Probabilistic Collision Risk Estimation for Autonomous Driving: Validation via Statistical Model Checking

Anshul Paigwar[1], Eduard Baranov[2], Alessandro Renzaglia[1], Christian Laugier[1] and Axel Legay[2]

***Abstract*— A crucial aspect that automotive systems need to face before being used in everyday life is the validation of their components. To this end, standard exhaustive methods are inappropriate to validate the probabilistic algorithms widely used in this field and new solutions need to be adopted. In this paper, we present an approach based on Statistical Model Checking (SMC) to validate the collision risk assessment generated by a probabilistic perception system. SMC represents an intermediate between test and exhaustive verification by relying on statistics and evaluates the probability of meeting appropriate Key Performance Indicators (KPIs) based on a large number of simulations. As a case study, a state-of-the-art algorithm is adopted to obtain the collision risk estimations. This algorithm provides an environment representation through Bayesian probabilistic occupancy grids and estimates positions in the near future of every static and dynamic part of the grid. Based on these estimations, time-to-collision probabilities are then associated with the corresponding cells. Using CARLA simulator, a large number of execution traces are then generated, considering both collisions and almost-collisions in realistic urban scenarios. Real experiments complete the analysis and show the reliability of the simulation results.**

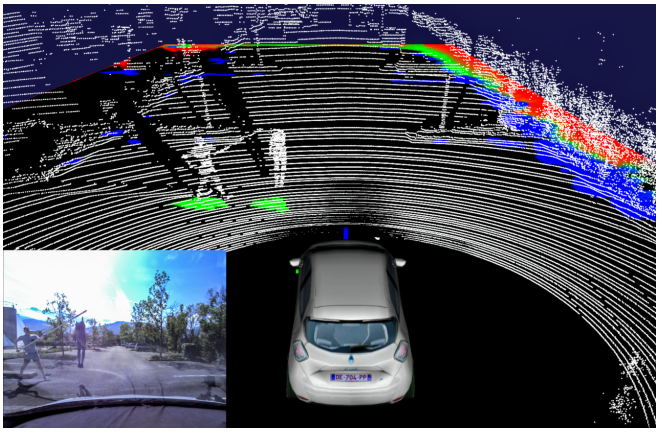


Fig. 1. Reproduction of a pedestrian crossing scenario using a mannequin for validation of collision risk estimation. The environment around the car is reproduced based on a 64-layered LiDAR and a dense probabilistic occupancy grid output from the CMCDOT framework. Colors represent different states for each cell in the grid: occupied static (blue), occupied dynamic (green), empty (black), unknown area (red).

I. Introduction

In the last years, more and more solutions for Automated Cyber-Physical Systems (ACPS) are based on probabilistic algorithms. Perception systems in intelligent vehicles [1] [2] dealing with dynamic and unexpected scenarios is one example where the use of probabilistic approaches becomes essential to accurately confront the uncertainties in the environment. Moreover, there is an increasing demand for regulating and validating intelligent vehicle systems to build public trust in their use. Yet, the analysis of safety and reliability poses a significant challenge due to the inevitable uncertainties on the roads. Due to the stochasticity of ACPSs, involving multiple states and transitions between them, validation through standard approaches is usually not viable.

More recently, researchers are exploring formal verification techniques for Probabilistic Model Checking [3]. Zao *et al.* have demonstrated the use of the PRISM model checker for the verification of autonomous robotic systems [3] [4]. Althoff *et al.* presented an online formal verification method based on reachability analysis to guarantee safety, when planing the trajectory of the automated road vehicles [5]. Inherently, the complexity of formal methods increases exponentially with the increase in the states of the system. In the context of intelligent vehicle systems where states (e.g. estimation of collisions) change and evolve every time step, exhaustively checking such properties is usually not affordable in many scenarios because of time, complexity and costs constraints. To overcome these limitations in highly dynamic systems, Statistical Model Checking (SMC) has been proposed for the evaluation of complex probabilistic frameworks [6]. In the autonomous driving domain, a recent work [7] presented a validation scheme based on SMC to evaluate a decision-making approach.

In this paper, we propose an SMC-based validation scheme to analyze the collision risk prediction provided by a probabilistic perception system. Risk assessment and Time-To-Collision (TTC) prediction are crucial information that every autonomous driving system needs for safe and acceptable navigation [8]. To generate such estimation, we adopt a state-of-the-art Bayesian perception algorithm, the Conditional Monte Carlo Dense Occupancy Tracker (CMCDOT) [9]. The CMCDOT solves this challenging problem with a grid based approach which avoids the pass through multi-object detection and tracking, and provides collision probabilities for

[1] Univ. Grenoble Alpes, Inria, 38000, Grenoble, France; e-mail: firstname.lastname@inria.fr

[2] Université Catholique de Louvain, Computer Science Department, B-1348 Louvain-la-Neuve, Belgium; e-mail: firstname.lastname@uclouvain.be

Published in: 2020 IEEE Intelligent Vehicles Symposium (IV)

DOI: 10.1109/IV47402.2020.9304821

©2020 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes,creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

each cell in the grid [10]. The validation method makes use of specifically defined Key Performance Indicators (KPIs), expressed as temporal properties depending on a set of identified metrics. By analyzing the behavior of these metrics through both real experiments and numerous simulations in realistic environments, a probability for the system to finally respect the KPIs is provided by the model checker.

A particular focus has been given on intersection crossing scenarios, which are among the most dangerous parts in road networks, with more than 8% of the total road fatalities in Europe [11]. Realistic simulations of collisions and risky situations in intersection crossing scenarios have been carried out in the CARLA simulator [12] and real experiments conducted with our autonomous car (Fig. 1). The traces of the conducted experiments are then analyzed with a PLASMA Lab statistical model checking platform [13]. PLASMA Lab has a modular design allowing the use of various simulators and model checkers. The analysis of the execution traces w.r.t. KPIs formalized as Bounded Linear Temporal Logic formulas gives the final evaluation of the collision risk prediction.

The key contributions of this study can be summarized as follows:

- We present an SMC-based approach for statistical validation of probabilistic collision risk estimation for automated driving systems.
- As a case study, we used CMCDOT to obtain collision risk estimations and show how our SMC-based approach can be adapted to validate probabilistic algorithms.
- Using CARLA simulator, we show how simulations can be parameterized to generate large numbers of execution traces reproducing realistic collisions and almost-collisions scenarios for statistical analysis.
- We perform real experiments and quantitative comparisons to prove the integrity of our simulation-based approach.

II. Statistical Model Checking

Reasoning about ACPS with exhaustive techniques is usually infeasible as they require checking the desired property in all reachable states. Due to the complex nature of ACPS and bad scalability of methods, the validation is only possible at a very abstract level. Furthermore, probabilistic methods are more suitable to validate stochastic algorithms than conventional ones [14].

Statistical Model Checking (SMC) [15], [16] is a simulation-based technique relying on statistics that can be placed between testing and exhaustive verification. It answers the question of whether the property is satisfied with a given confidence level. SMC requires an executable stochastic model of a system. Some of the transitions of the stochastic model are governed by probabilistic choices, therefore executions of the model do not follow a single trace. SMC checks the desired property on multiple traces and returns the probability of the property to be satisfied. Note that the property must be decidable on a finite trace.

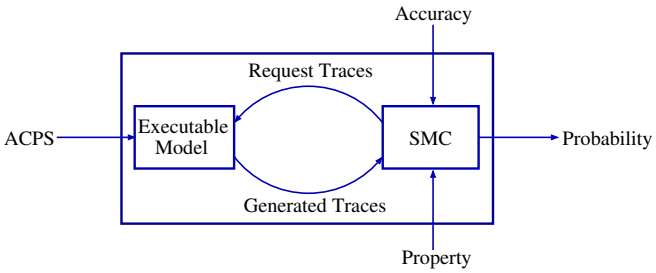


Fig. 2. An overview of the Statistical Model Checking approach.

An overview of the approach is shown in Figure 2. The SMC algorithm takes a formalized property and the desired accuracy. At each step, it requests a simulator to generate several traces from the executable model. The property is checked against each trace. If there is enough information to find a result with the required accuracy the algorithm terminates, otherwise the cycle repeats.

To give an example, Monte-Carlo algorithm can be considered as an SMC approach. It checks the property over N random independent executions, and returns the proportion $\hat{p}$ of executions satisfying the property.

$$\hat{p} = \frac{1}{N}\sum_{1}^{N} f(ex_i) \text{ where } f(ex_i) = \begin{cases} 1 & \text{if } ex_i \models P \\ 0 & \text{otherwise} \end{cases}$$

The number of executions defines the precision of the answer, which can be evaluated with a Chernoff bound:

$$Pr(|p - \hat{p}| \leq \epsilon) \geq 1 - \delta$$

where p is the value of the probability we want to evaluate and $\hat{p}$ is the estimation we compute. The formula states that the estimation error $|p-\hat{p}|$ is bounded by ϵ with a probability $1-\delta$. In other words, the probability that the error in the estimation is greater than ϵ is δ. By fixing ϵ and δ, the necessary number of executions N can be computed. For a good precision, both ϵ and δ shall be small, but in this case the required number of executions N would be high.

Formal analysis requires properties to be expressed with a formal semantics. ACPS by their nature evolve during the execution, their traces can be considered as sequences of system states, one state for each simulation step, and the metrics are evolving when the execution moves to the next state. Temporal formulas are a formalism allowing to specify properties or KPIs keeping in account the evolution of a system. In particular, we rely on Bounded Linear Temporal Logic (BLTL), a bounded version of LTL [17]. Properties are specified in BLTL with logical operators and bounded versions of temporal operators. The former ones are used to express a predicate on the current state and the latter ones can define properties about bounded period of time. In this paper we use the following temporal operators: *always* ($G_{\leq t}\ \phi$) requiring ϕ to hold in all states for the next t units of time and *eventually* ($F_{\leq t}\ \phi$) requiring ϕ to become true within t units of time. A full definition of BLTL and its semantics can be found in [18].

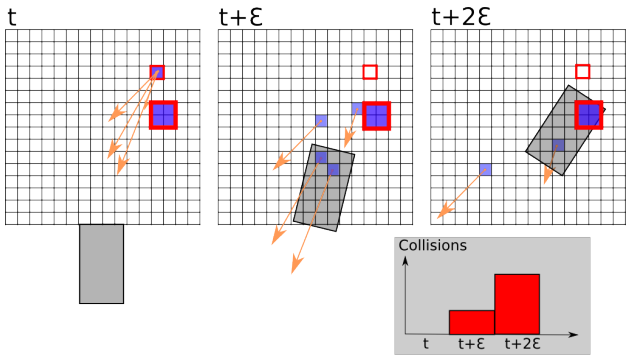


Fig. 3. Collision risk estimation of a specific cell. Both the future cell and the ego-vehicle positions are predicted according to their estimated velocity. The risk profile is computed for every cell, and then used to integrate over time the total collision risk.

III. Probabilistic Collision Risk Evaluation

The collision risk evaluation is one of the most crucial aspects of an autonomous driving system to ensure the safety of drivers, passengers, and other road users. For this reason, a thorough validation of this estimation is paramount.

Most of the existing risk estimation methods consist of detecting and tracking dynamic objects in the scene [19], [20]. The simplest solution is then to estimate a Time to Collision (TTC) by projecting the ego-vehicle and the trajectories of other moving objects in the future [21], [22]. The limitation of such approaches is that detection of objects, either by classical model fitting or deep-learning-based approaches [23], requires a predefinition of the object's shape. As the number and types of objects increase, detection and tracking become difficult and costly problems to solve.

An alternative approach is provided by grid-based solutions, where the estimation of the collision risk can be computed for each cell regardless of the object it belongs to. This grid-based approach is for instance used in the Conditional Monte Carlo Dense Occupancy Tracker (CMCDOT) [9]. Directly estimating the velocity of each cell in the grid, this algorithm predicts the position in the near future of every cell as well as the trajectory of the ego-vehicle. These estimations are computed over short periods, and a probability of potential collision is provided. TTC probabilities are then associated to the cell from which the colliding element came from (Fig. 3). In CMCDOT, the TTC probabilistic estimations are discretized corresponding to collisions within 1, 2 and 3 seconds. This strategy, originally presented in [10], enjoys the advantage of being model-free, avoids solving the complex problem of multi-object detection and tracking, while integrating the totality of the available information and providing a probabilistic estimation of the risk associated to each part of the scene. The result of this estimation can then be used as the starting point for any control and decision-making system. Though in [10] the authors have presented an evaluation of their approach, a proper validation is absent and poses a big challenge.

More generally, the CMCDOT is a generic spatial occupancy tracker, which provides a dense and generic representation of the environment [24], [25] through a probabilistic occupancy grid (see Fig. 4), based on Bayesian fusion, filtering of sensor data and Bayesian inference. It infers the dynamics of the scene via a hybrid representation, representing the environment with static and dynamic occupancy, free spaces and unknown regions. This differentiation enables the use of state-specific models, such as classic occupancy grids for static components and sets of moving particles for dynamic occupancy, as well as confidence estimation and management of areas with no information.

It is, however, worth noticing that the validation scheme proposed in this paper is not specific for this algorithm but it could instead be adopted to validate the collision risk estimation output by any other probabilistic solution.

A. SMC validation

To perform SMC validation, it is necessary to have: *i)* an executable stochastic model; *ii)* appropriate KPIs related to the system and scenarios under test expressed in the formal semantics.

For ACPS and intelligent vehicles, in particular, simulators are often used as an executable model. Simulators can allow having a realistic representation of ACPS as well as of the environment outside of the system under validation. By modifying the initial conditions of simulations, different executions can be easily obtained.

The collision risk estimation involves both the detection of objects in a grid and the estimation of their velocity. As a KPI, we decided to evaluate the collision risk within several seconds to a potential collision. The validation involves both positive cases (the collision is real unless actions are taken) and negative cases (the collision is not going to happen). The informal description of the KPI is that the high probability of collision predicted by a perception system should be followed by a real collision, while low probability of collision should guarantee its absence. For the formal description we use the following parameters and variables:

- The seconds i before the potential collision at the time we evaluate the collision risk;
- $cmcdot_risk_i$ is the maximum probability of collision in i seconds output by CMCDOT among all cells belonging to the approaching vehicle;
- The thresholds τ_h and τ_l are the boundaries when the collision risk is considered high and low, respectively;
- The number L of states in the longest trace;
- t is used to specify the interval within which the $cmcdot_risk$ is considered;
- $collided$ is $True$ in all states after a collision has occurred and $False$ otherwise.

We consider two KPIs formalized as follows:

- $G_{\leq L}\ ((F_{\leq t}\ collided) \Rightarrow (cmcdot_risk_i > \tau_h))$. This property states that, at any time, a collision happening within t seconds must be perceived by CMCDOT with a high collision risk. Any violation of this property for $t \leq i$ falls into false negative case: a real collision is not perceived in time.

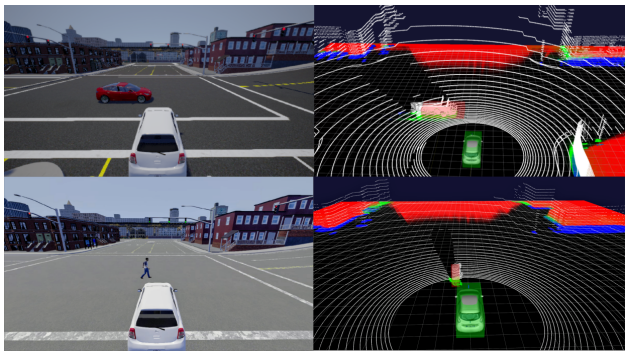


Fig. 4. Left: Simulated scenarios in CARLA with the ego-vehicle (white) colliding with another vehicle (red) and with a pedestrian crossing the road; Right: Simulated LiDAR sensor and corresponding dense occupancy grid output from CMCDOT.

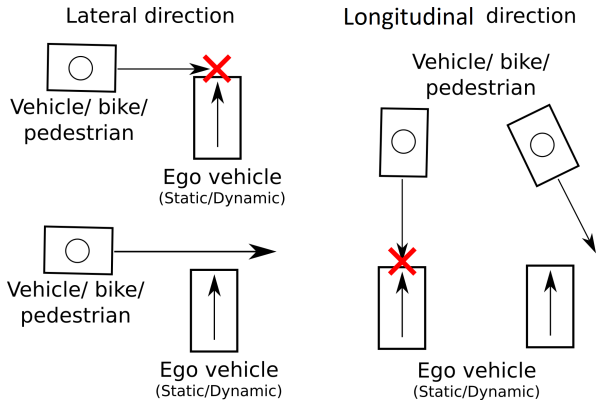


Fig. 5. Different sets of scenarios for collision and non-collision with ego-vehicle recreated in simulation and tested in real experiments.

- $G_{\leq L}\ ((G_{\leq t}\ \neg\, collided) \Rightarrow (cmcdot_risk_i < \tau_l))$. This property states that the collision risk estimation must be low in all states such that no collision will happen in t seconds. A violation of this property for $t \geq i$ falls into false positive case: a non-existent collision is predicted.

The CMCDOT framework that we consider as a case study outputs the collision risk as a set of probability of collision within 1, 2 and 3 seconds for each cell of the grid. Further, we adapt our approach to evaluate the CMCDOT based on this set of probabilities.

IV. Experimental Setup

A. Simulation for perception

In the experiments, the validation methodology consists of the following steps: *i)* for the chosen scenarios we make multiple simulations and store traces with the information from simulator as well as collision risks output by CMCDOT; *ii)* KPIs are checked on recorded traces with PLASMA Lab; *iii)* depending on the required precision, either more traces are requested from the step *i*, or the KPIs satisfaction probability is returned.

Simulations play an important role in our analysis since they allow us to consider dangerous scenarios not reproducible in reality, such as collisions, and to easily generate large numbers of trials for every considered scenario, which is hard with real experiments. To have a realistic reproduction of the real platform, and in particular of its perception system, we rely on CARLA, an open urban driving simulator [12]. CARLA provides digital assets (urban layouts, buildings, vehicles, and pedestrians) allowing for a realistic representation of real-world scenarios. It also provides different sensor models that can be configured and Gaussian noise can be added to match with actual system uncertainties. The ego-vehicle, as well as other road users, can easily be controlled and given desired trajectories (see Fig. 4).

The CMCDOT requires two types of sensors: LiDARs, which generate point clouds by measuring the time a beam of light takes to return after hitting an obstacle in the path, and wheel odometer. In comparison to other sensors (e.g. cameras), simulated LiDAR data are very similar to real data [26]. In the simulation, we appropriately position the LiDAR on the ego vehicle, with the same data format and the same sampling frequency as the physical sensor on our experimental platform (Renault Zoe).

We developed a parameter-based approach through which the non-ego vehicle class (car, motorcycle or pedestrian) and initial positions and speeds of ego and the non-ego vehicle are specified. These parameters are set accordingly upon request from the SMC model checker. In this way a large number of simulated scenarios are generated and tested in a looped manner as shown in Fig. 2. The strong advantage of this approach is the ease with which a large number of simulated scenarios can be generated, run and analyzed.

B. Scenario description and trace details

Our simulated scenarios aim at checking the collision risk estimated by an automated driving system at four-way crossroads. For simplicity of evaluation, we assume that there is only one non-ego vehicle present at the crossroad. The ego-vehicle and non-ego vehicle both start at the same time from a given location in the CARLA environment. A PID controller maintains constant velocity for both vehicles while the steering angle is fixed. In our study, we consider the vehicles colliding either laterally or longitudinally (Fig. 5). A scenario ends when the cars collide or the ego-vehicle passes the intersection without any collision.

It is important to remark that these simplifications in the scenario definition do not deteriorate the validation results for our case study. Regarding the angle of collision, CMCDOT relies only on 360-degree LiDAR's data for collision risk estimation, and the sensor model of LiDAR inherently makes the direction of approach to other objects a trivial problem. Similarly, as CMCDOT uses a grid-based approach, there is no distinction between different objects, but only their occupancy information is considered. As a result, the number of objects in a scene also becomes an irrelevant factor. The behavior estimation for a given object is also the same regardless of the number of objects as these estimations are computed at a cell-level and not for specific objects.

When the non-ego vehicle enters in the proximity of the intersection, represented as a grid of size (56m, 56m), the system starts recording the simulation trace. A given trace consists of a sequence of states of the simulation for each recorded timestamp. Each state contains the following information: timestamp (ms), ego vehicle velocity (m/s), non-ego vehicle velocity (m/s), max probability of collision in 1 sec, 2 sec and 3 sec, location of ego vehicle (x,y), location of non-ego vehicle (x,y), if-collision (boolean).

C. Real-world experiments

In the real world, it is infeasible to generate a statistically significant number of traces to evaluate the KPIs. We can, however, analyze how close the simulation traces are to real experiments. We collect several real traces by imitating the collision of the ego-vehicle (equipped Renault Zoe) with a pedestrian (by using a mannequin as in Fig. 6) and with another vehicle (by throwing a big ball). In these experiments we do not have access to real non-ego vehicles velocities and positions, therefore we record only timestamps, CMCDOT risk values and the ego-vehicle speed. The real-world traces are then compared with simulated ones of analogous scenarios, where the ego-vehicle's speed is in the range $[v-0.5m/s, v+0.5m/s]$, with v being the real car's speed. To compare the risk evolution over time, we use a partial curve mapping metric (PCM) [27] as a measure of similarity. The PCM maps each data point of one curve onto another and the similarity estimates the volume between mapped curves as $s = \Sigma_{i=1}^{n-1}((d_i + d_{i+1}) * l_i/2)$, where n is the number of data points on the curve, d_i is the distance between the i^{th} points of mapped curves, and l_i is a relative length of the i^{th} curve segment. Curve mapping makes this metric unaffected by having a separate point at timestamps that may not coincide in different traces. Also for traces of different length, PCM chooses an offset that gives the best mapping of curves.

V. Validation Results

For the KPIs defined in Section III-A, we have fixed $L = 200$, since all collected traces has less than 200 states, and the thresholds τ_h and τ_l at 75% and 50% respectively. We have performed several experiments to determine reasonable values for the thresholds and noticed that they can be chosen from a large range without considerable impact on the results. For almost all traces, when no collision is expected, collision risk is below 20%, and after the detection of the collision the risk is quickly raised above 80%. We keep rather high value for the low collision risk threshold in order to mitigate single peaks when the CMCDOT risk raises above 30% at a single point of time. The accuracy was set by $\epsilon = \delta = 0.05$ in order to have an estimation error below 5% with a probability 95%. To achieve this accuracy we needed approximately 750 simulations for each scenario. The evaluation of each scenario took approximately 2 hours spending most of the time on trace generation.

In order to study the dynamics of the collision risk, we evaluate the KPIs for various values of t. In practice, for the risk estimation in i seconds we consider a 2-second interval $[i-1, i+1]$ for t. We expect to have a high CMCDOT risk if $t < i$ and low CMCDOT risk otherwise. Thus, the first KPI, stating that a collision happening within t seconds implies high CMCDOT risk, should be satisfied for values $t < i$. For the evaluation of this property we used t in the interval $[i-1, i]$. On the contrary, the second KPI, stating that no collision in less than t seconds implies low CMCDOT risk, should be satisfied for values $t > i$. We used $t \in [i, i+1]$ for the second property.

For our experiments, we have taken into account cars, bicycles, and pedestrians. The results do not present any significant difference depending on the chosen class, showing that CMCDOT can equally detect small vehicles (bicycles and pedestrians) and larger ones (cars). In the following, we provide combined results from all these classes of vehicles.

Fig. 7 shows the KPIs evaluation - the probability of a property to be satisfied - depending on the parameter t for traces corresponding to a static ego vehicle. In the top plot, one can notice that the rise of the collision risk is often delayed: only at $t = 0.6$ seconds, all traces have high collision risk in 1 second. The bottom plot shows that in case of no expected collision in 1, 2 or 3 seconds, the CMCDOT algorithm rarely raises the corresponding collision risk, i.e. no false-positive predictions of collisions or very early predictions. The experiments also show that whenever a collision is predicted, the collision risk does not drop afterward. This fact is partially reflected by monotonicity of curves on the plot.

Fig. 8 shows the results when the ego vehicle is moving. As expected, the 1-second prediction is better than 2 or 3 seconds predictions. We can see that the correct prediction rate dropped to 95% for the 1-second risk estimation and to 90% for 2-seconds estimation. The second KPI has also shown a worse prediction. Investigation of the traces showed that the drop of precision is mostly caused by border-case scenarios when vehicles barely miss or barely touch each other. These cases are hard to predict since any minor imprecision of perception can cause a wrong prediction. Also, few traces have drops of collision risk for 2 and 3 seconds in isolated points.

To summarize, the collision prediction is accurate in case of a static ego vehicle independently of the size of the other vehicle, though there is a delay in risk raising. In the case of a moving ego vehicle, there are simulations that are not predicted correctly, nevertheless the 1-second prediction results correct in 95% of cases.

A. Evaluation of real-world experiments

The traces recorded during the real experiments have been finally compared with the ones corresponding to similar scenarios simulated in CARLA. While recreating real tests with clear collisions or no-collisions scenarios is easy, the near-miss scenarios are more challenging to reproduce. These cases are much more sensitive to several factors, such as changes in the velocities of the vehicles, and we were not able to find a reliable way to conduct experiments clearly

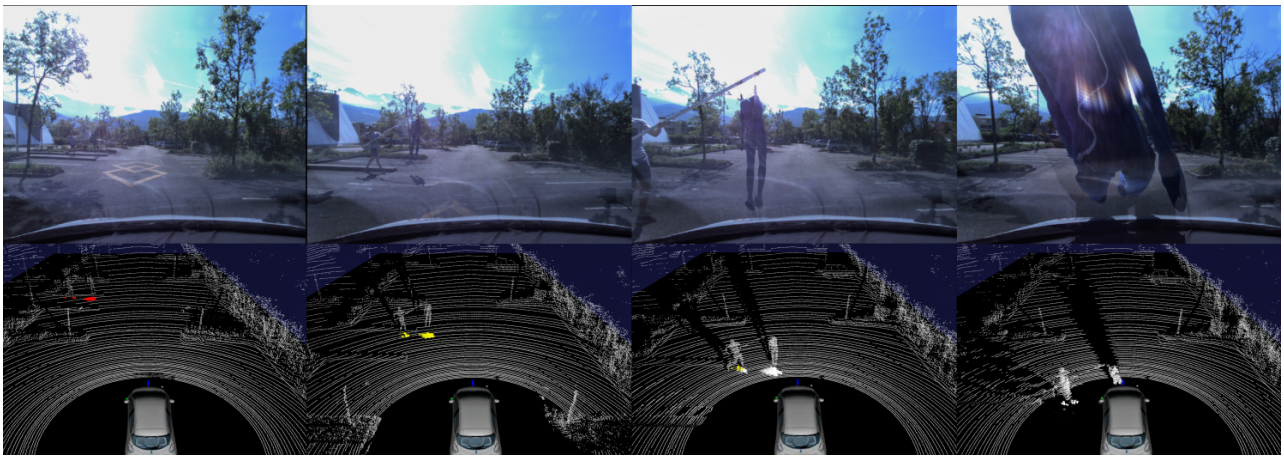


Fig. 6. Illustration of one of the experiments performed with INRIA's autonomous car. The scenario reproduces the car colliding with a pedestrian (mannequin) crossing the street. Top: camera view from the car. Bottom: the environment as seen by the LiDAR and the CMCDOT collision risk grid. The sequence from left to right represents different time stamps. Colors represent probability of collision in $3s$ (red), $2s$ (yellow) and $1s$ (white).

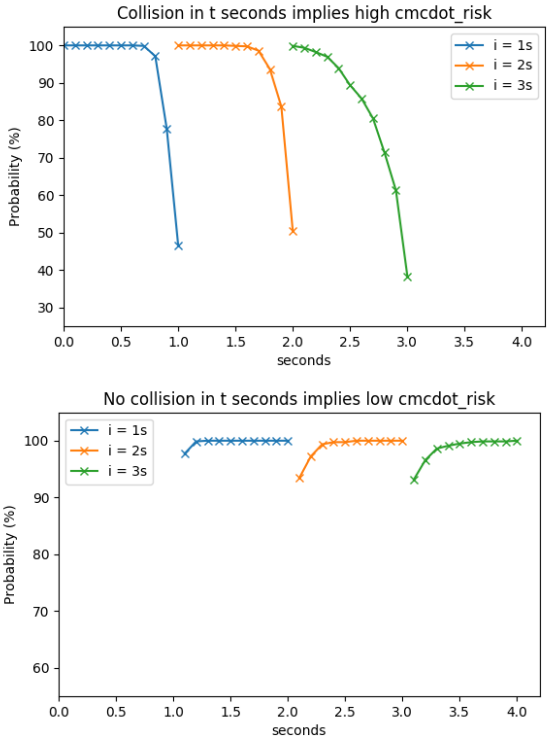


Fig. 7. Evaluation of KPIs with a static ego vehicle. X-axis represents the KPI parameter t - the interval within which the $cmcdot_risk$ is considered. Y-axis shows the statistical probability of the KPI to be satisfied.

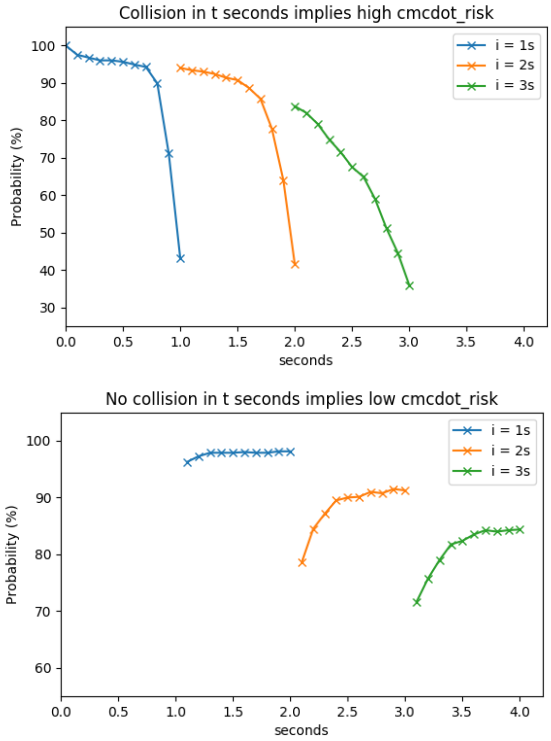


Fig. 8. Evaluation of KPIs with a moving ego vehicle. Axes as in Fig. 7.

matching simulated scenarios. For this reason, in this work, we do not consider this class of scenarios for our analysis.

For all except one real-world traces, we were able to obtain a PCM metric below 1 with respect to a simulated trace. Fig. 9 shows the evolution of the collision risk for the real trace and the simulated one for an instance where all the three pairs of curves for 1, 2 and 3 seconds had low PCM metric, specifically below 1 for the 1-second curves and below 2.5 for 2 and 3 seconds predictions. Considering all the possible sources of noise and instability in velocities, a perfect match between the two curves cannot be expected. However, the obtained results show that in most of the cases the collision risk estimated by CMCDOT over time in simulation closely reproduced the real world. Another relevant discovery is that in the real-world traces, the risk raise happens earlier than in simulated traces. This could suggest that the risk raise delay noticed in simulated traces might be exclusively caused by delays in the simulator.

VI. Conclusions

In this paper, we tackled the challenging problem of validating probabilistic algorithms for intelligent vehicles to

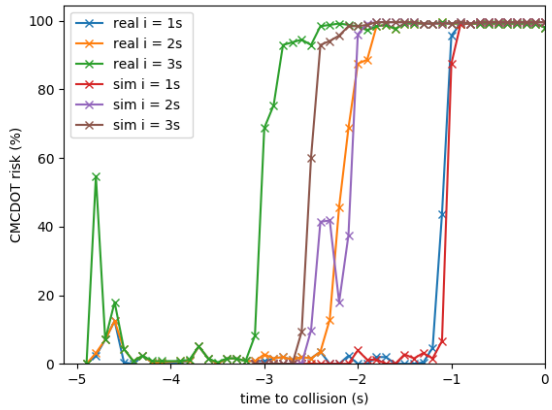


Fig. 9. Comparison of real-world and simulated estimations. Curves show CMCDOT risk in i seconds for real and simulated traces.

ensure safety in autonomous driving scenarios. In particular, we presented an approach coupling Statistical Model Checking and simulations to validate the collision risk estimation provided by a perception-based algorithm. After a suitable definition of KPIs via logic formula to be fulfilled, several execution traces have been generated in simulation considering realistic models of vehicles and sensors. Results for both collision and almost-collision scenarios have been provided analyzing the estimations generated by a state-of-the-art solution, the CMCDOT algorithm. Furthermore, real experiments recreating similar scenarios have been conducted in a controlled environment and the recorded data has been compared to simulations to show their robustness and ability to reproduce real-world scenarios.

In the future, we intend to continue this analysis both in simulation and in real scenarios. In particular, we intend to better study the border-case scenarios where the vehicles pass extremely close one each other without generating any collision. These are the most complex cases to predict and a specific analysis should be carried out.

ACKNOWLEDGMENT

The authors want to thank Jerome Lussereau for his support in carrying out the real experiments.

This work has been partly founded by the IRT Nanoelec, a french hub (”Investissement d’Avenirs” ANR-10-AIRT-05) for research and innovation in microelectronics and information & communication technologies.

REFERENCES

[1] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, *et al.*, “A perception-driven autonomous urban vehicle,” *Journal of Field Robotics*, vol. 25, no. 10, pp. 727–774, 2008.

[2] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, *et al.*, “Autonomous driving in urban environments: Boss and the urban challenge,” *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[3] X. Zhao, V. Robu, D. Flynn, F. Dinmohammadi, M. Fisher, and M. Webster, “Probabilistic model checking of robots deployed in extreme environments,” *arXiv preprint arXiv:1812.04128*, 2018.

[4] “Prism,” https://www.prismmodelchecker.org/.

[5] M. Althoff and J. M. Dolan, “Online verification of automated road vehicles using reachability analysis,” *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[6] A. Colombo, D. Fontanelli, A. Legay, L. Palopoli, and S. Sedwards, “Motion planning in crowds using statistical model checking to enhance the social force model,” in *52nd IEEE Conference on Decision and Control*, 2013, pp. 3602–3608.

[7] M. Barbier, A. Renzaglia, J. Quilbeuf, L. Rummelhard, A. Paigwar, C. Laugier, A. Legay, J. Ibañez-Guzmán, and O. Simonin, “Validation of perception and decision-making systems for autonomous driving via statistical model checking,” in *IEEE Intelligent Vehicles Symposium (IV)*, 2019.

[8] S. Lefèvre, D. Vasquez, and C. Laugier, “A survey on motion prediction and risk assessment for intelligent vehicles,” *ROBOMECH journal*, vol. 1, no. 1, p. 1, 2014.

[9] L. Rummelhard, A. Négre, and C. Laugier, “Conditional monte carlo dense occupancy tracker,” in *IEEE 18th International Conference on Intelligent Transportation Systems*, 2015, pp. 2485–2490.

[10] L. Rummelhard, A. Nègre, M. Perrollaz, and C. Laugier, “Probabilistic grid-based collision risk prediction for driving application,” in *International Synposium on Experimental Robotics*, Springer, Ed., 2014.

[11] J. Ibanez-Guzman, S. Lefevre, A. Mokkadem, and S. Rodhaim, “Vehicle to vehicle communications applied to road intersection safety, field results,” in *13th International IEEE Conference on Intelligent Transportation Systems*, 2010, pp. 192–197.

[12] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[13] A. Legay, S. Sedwards, and L.-M. Traonouez, “Plasma lab: a modular statistical model checking platform,” in *International Symposium on Leveraging Applications of Formal Methods*. Springer, 2016, pp. 77–93.

[14] E. T. Jaynes, *Probability theory: the logic of science*. Cambridge university press, 2003.

[15] T. Hérault, R. Lassaigne, F. Magniette, and S. Peyronnet, “Approximate probabilistic model checking,” in *Proceedings of the 5th International Conference on Verification, Model Checking, and Abstract Implementations*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, vol. 2937, pp. 73–84.

[16] K. Sen, M. Viswanathan, and G. Agha, “On statistical model checking of stochastic systems,” in *17th International Conference on Computer Aided Verification*, ser. Lecture Notes in Computer Science, K. Etessami and S. K. Rajamani, Eds. Springer Berlin Heidelberg, 2005, vol. 3576, pp. 266–280.

[17] A. Pnueli, “The temporal logic of programs,” in *Proc. of the 18th Annual Symposium on Foundations of Computer Science*. IEEE Computer Society, 1977, pp. 46–57.

[18] P. Zuliani, A. Platzer, and E. M. Clarke, “Bayesian statistical model checking with application to stateflow/simulink verification,” *Formal Methods in System Design*, 2013.

[19] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, “Multi-target tracking using joint probabilistic data association,” in *19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, vol. 19. IEEE, 1980, pp. 807–812.

[20] Z. Khan, T. Balch, and F. Dellaert, “An mcmc-based particle filter for tracking multiple interacting targets,” in *Computer Vision-ECCV*. Springer, 2004, pp. 279–290.

[21] R. Labayrade, C. Royere, and D. Aubert, “Experimental assessment of the rescue collision-mitigation system,” *Vehicular Technology, IEEE Transactions on*, vol. 56, no. 1, pp. 89–102, 2007.

[22] N. Kaempchen, B. Schiele, and K. Dietmayer, “Situation assessment of an autonomous emergency brake for arbitrary vehicle-to-vehicle collision scenarios,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 4, Jan 2009.

[23] A. Paigwar, O. Erkent, C. Wolf, and C. Laugier, “Attentional pointnet for 3d-object detection in point clouds,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[24] A. Elfes, “Using occupancy grids for mobile robot perception and navigation,” *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[25] H. Moravec, “Sensor fusion in certainty grids for mobile robots,” *AI magazine*, vol. 9, no. 2, p. 61, 1988.

[26] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl, “Remote attacks on automated vehicles sensors: Experiments on camera and lidar,” *Black Hat Europe*, vol. 11, p. 2015, 2015.

[27] K. Witowski and N. Stander, "Parameter identification of hysteretic models using partial curve mapping," in *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2012, p. 5580.