofs of the lemma and theorem can be found in the Appendix of the extended online version [7].

Definition 10 (State-level true detection probability)**.** *Let $P_x(h)$ for $x \in \mathcal{X}$ denote the (ground-truth)* detection probability *of the system's perception component h performing a positive detection when the system M_{sys} is in state x.*

Definition 11 (Bin-level true detection probability range)**.** *Let bin $s_i \in S$ denote a discrete perception bin with corresponding state space region $\mathcal{X}_i \subseteq \mathcal{X}$, as described in Section V-A. Now let $P_{s_i}(h)$ denote the* range of the detection probabilities *of perception h over the states $x \in s_i$:*

$$P_{s_i}(h) = \left[\min_{x \in s_i} P_x(h), \max_{x \in s_i} P_x(h)\right] \tag{15}$$

Lemma 1 (Bin-level detection probability conservatism)**.** *Consider the same inputs as in Problem 1. Let $[\underline{\hat{p}}_{s_i}, \bar{\hat{p}}_{s_i}]$ denote the $1 - \frac{\alpha_{MC}}{|S|}$ confidence interval of the perception component detection probability in bin s_i (as described in Section V-B). Assume*[5] *the probability enlargement factor δ_{s_i} as computed in Equation* (12) *upper-bounds the size of $P_{s_i}(h)$:*

$$\delta_{s_i} \geq \left| \max_{x \in s_i} P_x(h) - \min_{x \in s_i} P_x(h) \right| \tag{16}$$

Then, with probability $1 - \frac{\alpha_{MC}}{|S|}$, the interval from Equation (13) *contains the true detection probability interval $P_{\mathsf{M}_{per}}(s_i)$. Precisely:*

$$\underline{\hat{p}}_{s_i} - \delta_{s_i} \leq \min_{x \in s_i} P_x(h) \text{ and } \bar{\hat{p}}_{s_i} + \delta_{s_i} \geq \max_{x \in s_i} P_x(h) \tag{17}$$

[5]In practice, this assumption means that the true detection probabilities can be instantiated by some parameter values of Step 3's surrogate model, e.g., the logistic regression. In our case studies, this assumption has been largely upheld. Its theoretical investigation is left for future work.

Model Symbol	Model Description
M_{noCI}	Baseline using $\hat{p}_s$ from Equation (10) as detection probabilities
M_{GTPer}	Uses the ground-truth detection probabilities
$\mathsf{M}_{logRegCI}$	Approach using the logistic regression confidence intervals
$\mathsf{M}_{oursNPE}$	Our approach with no probability enlargement
M_{ours}	Our approach: enlargement with derivatives from logistic regression

TABLE I: Perception abstractions in our experiments.

holds with probability $1 - \frac{\alpha_{MC}}{|S|}$.

Theorem 1 (Model-level safety probability conservatism)**.** *Consider the same inputs as in Problem 1, the assumption of Lemma 1, and perception model* M_{per} *with the transition probabilities from Equation* (13) *using confidence level* $1 - \frac{\alpha_{MC}}{|S|}$. *Letting* $\mathsf{M}_{abs} = \mathsf{M}_{cpl} \,||\, \mathsf{M}_{per}$, *it holds that*

$$P\big(P_{\mathsf{M}_{sys}}(\psi) \in [Pr^{min}_{\mathsf{M}_{abs}}(\psi), Pr^{max}_{\mathsf{M}_{abs}}(\psi)]\big) \geq 1 - \alpha_{MC} \quad (18)$$

VII. Case Studies

We demonstrate our approach in the CARLA self-driving simulator [13] on an automatic emergency braking system (AEBS). For a detailed description of the AEBS system, see the extended online version [7]. For IMDP verification, we use PRISM model checker 4.8.1 with native IMDP support.

A. Comparative Abstraction Methods

We compare our approach to a few baseline/ablation methods of creating M_{per} summarized in Table I: a baseline approach [5] that doesn't use confidence intervals, our approach with no probability enlargement (this is our approach, but skipping Section V-C), and an approach that uses confidence intervals from the logistic regression we employ in Section V-C to compute the bin probability intervals (this is our approach, but skipping Section V-B). We expect our approach to be more conservative than all of these due to its guarantees.

B. CARLA case study

Here, we evaluate the feasibility of our approach for a realistic object detection LEC in a high-fidelity simulation. First, we set up the AEBS controller in the autonomous driving simulator CARLA [13], using a red Toyota Prius as the obstacle and a classification variant of Yolov11 as the object detector. Then we performed two rounds of data collection in CARLA: (i) constant-v runs to gather image-distance data to build our perception abstractions, (ii) AEBS-controlled runs to approximate the true safety probability of the AEBS controller.

For the constant-v runs, we ran 500 trials with a speed of 6m/s and initial distance around 210m, making 181000 perception datapoints. This data leads to 5m-wide binned estimates of the YoloNet detection chances in Figure 2. For the AEBS-controlled runs, we ran 500 trials with the initial distance 50m and speed 20m/s. This resulted in 330 crashes, yieldinga $[0.62, 0.70]$ 95%-CI for the true collision chance.

We used the perception data to build models: using our approach M_{ours}, our approach w/o probability enlargement

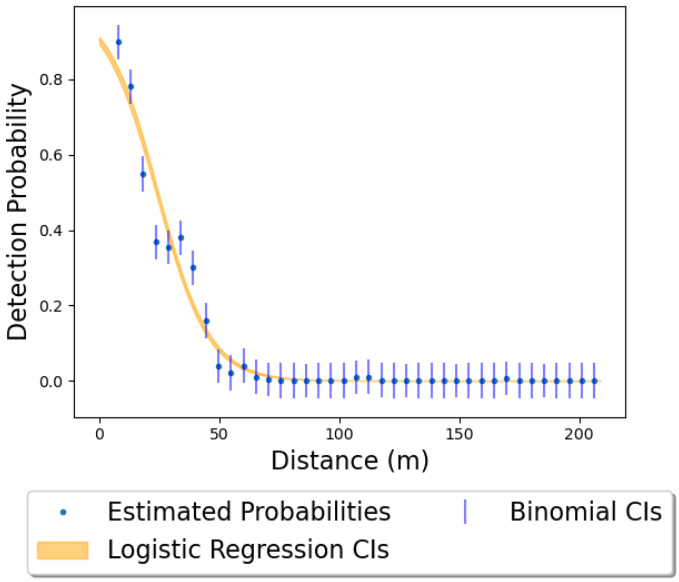

Fig. 2: Detection probabilities vs. distance to obstacle for our Yolov11 LEC in CARLA case study.

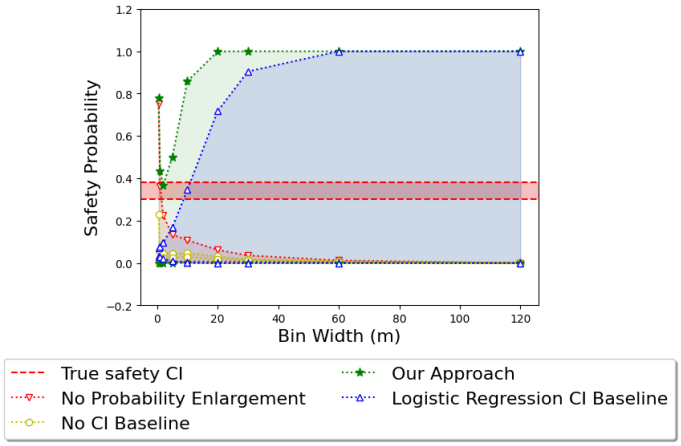

Fig. 3: Safety probability estimates over a range of bin widths in CARLA case study.

$\mathsf{M}_{oursNPE}$, the approach from [5] w/o confidence intervals M_{noCI}, and the approach with the logistic regression confidence intervals $\mathsf{M}_{logRegCI}$. Note that we cannot build M_{GTPer} since we do not know the ground truth detection probabilities.

We first vary the bin width parameter used to form the models to determine how it affects the estimated safety probability intervals. The results are shown in Figure 3. Note that for every bin width the safety probability intervals from our approach contain the confidence interval for the true underlying safety probability. Also, the probabilities from $\mathsf{M}_{oursNPE}$ are only empirically valid for small bins. Additionally, the probabilities from M_{noCI} never contain the true safety probability. Finally, the safety probabilities from $\mathsf{M}_{logRegCI}$ are not valid for small bin widths. This highlights the drawbacks of solely relying on logistic regression to estimate detection probabilities.

VIII. Conclusion and Future Work

This paper has presented a novel method for constructing provably conservative probabilistic abstractions of perception and sensing components in closed-loop autonomous systems. Our abstraction method combines (i) binomial confidence intervals to account for the inherent uncertainty of data-driven estimation and (ii) logistic regression to account for the perception behaviors changing continuously in the system's state space. We prove that when our perception/sensing abstractions are composed with conservative control and dynamics models, the resulting model will lead to conservative verification with high probability. We demonstrate the conservatism of our approach in case studies of an AEBS system with both synthetic perception models and in CARLA using a realistic object detector. Future work can focus on reducing conservatism by better estimating intra-bin changes in detection probabilities of the perception component. Additionally, we will investigate more expressive M_{per} models, such as conditioning detections on previous M_{per} outputs, and explore abstractions for multi-class LECs.

Acknowledgements

The authors thank Edgar Dobriban for discussing uncertainty quantification for logistic regression. This work was supported in part by NSF grants CNS 2143274, CCF 2403616, and ARO MURI W911NF-20-1-0080. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF), the Army Research Office (ARO), or the US Government.

References

[1] Ajeleye, D., Lavaei, A., Zamani, M.: Data-driven controller synthesis via finite abstractions with formal guarantees. IEEE Control Systems Letters **7**, 3453–3458 (2023)

[2] Alasmari, N., Calinescu, R., Paterson, C., Mirandola, R.: Quantitative verification with adaptive uncertainty reduction. Journal of Systems and Software **188**, 111275 (Jun 2022)

[3] Antonante, P., Nilsen, H.G., Carlone, L.: Monitoring of perception systems: Deterministic, probabilistic, and learning-based fault detection and identification. Artif. Intell. **325**(C) (Dec 2023)

[4] Badings, T., Romao, L., Abate, A., Jansen, N.: Probabilities Are Not Enough: Formal Controller Synthesis for Stochastic Dynamical Models with Epistemic Uncertainty. In: Proc. of AAAI 2023 (2023)

[5] Badithela, A., Wongpiromsarn, T., Murray, R.M.: Evaluation Metrics of Object Detection for Quantitative System-Level Analysis of Safety-Critical Autonomous Systems. In: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE (2023)

[6] Boole, G.: The mathematical analysis of logic. CreateSpace Independent Publishing Platform (1847)

[7] Cleaveland, M., Lu, P., Sokolsky, O., Lee, I., Ruchkin, I.: Conservative perception models for probabilistic verification (2025), https://arxiv.org/abs/2503.18077

[8] Cleaveland, M., Ruchkin, I., Sokolsky, O., Lee, I.: Monotonic Safety for Scalable and Data-Efficient Probabilistic Safety Analysis. In: ACM/IEEE 13th Intl. Conf. on Cyber-Physical Systems (ICCPS). pp. 92–103 (May 2022)

[9] Cleaveland, M., Sokolsky, O., Lee, I., Ruchkin, I.: Conservative Safety Monitors of Stochastic Dynamical Systems. In: Proc. of the NASA Formal Methods Conference (May 2023)

[10] Clopper, C.J., Pearson, E.S.: The use of confidence or fiducial limits illustrated in the case of the binomial. Biometrika **26**(4), 404–413 (1934)

[11] Corso, A., Katz, S., Innes, C., Du, X., Ramamoorthy, S., Kochenderfer, M.J.: Risk-driven design of perception systems. Advances in Neural Information Processing Systems **35**, 9894–9906 (2022)

[12] Delahaye, B., Larsen, K.G., Legay, A., Pedersen, M.L., Wasowski, A.: Decision problems for interval markov chains. In: Language and Automata Theory and Applications: LATA 2011. Springer

[13] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V.: CARLA: An Open Urban Driving Simulator. In: Proceedings of the 1st Annual Conference on Robot Learning. pp. 1–16 (2017)

[14] Dreossi, T., Fremont, D.J., Ghosh, S., Kim, E.J., Ravanbakhsh, H., Vazquez-Chanlatte, M., Seshia, S.A.: Verifai: A toolkit for the formal design and analysis of artificial intelligence-based systems. In: International Conference on Computer Aided Verification (2019)

[15] Fleiss, J., Levin, B., Paik, M.: Statistical Methods for Rates and Proportions, Third Edition, pp. 187 – 233 (01 2004)

[16] Ivanov, R., Carpenter, T., Weimer, J., Alur, R., Pappas, G., Lee, I.: Verisig 2.0: Verification of Neural Network Controllers Using Taylor Model Preconditioning. In: Computer Aided Verification. pp. 249–262. Springer International Publishing, Cham (2021)

[17] Ivanov, R., Carpenter, T.J., Weimer, J., Alur, R., Pappas, G.J., Lee, I.: 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 (HSCC). Association for Computing Machinery (2020)

[18] Jiang, J., Zhao, Y., Coogan, S.: Safe Learning for Uncertainty-Aware Planning via Interval MDP Abstraction. IEEE Control Systems Letters **6**, 2641–2646 (2022)

[19] Jonsson, B., Larsen, K.: Specification and refinement of probabilistic processes. In: Proceedings Sixth Annual IEEE Symposium on Logic in Computer Science. pp. 266–277 (Jul 1991)

[20] Joshi, K., Hsieh, C., Mitra, S., Misailovic, S.: Gas: Generating fast accurate surrogate models for simulations of autonomous vehicle systems. In: 2024 IEEE 35th International Symposium on Software Reliability Engineering (ISSRE). pp. 260–271 (2024)

[21] Katz, S.M., Corso, A.L., Yel, E., Kochenderfer, M.J.: Efficient determination of safety requirements for perception systems. In: 42nd Digital Avionics Systems Conference (DASC) (2023)

[22] Kim, E., Gopinath, D., Pasareanu, C., Seshia, S.A.: A programmatic and semantic approach to explaining and debugging neural network based object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11128–11137 (2020)

[23] Kupferman, O., Vardi, M.Y.: Model checking of safety properties. Formal methods in system design **19**, 291–314 (2001)

[24] Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of Probabilistic Real-Time Systems. In: Computer Aided Verification. Springer (2011)

[25] Kwiatkowska, M., Norman, G., Parker, D.: Probabilistic Model Checking and Autonomy. Annual Review of Control, Robotics, and Autonomous Systems **5**(1), 385–410 (2022)

[26] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards Deep Learning Models Resistant to Adversarial Attacks (Feb 2018)

[27] P. Antonante, S. Veer, K.L.X.W.L.C., Pavone, M.: Task-aware risk estimation of perception failures for autonomous vehicles. In: Proceedings of Robotics: Science and Systems (2023)

[28] Păsăreanu, C.S., Mangal, R., Gopinath, D., Getir Yaman, S., Imrie, C., Calinescu, R., Yu, H.: Closed-loop analysis of vision-based autonomous systems: A case study. In: International conference on computer aided verification. pp. 289–303. Springer (2023)

[29] Pnueli, A.: The temporal logic of programs. In: 18th Annual Symposium on Foundations of Computer Science, 1977. pp. 46–57 (Oct 1977)

[30] Richter, C., Roy, N.: Safe Visual Navigation via Deep Learning and Novelty Detection. MIT Web Domain (Jul 2017)

[31] Ruchkin, I., Cleaveland, M., Ivanov, R., Lu, P., Carpenter, T., Sokolsky, O., Lee, I.: Confidence Composition for Monitors of Verification Assumptions. In: ACM/IEEE 13th Intl. Conf. on Cyber-Physical Systems (ICCPS). pp. 1–12 (May 2022)

[32] Santa Cruz, U., Shoukry, Y.: NNLander-VeriF: A Neural Network Formal Verification Framework for Vision-Based Autonomous Aircraft Landing. In: NASA Formal Methods. pp. 213–230. Lecture Notes in Computer Science, Springer International Publishing, Cham (2022)

[33] Wooding, B., Lavaei, A.: Impact: Interval mdp parallel construction for controller synthesis of large-scale stochastic systems. In: Quantitative Evaluation of Systems and Formal Modeling and Analysis of Timed Systems. Cham (2024)