Object-Level Data-Driven Sensor Simulation for Automotive Environment Perception

László Lindenmaier, Szilárd Aradi, *Member, IEEE*, Tamás Bécsi, *Member, IEEE*, Olivér Törő, and Péter Gáspár

***Abstract*—The gradually evolving automated driving and ADAS functions require more enhanced environment perception. The key to reliable environmental perception is large amounts of data that are hard to collect. Several simulators provide realistic, raw sensor data based on physical sensor models. However, besides their high price, they also require very high computation capacity. Furthermore, most sensor suppliers provide high-level data, such as object detections, that is complicated to reproduce from simulated raw sensor data. This paper proposes a method that directly simulates the detections or object tracks provided by smart sensors. The model involves several uncertainties of the sensors, such as missed-, false detections, and measurement noise. In contrast to the conventional sensor models, this method tackles with state-dependent clutter model and considers the field of view in the detections model. The parameters of the proposed model are identified for an automotive smart radar and camera based on pre-evaluated real-world measurements. The resulting model provides synthetic object-level data with higher fidelity than the conventional probabilistic models, differing less than 2% from the precision and recall metrics of the actual sensors.**

***Index Terms*—Advanced driver assistance, object detection, radar, radar clutter, sensor systems, simulation, smart cameras.**

I. Introduction

THE ADAS (Advanced Driver Assistance System) and AD (Automated Driving) functions promise a number of benefits in terms of sustainability [1], traffic-flow optimization [2], and economics [3]. Besides these advantages, the greatest motivation of ADAS functions is to increase traffic safety [4]. Therefore, the EU's GSR (General Safety Regulations) forces the OEMs (Original Equipment Manufacturer) to introduce new ADAS features into their vehicles. None of the automotive sensors available on the market can satisfy the requirements of these advanced functions in terms of environment perception [5]. Therefore, the perception of intelligent vehicles equipped with ADAS functions is performed by the fusion of several sensors with different advantages [6], [7], [8]. Several tests ensure the functional safety of these systems on different levels, such as unit, module, and integration tests. In the early stages of development, testing and verification on real-world measurement are not cost-efficient, and data acquisition is time-consuming; hence, using simulation tools is beneficial [9]. Furthermore, in the case of HAD (Highly Automated Driving), testing of corner cases is complicated in a real environment due to safety and reproducibility problems. Simulation of traffic flow and vehicle dynamics are already widespread techniques in the testing of high-level decision-making and control algorithms [10], [11], [12]. Synthetic data augmentation is also commonly used to develop and test environment perception functionalities such as image processing, object detection, and sensor data fusion. In this article, we propose a generic model of various smart sensors with different outputs, such as detections and tracked objects. The proposed model can generate synthetic sensor data, supporting the development and testing of multi-object tracking and sensor fusion algorithms on different abstraction levels such as detection or track level [13], [14].

A. Related Work

A wide range of works focuses on the automatic scenario generation and simulation of automotive sensors based on various approaches due to the increasing need for testing ADAS and HAD functions [15], [16], [17], [18], [19]. Three types of simulators can be distinguished according to the underlying sensor model, i.e., ideal, probabilistic, and physics-based models [18], [20]. Low-, medium-, and high-fidelity models are closely related to the model types, which are also distinguished in [18]. The ideal sensor models simply consider the ground truth objects within the sensor's field of view (FoV) without data manipulation [21]. The physics-based sensor models have a high fidelity in the simulation of raw sensor data [22], [23], [24], [25]. Recently, ray tracing has become a key technology in this field that allows the simulation of different sensors, such as radars [26], [27], [28], LiDARs [29], cameras [30], and ultrasonic sensors [31]. A scalable, generic physically-based simulator using ray tracing is proposed in [32]. The commercial, automotive simulation tools, such as Microsoft AirSim [33],

Manuscript received 8 May 2023; accepted 9 June 2023. Date of publication 19 June 2023; date of current version 15 November 2023. This work was supported in part by the European Union within the framework of the National Laboratory for Autonomous Systems under Grant RRF-2.3.1-21-2022-00002 and in part by the Ministry of Innovation and Technology of Hungary from the National Research, Development and Innovation Fund, through Project BME-NVA-02 under Grant TKP2021 funding scheme. *(Corresponding author: Tamás Bécsi.)*

László Lindenmaier, Szilárd Aradi, Tamás Bécsi, and Olivér Törő are with the Department of Control for Transportation and Vehicle Systems, Budapest University of Technology and Economics, 1111 Budapest, Hungary (e-mail: lindenmaier.laszlo@kjk.bme.hu; aradi.szilard@kjk.bme.hu; becsi.tamas@kjk.bme.hu; toro.oliver@kjk.bme.hu).

Péter Gáspár is with the Systems and Control Laboratory, Institute for Computer Science and Control, 1111 Budapest, Hungary (e-mail: gaspar.peter@sztaki.mta.hu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIV.2023.3287278.

Digital Object Identifier 10.1109/TIV.2023.3287278

Vector DYNA4 [34], Hexagon VTD [35], dSPACE ASM [36], IPG CarMaker [37], and Siemens PreScan [38], besides the vehicle and traffic model, usually include a simplified simulation of specific sensor types using game engines, such as Unreal [39] or Unity [40]. The less complex rasterization and ray casting techniques are commonly used in these render engines, simulating digital cameras [41], [42], [43], [44], and raw LiDAR point clouds [45], [46], [47], [48]. There are a lot of alternative approaches for the simulation of raw radar data as well [49], [50], such as time-domain electromagnetic models as in [51].

However, physical sensor model-based simulation tools generally have significant computational and hardware resource requirements [24], [25], [52], [53]. The probabilistic sensor models can provide a reasonable trade-off between computational complexity and fidelity level [54]. Another question lies in the abstraction level of the data [55]. Most automotive sensor suppliers develop smart sensors that provide processed, object-level data, such as detections or tracked objects, instead of raw measurements. Therefore, physical sensor models should incorporate the processing algorithms of these sensors (e.g., detection and tracking) to reproduce their data. However, these algorithms of the automotive sensors are not public, and there is a tremendous number of approaches for the different sensors, like radars [56], [57], cameras [58], [59], [60], and LiDARs [61]. Therefore, it is challenging to reproduce the object-level output from the raw data of a specific sensor with high fidelity, and this approach is not modular for different types of sensors. The possible errors of the different sensors for probabilistic models are collected in [62]. Some works provide a generic model for sensor simulation, but they mainly tackle the measurement noise [63], [64] and the detectability considering the sensor FoV [65]. Object-level simulations with more detailed detection models are proposed in [66], [67] considering the physical aspects of radar reflections. The radar model in [66] was extended with further physical aspects [68] and clutter detection simulation [69]. A generic object-based data-driven model is proposed in [70]; however, the detection model follows a Bernoulli distribution unsuitable for tracked objects. The clutter model uses uniform distribution, neglecting the static reflecting areas (e.g., guardrails, bridges), resulting in a non-uniform spatial distribution of false detections.

B. Contributions of the Article

In this paper, we propose a high-level generic sensor model that simulates the object-level information, such as detections or tracked objects, of automotive smart sensors with low computational effort. The proposed sensor model consists of three sub-models, the detection and tracking, clutter, and measurement model, tackling the missed and clutter detections and measurement noise. The sensor model parameters are identified using real-world measurements of a simulated camera and radar, resulting in a data-driven simulator with high fidelity. This simulation can support developing and validating object tracking and sensor fusion algorithms. The detection and tracking model imposes a state-dependent detectability of the objects considering single-shot detections and tracked objects. Compared to the conventional simulators, our clutter model deals with the reasons for false detections, including potentially false objects, instead of the conventional Poisson Point Process (PPP) model. The measurement model simulates the observation noise considering the built-in state estimations used in smart sensors. The proposed model of automotive smart sensors is detailed in Section II, organized according to the sub-modules. The parameter identification of the model relying on real-world data is presented in Section III following the same structure as in Section II. The results are shown in Section IV, while the conclusions are drawn, extended with a brief outlook on the future work in Section V.

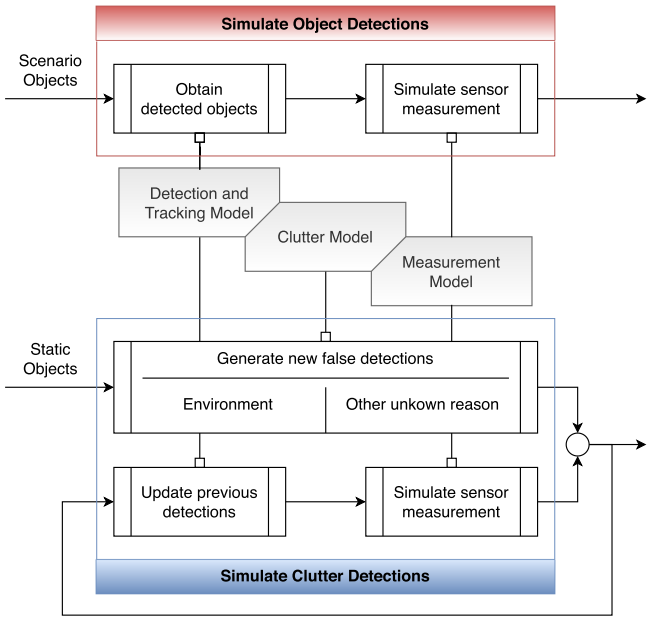


Fig. 1. Generic architecture of the proposed sensor simulation.

II. Sensor Model

The simulation of the automotive environment sensors has to face two questions: what the sensor detects and how reliably. The proposed sensor model is derived from the multi-object tracking algorithms applied in a cluttered environment as in [71], [72] considering the detectability, clutter detections, and the measurement uncertainties of the sensors with the three modules shown in Fig. 1. This concept provides an end-to-end simulation of the objects provided by smart sensors, modeling their built-in perception algorithm. The detection and tracking, and clutter model answer the first question: what does the sensor detect? The sensor detections may be generated by real targets, but some of them are usually false detections. Therefore, the simulation workflow illustrated in Fig. 1 consists of two main processes generating the object- and clutter, i.e., true positive and false positive detections. The detection and tracking model determines which scenario objects, such as passenger cars, trucks, and buses, are assumed to be detected by the sensor, as shown in Fig. 1. The clutter model describes the appearance of false detections generated by static environmental objects, such as guardrails, bridges, traffic signs, or other unknown reasons. Since usually

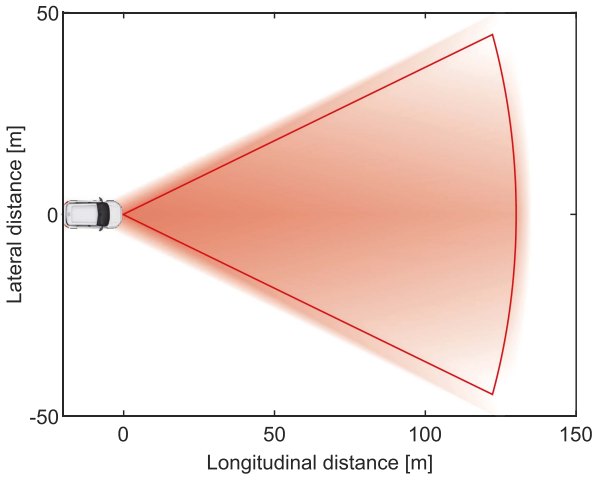


Fig. 2. The detection model derived from the sensor FoV.

automotive smart sensors include a built-in tracking algorithm, the false detections do not disappear immediately, but they survive a couple of cycles. Therefore, the detection and tracking model is utilized in the simulation of clutter detections to update previous false detections, determining which survive. The overall clutter detections are formed by the union of the appearing and tracked detections. The measurement model simulates the measurements generated by the targets, i.e., the surrounding and static objects corresponding to the object- and clutter detections, respectively. It considers the observation noise of the simulated sensors. The modules of the proposed model, the detection and tracking, clutter, and measurement models, are detailed in the following subsections.

A. Detection and Tracking Model

The detection model intends to describe how likely the sensor is to detect a present object, thus simulating false negative detections. In multi-object tracking algorithms, the detection probability $p_D(\boldsymbol{x})$ corresponds to the event that an object with state $\boldsymbol{x}$ is detectable; a constant value usually gives that as in [73], [74]. There are works tackling object tracking problems in varying detectability environments as in [75], [76], [77], [78]. In this article, we propose a state-dependent detection probability model for $p_D(\boldsymbol{x})$, considering the sensor FoV (field of view) and the signal processing of smart sensors. A general detection model is demonstrated in Fig. 2, where the color intensity corresponds to the value of detection probability, and the solid line represents the manufacturer datasheet FoV. It is common to model the FoV of a sensor by circle sectors or annuli defined by a distance and angle range. Therefore, the detection probability over the surveillance area is approximated by the following bivariate function of the d distance and φ azimuth angle of the object with state $\boldsymbol{x}$:

$$\hat{p}_D(d,\varphi) = \max\left(p_{D,\max} - f_d(d) - f_\varphi(\varphi), 0\right) , \quad (1)$$

where $p_{D,\max}$ denotes the constant corresponding to the maximum detection probability. The $f_d(d)$ distance and $f_\varphi(\varphi)$ angle-dependent functions in (1) are defined as

$$f_d(d) = \begin{cases} 0, & \text{if } d \le b_d \\ c_d(d - b_d), & \text{otherwise} \end{cases} , \quad (2)$$

$$f_\varphi(\varphi) = \begin{cases} 0, & \text{if } |\varphi| \le b_\varphi \\ c_\varphi(|\varphi - \varphi_0| - b_\varphi) , & \text{otherwise} \end{cases} , b_\varphi \ge 0 \quad (3)$$

where c_d, c_φ, b_d, b_φ are the coefficients and breakpoints of distance and angle-dependent linear functions, respectively, and φ_0 denotes the orientation angle of the sensor. Many automotive radars have multiple FoVs (e.g., a near-range and a far-range). In this case, the approximated detection probability function $\hat{p}_D(d,\varphi)$ is constructed separately.

The simulation of the object detections provided by a sensor requires the set of surrounding objects $\boldsymbol{X}$ of the simulated scenario. Furthermore, most sensors have an interface limitation on the number of provided detections due to the bandwidth; therefore, the scenario objects should be sorted according to their relevancy (e.g., distance), assuming that the sensor provides the detections in this order. Then, the set of single-shot detections $\boldsymbol{Z}_o^s$ at the k-th timestamp is simulated, assuming that the perception follows a Bernoulli distribution with $\hat{p}_D(d,\varphi)$ parameter considering the $n_{Z,\max}$ maximum number of detections provided by the sensor.

However, several smart sensors are equipped with an object tracking algorithm resolving some missed detections of a single-shot detector. Therefore, the Bernoulli distribution is not suitable for modeling the detectability of tracked objects. In this case, the $p_{init}(\boldsymbol{x}_k^i)$ initiation and $p_{del}(\boldsymbol{x}_k^i)$ deletion probabilities of object $\boldsymbol{x}_k^i$ are defined as

$$p_{init}(\boldsymbol{x}_k^i) = \max(p_t(\boldsymbol{x}_k^i) - p_{rc}(\boldsymbol{x}_k^i), 0) , \quad (4)$$

$$p_{del}(\boldsymbol{x}_k^i) = \max(p_{rc}(\boldsymbol{x}_k^i) - p_t(\boldsymbol{x}_k^i), 0) , \quad (5)$$

where $p_{rc}(\boldsymbol{x}_k^i)$ denotes the recall of object $\boldsymbol{x}_k^i$ at the k-th timestamp, and it is computed as the proportion of the $N_d(\boldsymbol{x}_k^i)$ and $N_p(\boldsymbol{x}_k^i)$ number of cycles in what the object was detected and present. The $p_t(\boldsymbol{x}_k^i)$ tracking probability considers the detection probability of the object during its life cycle as

$$p_t(\boldsymbol{x}_k^i) = \frac{p_D(\boldsymbol{x}_k^i) + N_p(\boldsymbol{x}_k^i) - p_t(\boldsymbol{x}_{k-1}^i)}{N_p(\boldsymbol{x}_k^i)} . \quad (6)$$

Then, the set of tracked object detections $\boldsymbol{Z}_o^t$ at the k-th timestamp is simulated according to Algorithm 1, where $\gamma(\boldsymbol{x}_{k-1}^i)$ indicates whether the object $\boldsymbol{x}_{k-1}^i$ was detected at timestamp $k-1$, and $p_{del,th}$ is the deletion threshold probability establishing a more stable tracking model. If an object $\boldsymbol{x}_k^i$ is assumed to be perceived at time k, the sensor measurement considering the observation noise is simulated according to the $g_k(\boldsymbol{x}_k^i)$ measurement model detailed in Section III-C.

B. Clutter Model

The clutter model intends to produce the false positive detections of the simulated sensors as the physical sensors would provide them. In many multi-object tracking and sensor fusion

Algorithm 1: Constructing tracked object detections.

given $\boldsymbol{X} = \{\boldsymbol{x}_{k-1:k}^i\}_{i=1}^n$ real objects at time k and $k-1$
$\boldsymbol{Z}_o^t = \emptyset, \quad n_{Z,k} = 0$
for $i = 1$ **to** n **do**
  Compute $p_{init}(\boldsymbol{x}_k^i)$ and $p_{del}(\boldsymbol{x}_k^i)$ via (5) and (6)
  $r = \text{rand} \sim \mathcal{U}(0,1)$
  if $\gamma(\boldsymbol{x}_{k-1}^i)$ **then**
    if $r < p_{del}(\boldsymbol{x}_k^i) \wedge p_{del}(\boldsymbol{x}_k^i) \geq p_{del,th}$ **then**
      $i = i + 1$ (False negative detection)
    end if
  else
    if $r \geq p_{init}(\boldsymbol{x}_k^i)$ **then**
      $i = i + 1$ (False negative detection)
    end if
  end if
  if $n_{Z,k} < n_{Z,\max}$ **then**
    $\boldsymbol{z} \sim g_k(\boldsymbol{x}_k^i)$
    $\boldsymbol{Z}_o^t = \boldsymbol{Z}_o^t \cup \boldsymbol{z}$
    $n_{Z,k} = n_{Z,k} + 1$
  end if
end for
output: $\boldsymbol{Z}_o^t = \{\boldsymbol{z}_k^{i_z}\}_{i_z=1}^{n_{Z,k}}$

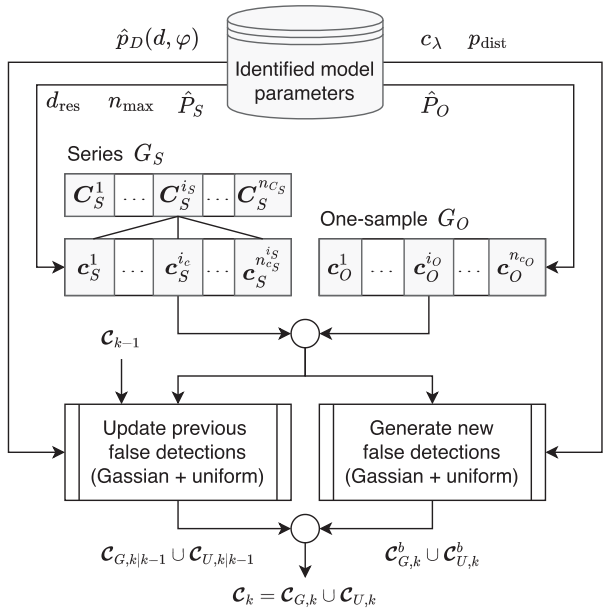


Fig. 3. The generic overview of the proposed clutter model.

algorithms, the clutter model is given by a Poisson Point Process (PPP) as in [79], [80]. The PPP model assumes uniformly distributed clutter detections in a given measurement volume with its cardinality distributed according to a Poisson distribution. Due to its simplicity, it is also commonly used in sensor simulations, as in [69], [70]. However, most of the sensors have non-uniform clutter density, and the cardinality of the false detections does not follow a Poisson distribution.

In this article, we provide a non-uniform clutter model that enables a realistic simulation of the sensors' clutter detections. The first step of such a clutter model is to explore the reasons for false detections. For instance, radars usually detect static objects with significant, reflecting areas such as guardrails, poles, highway bridges, etc., resulting in a non-uniform clutter density. The ADAS and AD functions simulation is usually performed based on standard static environment and scenario descriptors such as OpenDRIVE and OpenSCENARIO. The descriptor of the environment, including the road network, lanes, and stationary objects (e.g., traffic lights, traffic signs, etc.), enables one to identify the static and potentially false objects. Smart cameras usually provide fewer false detections, but it is very challenging to identify their exact spatial distribution since it depends highly on the image processing algorithm. Therefore, the proposed clutter model considers the false detections generated by static environment objects and other unknown reasons (e.g., random sensor noise, object detection errors). The workflow of simulating the clutter detections is illustrated in Fig. 3. The set of clutter detections, $\mathcal{C}_k$, at time k consists of the $\mathcal{C}_G$ Gaussian and the $\mathcal{C}_U$ uniform detections generated by the static environmental objects and other random events, respectively. The clutter model considers that the false detections provided by smart sensors with a built-in tracking algorithm usually do not appear for only one frame. Therefore, in the proposed method, the clutter detections from the previous step, $k-1$, are first updated using the detection and tracking model. Although the total cardinality of the tracked clutters does not follow a Poisson distribution as assumed by conventional PPP models, it gives a reasonable estimate of the number of new false detections. Therefore, the new clutters are generated by Poisson distributions with unique parameters corresponding to the different kinds of assumed origins (e.g., guardrails, lamp poles, bridges) of the detections.

In the following, the construction of the $\mathcal{C}_G$ Gaussian and $\mathcal{C}_U$ uniform clutter groups is detailed. The two groups are generated separately, starting with the Gaussian group. The first step of the static clutter detection simulation is identifying the Gaussian components corresponding to the potentially false static objects. The Gaussian components can be divided into two types: $\boldsymbol{G}_S = \{\boldsymbol{C}_S^{i_S}\}_{i_S=1}^{n_{C_S}}$ contains clusters of ordered series of components and $\boldsymbol{G}_O = \{\boldsymbol{c}_O^{i_O}\}_{i_O=1}^{n_{c_O}}$ consists of individual one-sample components as shown in Fig 3, where n_{C_S} and n_{c_O} denotes the number of component clusters and the individual one-sample components. In the parameter identification and evaluation, we considered the guardrails ($i_S = 1$) and lamp poles ($i_S = 2$) as series components since they usually generate a sequence of detections, therefore $n_{C_S} = 2$ in this case. The highway bridges are regarded as individual Gaussian components. The number of bridges in the view range of the simulated sensors determines n_{c_O}.

A cluster $\boldsymbol{C}_S^{i_S}$ that corresponds to the $l_S^{i_S}$ class of static, potentially false object (i.e., l_S^1 = guardrail, l_S^1 = lamp poles) consists of $\boldsymbol{c}_S^{i_c} = \{\hat{\boldsymbol{c}}_S^{i_c}, \hat{\boldsymbol{P}}_S^{i_c}, w_S^{i_c}, l_S^{i_S}\}$ series of Gaussian components as in Fig. 3, where i_c denotes the index of the components within the i_S series. The components are following each other with d_{res} distance resolution along the road path within the

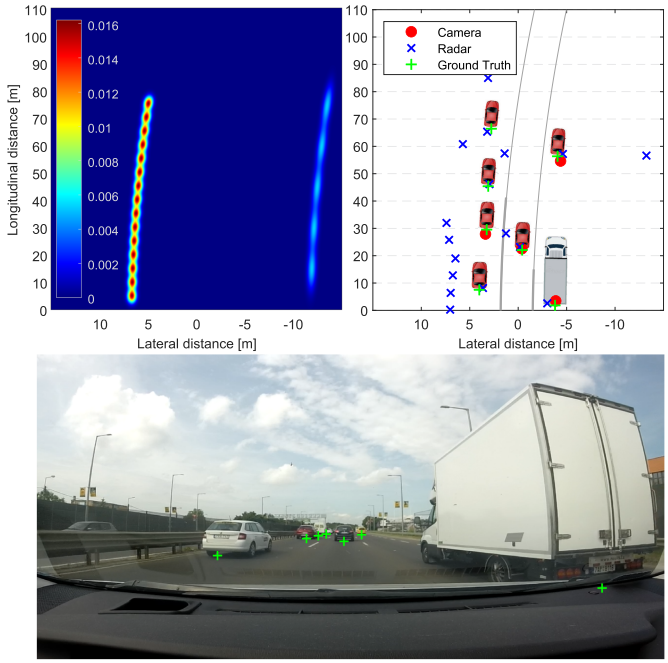


Fig. 4. The spatial PDF of the guardrail (left side) and lamp poles (right side) Gaussian components.

sensor's view range. The $\hat{\boldsymbol{c}}_S^{i_c}$ state of the i_c-th component is determined by the location of the corresponding objects, i.e., guardrails and lamp poles, stored in the map data of the simulated environment. This information can usually be extracted from many simulation tools in standard format (e.g., OpenDRIVE). The covariance, $\hat{\boldsymbol{P}}_S^{i_c}$, is set so that the eigenvector corresponding to the longitudinal position of the component is parallel to the road path. The series of the guardrail's and lamp poles' Gaussian components are demonstrated in Fig. 4. The $w_S^{i_c}$ weights of the Gaussian components are computed based on $\gamma(\boldsymbol{c}_S^{i_c})$, that indicates whether the $\boldsymbol{c}_S^{i_c}$ component is already detected at the current timestamp and $p_{\text{dist}}(\hat{\boldsymbol{c}}_S^{i_c}, l_S^{i_c})$ distance probability density as

$$w_S^{i_c} = p_{\text{dist}}(\hat{\boldsymbol{c}}_S^{i_c}, l_S^{i_S})(1-\gamma(\boldsymbol{c}_S^{i_c}))\,. \tag{7}$$

The number of new false detections is computed for each $\boldsymbol{C}_S^{i_S} \in \boldsymbol{G}_S$ Gaussian series classes (i.e., guardrail, lamp pole) with label $l_S^{i_S}$ according to a Poisson distribution with Poisson rate $\bar{\lambda}_{l_S^{i_S}}$. The Poisson rate is proportional to the current number of components, $n_{cS}^{i_S}$, constructing the object class $l_S^{i_S}$ and to the $v_{\text{ego},k-1}(t_k - t_{k-1})$ distance that is traveled between timestamp $k-1$ and k as

$$\bar{\lambda}_{\boldsymbol{C}_S^{i_S}} = \bar{\lambda}_{l_S^{i_S}} = c_\lambda(l_S^{i_S})\, v_{\text{ego},k-1}(t_k - t_{k-1}) \frac{n_{cS}^{i_S}}{n_{\max}(l_S^{i_S})}\,, \tag{8}$$

where $c_\lambda(l_S^{i_S})$ and $n_{\max}(l_S^{i_S})$ denote the constant Poisson rate factor and the maximum number of components constructing static false object with label $l_S^{i_S}$.

The $\{\boldsymbol{c}_O^{i_O}\}_{i_O=1}^{n_{cO}} = \{\hat{\boldsymbol{c}}_O^{i_O}, \hat{\boldsymbol{P}}_O^{i_O}, w_O^{i_O}, l_O^{i_O}\}_{i_O=1}^{n_{cO}}$ one-sample Gaussian components, shown in Fig. 3, are generated based on the static environment description of the generated scenario. The state $\hat{\boldsymbol{c}}_O^{i_O}$ is computed according to the location of the object and the dynamics of the ego vehicle. The covariance matrix $\hat{\boldsymbol{P}}_O^{i_O}$ is given so that the spatial distribution of the components corresponds to the shape and angle of the object and $\text{tr}(\hat{\boldsymbol{P}}_O^{i_O})$ is proportional to the object's reflecting area $|A_r|$. The weights $w_O^{i_O} = \bar{\lambda}_{\boldsymbol{c}_O^{i_O}}$, as opposed to the clusters including the series of Gaussian components, directly define the Poisson rate of the cardinality of new clutter detections generated by the one-sample component $\boldsymbol{c}_O^{i_C}$ and are computed as

$$w_O^{i_O} = \bar{\lambda}_{\boldsymbol{c}_O^{i_O}} = f_{w_O}(\bar{\lambda}_{l_O^{i_O}}, \hat{\boldsymbol{c}}_O^{i_O})\,, \tag{9}$$

where the Poisson rate $\bar{\lambda}_{l_O^{i_O}}$ of the static object class with label $l_O^{i_O}$ (i.e., highway bridges) is given similar to (8) by neglecting the number of components of the cluster as

$$\bar{\lambda}_{l_O^{i_O}} = c_\lambda(l_O^{i_O})\, v_{\text{ego},k-1}(t_k - t_{k-1})\,. \tag{10}$$

The f_{w_O} function that computes the Poisson rate of each component is defined by the following two rules:

$$\int_{V_{FOV}} f_{w_O}(\bar{\lambda}_{l_O^{i_O}}, \hat{\boldsymbol{c}}_O^{i_O})\, dV = \bar{\lambda}_{l_O^{i_O}} V_{FOV}\,, \tag{11}$$

$$f_{w_O}(\bar{\lambda}_{l_O^{i_O}}, \hat{\boldsymbol{c}}_O^{i_O}) \propto p_{\text{dist}}(\hat{\boldsymbol{c}}_O^{i_O}, l_O^{i_O}), \quad \forall i_O \in \boldsymbol{G}_O\,, \tag{12}$$

imposing that the $\bar{\lambda}_{l_O^{i_O}}$ overall Poisson rate of the object class with label l_O does not change and the state-dependent Poisson rate of the component $\boldsymbol{c}_O^{i_O}$ is proportional to $p_{\text{dist}}(\hat{\boldsymbol{c}}_O^{i_O}, l_O^{i_O})$.

The Gaussian clutter detections $\boldsymbol{\mathcal{C}}_{G,k}$ at timestamp k are constructed by Algorithm 2. First, the false Gaussian objects, $\boldsymbol{\mathcal{C}}_{G,k-1} = \{\boldsymbol{c}_{G,k-1}^{i_G}\}_{i_G=1}^{n_{cG,k-1}}$ of the previous cycle, $k-1$, are updated by the measurement model g_k detailed in Section II-C using the Gaussian components $\boldsymbol{c}_k^\star \in \{\boldsymbol{G}_s, \boldsymbol{G}_s\}$ corresponding to the static object that originally generated the detection. The detection survives if its survive probability is less than the uniformly distributed $r \sim \mathcal{U}(0,1)$ random number. The survive probability $p_s(\boldsymbol{c}_{G,k-1}^{i_G})$ is given according to the detection and tracking model by the $\hat{p}_D(\hat{\boldsymbol{c}}_{G,k-1}^{i_G})$ detection probability if a single-shot detector is simulated and by $1 - p_{del}(\hat{\boldsymbol{c}}_{G,k-1}^{i_G})$ when the sensor provides tracked objects.

In the second step, the appearance of the new Gaussian false detections are simulated. The n_G^b number of new clutters generated by the $\{\boldsymbol{C}_S^{i_S}\}_{i_S=1}^{n_{CS}}$ clusters including the series of Gaussian components, and the $\{\boldsymbol{c}_O^{i_O}\}_{i_O=1}^{n_{cO}}$ one-sample static objects, is computed by a Poisson random number with Poisson rate $\bar{\lambda}_{\boldsymbol{C}_S^{i_C}}$ and $\bar{\lambda}_{\boldsymbol{c}_O^{i_C}}$ according to (8) and (9), respectively. The $\boldsymbol{c}_G^b$ state of a new false detection corresponding to a series of Gaussian components is performed by a random vector distributed according to a weighted Gaussian mixture in line 24, where the weight is computed according to (7). The one-sample components initiate the state of a newborn false detection defined by the corresponding Gaussian distribution. Furthermore, the maximum number of detections $n_{Z,\max}$ of the sensor is considered.

The uniform clutter group $\boldsymbol{\mathcal{C}}_{U,k}$ representing the false detections not generated by static objects but for other unknown

Algorithm 2: Constructing Gaussian False Detections.

given $\boldsymbol{G}_S = \{\boldsymbol{C}_S^{i_S}\}_{i_S=1}^{n_{C_S}}$ series Gaussian components, and $\boldsymbol{G}_O = \{\boldsymbol{c}_O^{i_O}\}_{i_O=1}^{n_{c_O}}$ one-sample Gaussian components, $\boldsymbol{\mathcal{C}}_{G,k-1} = \{\boldsymbol{c}_{G,k-1}^{i_G}\}_{i_G=1}^{n_{c_G,k-1}}$ Gaussian false objects, and $n_{Z,k}$ number of object detections at time $k-1$ and k

$\boldsymbol{\mathcal{C}}_{G,k|k-1} = \emptyset, \quad \boldsymbol{\mathcal{C}}_{G,k}^b = \emptyset, \quad n_{c_G,k} = 0$

Update previous Gaussian false detections:

for $i_G = 1$ **to** $n_{c_G,k-1}$ **do**
  $r = \text{rand} \sim \mathcal{U}(0,1)$
  if single-shot detector **then**
    $p_s(\boldsymbol{c}_{G,k-1}^{i_G}) = \hat{p}_D(\hat{\boldsymbol{c}}_{G,k-1}^{i_G})$
  else
    $p_s(\boldsymbol{c}_{G,k-1}^{i_G}) = 1 - p_{del}(\hat{\boldsymbol{c}}_{G,k-1}^{i_G})$
  end if
  if $r < p_s(\boldsymbol{c}_{G,k-1}^{i_G}) \wedge n_{Z,k} + n_{c_G,k} < n_{Z,\max}$ **then**
    $\boldsymbol{c}_G \sim g_k(\boldsymbol{c}_k^\star \leftrightarrow \boldsymbol{c}_{G,k-1}^{i_G})$
    $\boldsymbol{\mathcal{C}}_{G,k|k-1} = \boldsymbol{\mathcal{C}}_{G,k|k-1} \cup \boldsymbol{c}_G$
    $n_{c_G,k} = n_{c_G,k} + 1$
  end if
end for

Generate new Gaussian false detections:

for $i_C = 1$ **to** $n_{C_S} + n_{c_O}$ **do**
  if $i_C \le n_{C_S}$ **then**
    $n_G^b = \text{rand} \sim \text{Pois}(\bar{\lambda}_{\boldsymbol{C}_S^{i_C}})$
  else
    $n_G^b = \text{rand} \sim \text{Pois}(\bar{\lambda}_{\boldsymbol{c}_O^{i_C}})$
  end if
  for $i_G^b = 1$ **to** n_G^b **do**
    if $i_C \le n_{C_S}$ **then**
      $\boldsymbol{c}_G^b = \text{rand} \sim \sum_{\forall j_c \in \boldsymbol{C}_S^{i_C}} w_S^{j_c} \mathcal{N}(\cdot, \hat{\boldsymbol{c}}_S^{j_c}, \hat{\boldsymbol{P}}_S^{j_c})$
    else
      $\boldsymbol{c}_G^b = \text{rand} \sim \mathcal{N}(\cdot, \hat{\boldsymbol{c}}_O^{i_C}, \hat{\boldsymbol{P}}_O^{i_C})$
    end if
    if $n_{Z,k} + n_{c_G,k} < n_{Z,\max}$ **then**
      $\boldsymbol{\mathcal{C}}_{G,k}^b = \boldsymbol{\mathcal{C}}_{G,k}^b \cup \boldsymbol{c}_G^b$
      $n_{c_G,k} = n_{c_G,k} + 1$
    end if
  end for
end for
$n_{Z,k} = n_{Z,k} + n_{c_G,k}$

output: $\boldsymbol{\mathcal{C}}_{G,k} = \boldsymbol{\mathcal{C}}_{G,k|k-1} \cup \boldsymbol{\mathcal{C}}_{G,k}^b = \{\boldsymbol{c}_{G,k}^{i_G}\}_{i_G=1}^{n_{c_G,k}}$

reasons is constructed similarly. In the first step, the survival of the previous $\{\boldsymbol{c}_{U,k-1}^{i_U}\}_{i_U=1}^{n_{c_U,k-1}}$ uniform clutter detections are simulated based on the detection or tracking model as in lines 3–15 of Algorithm 2, but their $\boldsymbol{c}_k^\star$ states are updated using a CA (Constant Acceleration) model prediction. In the second step, the new uniform clutter detection are initiated at time k according to Algorithm 3. The $n_{U,b}$ number of new, uniformly distributed false objects is drawn from a Poisson distribution with rate $\bar{\lambda}_U$ that is computed similarly to (10). However, it is assumed to be proportional to the $t_k - t_{k-1}$ elapsed time instead of the traveled

Algorithm 3: Constructing Uniform False Detections.

given $\boldsymbol{\mathcal{C}}_{U,k-1} = \{\boldsymbol{c}_{U,k-1}^{i_U}\}_{i_U=1}^{n_{c_U,k-1}}$ uniform false objects, $n_{Z,k}$ number of detections at time $k-1$ and k, respectively

Update $\boldsymbol{\mathcal{C}}_{U,k-1}$ to $\boldsymbol{\mathcal{C}}_{U,k|k-1}$, as in Algorithm 2

Generate new uniformly distributed false detections:

$n_U^b = \text{rand} \sim \text{Pois}(\bar{\lambda}_U = c_\lambda(U)(t_k - t_{k-1}))$
for $i_U^b = 1$ **to** n_U^b **do**
  $d = \text{rand} \sim p_{dist,U}$
  $\boldsymbol{c}_U^b = \text{rand} \sim \mathcal{U}(d \in V^\star \subset V_{FOV})$
  if $n_{Z,k} + n_{c_U,k-1} < n_{Z,\max}$ **then**
    $\boldsymbol{\mathcal{C}}_{U,k}^b = \boldsymbol{\mathcal{C}}_{U,k}^b \cup \boldsymbol{c}_U^b$
    $n_{c_U,k} = n_{c_U,k} + 1$
  end if
end for

output: $\boldsymbol{\mathcal{C}}_{U,k} = \boldsymbol{\mathcal{C}}_{U,k|k-1} \cup \boldsymbol{\mathcal{C}}_{U,k}^b = \{\boldsymbol{c}_{U,k}^{i_U}\}_{i_U=1}^{n_{c_U,k}}$

distance. The states of the new uniform clutter detections follow an intermittently uniform distribution. This means that the V_{FOV} sensor FoV is divided to $V_i \in V_{FOV}$ subspaces with equivalent distance intervals. The d distance of the detection is defined by the $p_{dist,U}$ distance probability density, and the state of the newborn clutter detection, $c_{U,b}$, is assumed to follow a uniform distribution within the subspace $V^\star$ that corresponds to the d distance.

C. Measurement Model

The measurement model intends to describe the observations and the measurement uncertainties of the sensors. The state space of the scenario- and static objects is defined by

$$\boldsymbol{x} = \begin{bmatrix} d_x & v_x & a_x & d_y & v_y & a_y \end{bmatrix}^\top, \tag{13}$$

where d_x, d_y, v_x, v_y, a_x, and a_y denote the longitudinal and lateral position, velocity, and acceleration, respectively. Note that the $\hat{p}_D(d, \varphi)$ state-dependent detection probability is given in polar coordinate system; therefore, the d distance and φ azimuth are created from the d_x and d_y Cartesian coordinates. It should be mentioned that the state space of the objects can be extended with other attributes (e.g., dimensions, orientation) if one of the sensors can provide information on them.

A single-shot measurement $\boldsymbol{z}_k$ generated by an object with state $\boldsymbol{x}_k$ is simulated by adding $\boldsymbol{\eta}_k$ measurement noise to the real object state $\boldsymbol{x}_k$ transformed to the measurement space with the observation model $h_k(\boldsymbol{x}_k)$ as

$$\boldsymbol{z}_k = h_k(\boldsymbol{x}_k) + \boldsymbol{\eta}_k, \tag{14}$$

where $\boldsymbol{\eta}_k$ is assumed to follow a $\mathcal{N}(\cdot, \boldsymbol{0}, \boldsymbol{R}_k)$ Gaussian distribution with $\boldsymbol{0}$ mean value and $\boldsymbol{R}_k$ covariance. This assumption is a commonly used measurement model in different recursive filtering algorithms such as Kalman Filter (KF) [81] or Extended Kalman Filter (EKF) [82].

However, in the case of tracked detections, the $\boldsymbol{x}_k$ state is recursively estimated by a KF or EKF, resulting in $\hat{\boldsymbol{x}}_{k|k}$. In many cases, the measurement $\boldsymbol{z}_k$ is not identical to the estimated state,

but it is a result of the transformation

$$\boldsymbol{z}_k = h_{s,k}(\hat{\boldsymbol{x}}_{k|k})\,, \tag{15}$$

where $h_{s,k}$ denotes the state estimate transformation function that is not necessarily identical to the h_k function. The first step of the state estimation is to compute the $\hat{\boldsymbol{x}}_{k|k-1}$ a priori state estimate. The process model $f_k(\boldsymbol{x}_k)$ predicting the objects is given by the commonly used constant acceleration (CA) model with $\boldsymbol{F}_k$ transition matrix as

$$f_k(\boldsymbol{x}_k) = \boldsymbol{F}_k\boldsymbol{x}_k = I_2 \otimes \begin{bmatrix} 1 & \Delta t & \Delta t^2/2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}_k\,, \tag{16}$$

where Δt denotes the time elapsed between t_k and t_{k-1}. The linear process model in (16) is applied for the state prediction of the uniform false detections as well, since they are not generated by the static environment objects. The $\hat{\boldsymbol{P}}_{k|k-1}$ predicted estimate covariance is computed as

$$\hat{\boldsymbol{P}}_{k|k-1} = \boldsymbol{F}_k\hat{\boldsymbol{P}}_{k|k-1}\boldsymbol{F}_k^\top + \boldsymbol{Q}_k\,, \tag{17}$$

where $\boldsymbol{Q}_k$ denotes the process noise covariance that is given based on the $\sigma_{a,x}$ and $\sigma_{a,y}$ acceleration scales as

$$\boldsymbol{Q}_k = \begin{bmatrix} \sigma_{a,x}^2\,\boldsymbol{q}_k & \boldsymbol{0} \\ \boldsymbol{0} & \sigma_{a,y}^2\,\boldsymbol{q}_k\,. \end{bmatrix}. \tag{18}$$

The submatrix $\boldsymbol{q}_k$ of process noise covariance is computed by the following diadic product:

$$\boldsymbol{q}_k = \boldsymbol{v}\,\boldsymbol{v}^\top, \quad \boldsymbol{v} = \begin{bmatrix} \Delta t^2/2 & \Delta t & 1 \end{bmatrix}^\top \tag{19}$$

The $\hat{\boldsymbol{x}}_{k|k-1}$ a priori state estimate is updated by the $\boldsymbol{z}_k^{\text{sim}}$ simulated measurement based on the $\boldsymbol{K}_k$ Kalman gain as

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k(z_k^{\text{sim}} - h_k(\hat{\boldsymbol{x}}_{k|k-1}))\,. \tag{20}$$

The $\boldsymbol{z}_k^{\text{sim}}$ measurement is simulated according to the simple measurement model described in (14). The $\boldsymbol{K}_k$ Kalman gain and the $\hat{\boldsymbol{P}}_{k|k}$ a posteriori estimate covariance are computed by the equations of the Kalman Filter.

III. Parameter Identification

The parameters of the sensors are identified by a pre-evaluated real-world measurement in the offline training phase to obtain a more realistic simulation. Then in the online phase, the identified parameters can be used to simulate the real sensors in arbitrary environments and scenarios generated with a 3D simulation environment (e.g., Carla, IPG CarMaker), as illustrated in Fig. 5. The simulated sensor cluster consists of a Continental ARS408 smart radar and a Mobileye EyeQ2 smart camera, providing tracked object detections. Therefore, in this article, only the model of sensors providing tracked objects is validated; however, the single-shot detectors have a much more straightforward model. The field of view of the simulated sensors and the overall surveillance area is illustrated in Fig 6. The measurement setup includes a separate digital camera, too, synchronized with the aforementioned simulated smart sensors, allowing video annotation for ground truth generation. The measurement that serves

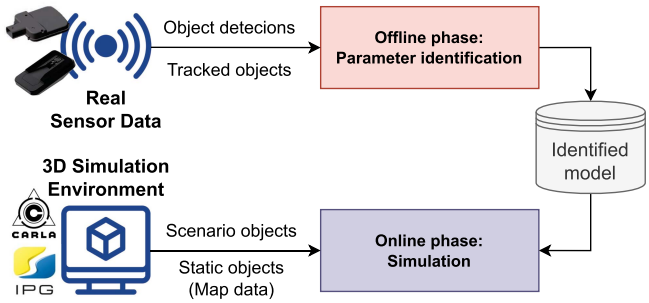


Fig. 5. The relation of parameter identification and simulation.

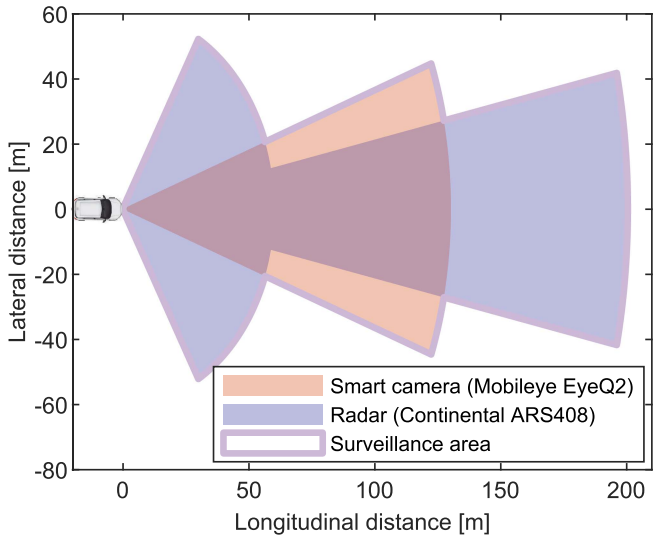


Fig. 6. The field of views (FoV) of the simulated sensors.

as the basis of the parameter identification was recorded on a Hungarian highway in nice weather and usual afternoon traffic conditions, and it consists of 7000 frames ($\approx$120 seconds), including guardrails, lamp poles, and highway bridges. Therefore, the clutter model of static environment objects includes these classes, but it can be extended with other classes of potentially false, static objects (e.g., traffic signs, buildings) by labeling them in the evaluation of the training data used for the parameter identification. Straightforward rules on the relative velocity and position of the false positive detections can support the labeling. The ground truth objects of the road scenario are identified by annotating the video record of the digital camera in the open-source DarkLabel tool. The annotated objects are transformed onto the ego vehicle coordinate system by a projective transformation.

The first step of the measurement evaluation is the association between the annotated ground truth- and detected objects which is performed by a GNN (Global Nearest Neighbor) algorithm [83]. The algorithm assigns the detected objects to the reference objects within a $d_c = 10$ m overall and $d_{c,lat} = 1.5$ m lateral cutoff distance so that the global distance of the association is minimal. The distance matrix $\boldsymbol{D}_{ij} \in \mathbb{R}^{+(N,M)}$, as the basis of the GNN algorithm, is defined by the Mahalanobis-distance $d_{\text{MH}}(\boldsymbol{x}_i, \boldsymbol{z}_j)$ of the reference object $\boldsymbol{x}_i$ and object detection $\boldsymbol{z}_j$

as

$$\boldsymbol{D}_{ij} = d_{\text{MH}}(\boldsymbol{x}_i, \boldsymbol{z}_j) = (\boldsymbol{x}_i^{\text{pos}} - \boldsymbol{z}_j^{\text{pos}})^\top \boldsymbol{S}^{-1} (\boldsymbol{x}_i^{\text{pos}} - \boldsymbol{z}_j^{\text{pos}}), \quad (21)$$

where N and M denote the number of reference- and detected objects, while $\boldsymbol{x}_i^{\text{pos}}$ and $\boldsymbol{z}_j^{\text{pos}}$ are the position vector of the corresponding objects. The covariance matrix $\boldsymbol{S}$ is given based on the ratio $d_r = d_c/d_{c,lat}$ of the cutoff distances as

$$\boldsymbol{S} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & d_r^2 \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}^\top, \quad (22)$$

where α denotes the angle of the road path at the longitudinal distance of object $\boldsymbol{x}_i$.

The detected objects that cannot be assigned to any of the ground truth objects within the distance d_c are assumed to be false detections, while the matched objects are considered as true positive detections. The reference objects that do not have any associated object detections provided by the evaluated sensor are denoted as false negatives.

A. Detection Model

The parameters of the proposed detection model are estimated based on the recall map of the evaluated sensor. The overall recall of a sensor is computed by the ratio of the true positive and reference objects' cardinality as

$$Rc = \frac{TP}{TP + FN}, \quad (23)$$

where TP and FN denote the number of true positive and false negative objects as in [6]. The surrounding environment is divided into elementary cells by discretizing the distance and azimuth angle of the measurement space with 1 m distance and 1° angle resolution. The $Rc([d_i, d_{,i+1}], [\varphi_j, \varphi_{j+1}])$ recall map is constructed by computing the local recall of $[d_i, d_{i+1}] \times [\varphi_j, \varphi_{j+1}]$ intervals. The coasted, i.e., missed-detected but tracked objects that are currently outside but were previously located within the FoV, are not considered in the recall map to neglect the tracking of the sensor. Finally, the proposed model is fitted to the recall map by minimizing the Mean Squared Error (MSE). We have used the Matlab Optimization Toolbox to identify the optimal parameters of the proposed detection model considering the criteria that the distance b_d and the angle breakpoints b_φ cannot be located outside the FoV and the maximal detection probability $p_{D,\max}$ shall be in the range [0,1]. The optimal parameters in (2) and (3) determining the $\hat{p}_D(d, \varphi)$ detection probability map of the sensors, including the two different scan zones of the radar, are summarized in Table I. The fitted detection models of the smart camera and radar are visualized and compared with the manufacturer datasheet FoV of the sensors in Figs. 7 and 8. It can be seen that the fitted detection models give a reasonable estimate of the FoV limits, except in the far scan zone of the radar. Since it is challenging to annotate the distant objects of the road scenario, the reference objects are provided up to 100 meters. Therefore, the detection model cannot be fitted with high reliability over this distance.

TABLE I
ESTIMATED PARAMETERS OF THE PROPOSED DETECTION MODEL IN (2) AND (3)

Parameter	Camera	Radar	
		Near scan zone	Far scan zone
c_d [1/m]	0.0082	0.0047	0.0089
b_d [m]	17.8348	5.9999	70.7781
c_φ [1/°]	0.1288	0.0122	0.1447
b_φ [°]	15.1318	27.0001	3.0002
$p_{D,\max}$ [-]	1.0000	0.9969	0.9294

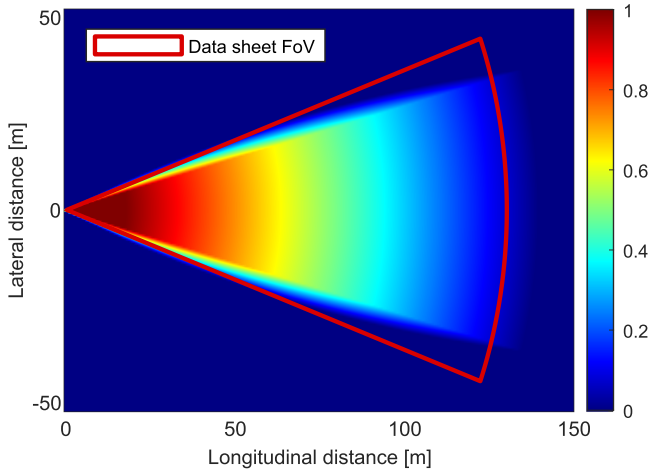


Fig. 7. Comparison of the fitted detection probability map and the manufacturer datasheet FoV of the smart camera.

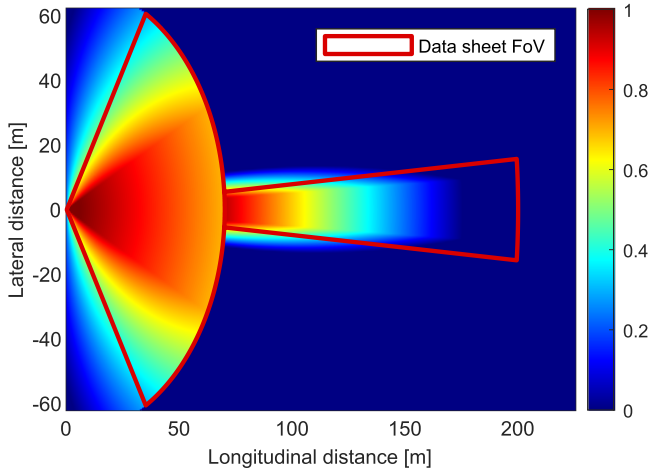


Fig. 8. Comparison of the fitted detection probability map and the manufacturer datasheet FoV of the smart radar.

B. Clutter Model

The parameter identification of the clutter model is more complex since it is not only necessary to identify the false positive detections, but one must classify them as well according to the clutter types. The classification of the clutter detections has been performed by several rules depending on their relative lane position and velocity. This research has considered the

TABLE II
ESTIMATED PARAMETERS OF THE CLUTTER CLASSES

Parameter	Guardrail	Lamp poles	Bridge	Other
Distribution	Gaussian			Uniform
Type	Series		One-sample	N/A
Radar c_λ [-]	0.2912	0.0606	0.0817	1.2450
Camera c_λ [-]	0	0	0	0.0601
d_{res} [m]	5	15	N/A	N/A
$n_{\max}$ [-]	14	6	N/A	N/A
σ_x^2 [m^2]	2.78	25	25	N/A
σ_y^2 [m^2]	0.44	2.25	1.44	N/A

clutter detections generated by guardrails, lamp poles, bridges, or other unknown reasons (e.g., random sensor noise, detection algorithm). The statistics of these detections, such as the total cardinality, the number of new detections, and their spatial distributions, are determined frame by frame. The new detections generated by an object class with label l at k-th cycle are identified based on the GNN data association performed between the detections of the current and previous cycles. The Poisson rate factor $c_\lambda(l)$ of object class l is estimated, in the timestamp interval $[k_s, k_e]$ within the object is present, as nearly constant as

$$c_\lambda(l) = \frac{\bar{\lambda}_l}{\sum_{k=k_s}^{k_e-1} v_{\text{ego},k}(t_{k+1} - t_k)}(k_e - k_s + 1)\,, \tag{24}$$

where k_s and k_e denote the starting and ending timestamp index of the time interval, and v_{ego} is the velocity of the ego vehicle. The $\bar{\lambda}_l$ rate at the average velocity within the estimation time interval is computed by the Matlab Distribution Fitter by fitting a Poisson distribution on the number of new detections. The distance resolution $d_{\text{res}}(l)$ and the maximum number of Gaussian components $n_{\max}(l)$ constructing the objects with labels $l \in \{\boldsymbol{C}_S^{i_S}\}_{i_S=1}^{n_{C_S}}$ are determined based on the maximum distance d_{max} the objects can be still perceived and the maximum number of detections $\max(n(\boldsymbol{z_i} \leftrightarrow \boldsymbol{C}_S^{i_S})$ assumed to be generated by the object class l. The parameters of the different false classes are detailed in Table II, where c_λ denotes the birth constants in (8) and (10) of the corresponding clutter class, d_{res} is the distance between the $n_{\max}$ number of Gaussian components of a series. Note that d_{res} and $n_{\max}$ are only interpreted for the classes modeled by a series of Gaussian components and not for bridges and uniformly distributed false detections. The parameters σ_x^2 and σ_y^2 in Table II define the longitudinal and lateral position variances in $\hat{\boldsymbol{P}}$ of the Gaussian components modeling the corresponding potentially false object that is not interpreted for uniformly distributed false objects. The distance PDF $p_{\text{dist}}(\hat{\boldsymbol{c}}, l)$ of newborn clutter detections with label l is described by a custom piece-wise linear distribution function based on the density values of the discrete distance intervals $[d_i, d_{i+1}] \subset V_{FOV}$.

The weight function $f_{wo}(\bar{\lambda}_l, \hat{\boldsymbol{c}})$ of a one-sample Gaussian component that computes the state-dependent Poisson rate of the component with label $l \in \{\boldsymbol{c}_O^{i_O}\}_{i_O=1}^{n_{c_O}}$ can be formed by the following linear system of equations:

$$\bar{\lambda}_l = \frac{1}{|\mathcal{D}|}\sum_{d_i \in \mathcal{D}} \bar{\lambda}([d_i, d_{i+1}]), \quad \mathcal{D} = \{d \mid d \in V_{FOV}\}\,, \tag{25}$$

$$\bar{\lambda}([d_i, d_{i+1}]) = c_{wo}\, p_{\text{dist}}\left(\frac{d_i + d_{i+1}}{2}, l\right)\,. \tag{26}$$

Although the mean value of the Gaussian components is given by the static environment descriptor of the road scenario, their initial covariance must be specified. As it was described in Section II-B, the spatial distribution of a Gaussian component $\boldsymbol{c}_S^{i_c} \in \boldsymbol{C}_S^{i_S}$ is set to be parallel with the road path. The eigenvalue σ_x^2 of the position covariance sub-matrix $\hat{\boldsymbol{P}}_{S,xy}^{i_c}$ is given by $\sigma_x^2 = (d_{\text{res}}(l_S^{i_S})/3)^2$ to obtain overlapping spatial PDFs. Whereas σ_y^2 is tuned based on the total lateral movement's average of the components $\boldsymbol{c}_S^{i_c} \in \boldsymbol{C}_S^{i_S}$. The position covariance $\hat{\boldsymbol{P}}_{O,xy}^{i_O}$ of a one-sample Gaussian component $\boldsymbol{c}_O^{i_O}$ is computed similar, but the σ_x^2 and σ_y^2 eigenvalues indicate the shape and the area of the reflecting surface A_r. The eigenvalues of the position covariance corresponding to the different object classes are detailed in Table II. The other elements of the $\hat{\boldsymbol{P}}_S^{i_c}$ and $\hat{\boldsymbol{P}}_O^{i_O}$ complete covariances are set in accordance with the parameters of the measurement model detailed in Section III-C.

C. Measurement Model

The parameters of the measurement model consist of the observation model $h_k(\boldsymbol{x}_k)$, the state estimate model $h_{s,k}(\hat{\boldsymbol{x}}_{k|k})$, and the measurement covariance $\boldsymbol{R}_k$ of the sensors. The state estimate transformation $h_{s,k}(\hat{\boldsymbol{x}}_{k|k})$ considers the measurements provided by the sensors. Besides their longitudinal and lateral positions, d_x and d_y, the smart camera provides the longitudinal velocity v_x and acceleration a_x of the objects. The radar divides both the velocity and position vector into d_x, d_y, v_x, and v_y longitudinal and lateral components, but it does not give information about the acceleration of the objects. Since the measurements of both sensors are included in the state vector $\hat{\boldsymbol{x}}_{k|k}$, the function $h_{s,k}(\hat{\boldsymbol{x}}_{k|k})$ can be described by the linear transformation:

$$\boldsymbol{z}_k = h_{s,k}(\hat{\boldsymbol{x}}_{k|k}) = \boldsymbol{H}_s \hat{\boldsymbol{x}}_{k|k}\,, \tag{27}$$

where the state transform matrices of the sensors are given as

$$\boldsymbol{H}_s^{\text{radar}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{28}$$

$$\boldsymbol{H}_s^{\text{camera}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}\,. \tag{29}$$

The observation model $h_k(\boldsymbol{x}_k)$ defines the connection between the state space and the raw measurements. In general, radars can measure objects' distance, azimuth angle, and radial velocity, while smart cameras observe their position and, in some cases,

TABLE III
ESTIMATED PARAMETERS OF THE SENSORS' MEASUREMENT MODEL

Variances	Camera	Radar
σ_x^2 [m^2]	11.8144	4.5307
σ_y^2 [m^2]	0.2041	0.2792
$\sigma_{v_x}^2$ [(m/s)^2]	2.5341	0.1201

the velocity based on optical flow. However, for simplicity, the sensors are assumed to measure the subspace of the state directly so the observation model can also be expressed as a linear transformation as

$$\boldsymbol{z}_k^{\text{sim}} = h_k(\boldsymbol{x}_k) + \boldsymbol{\eta}_k = \boldsymbol{H}\boldsymbol{x}_k + \boldsymbol{\eta}_k\,. \tag{30}$$

The observation matrix $\boldsymbol{H}^{\text{radar}}$ of the radar represents that it can measure the position of the objects, and since it is a front-looking radar, the radial velocity is estimated by the longitudinal velocity. The $\boldsymbol{H}^{\text{camera}}$ is given identically since it can directly observe the position of the objects and the longitudinal velocity based on optical flow as

$$\boldsymbol{H}^{\text{radar}} = \boldsymbol{H}^{\text{camera}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{31}$$

The observation matrix must be extended accordingly if the target sensors can measure other attributes (e.g., dimensions). The covariance, $\boldsymbol{R}_k^i$, of the measurement noise $\boldsymbol{\eta}_k^i$ is estimated based on the $\boldsymbol{x}_k^i$ reference object's state and $\boldsymbol{z}_k^i$ measurement:

$$\boldsymbol{R}_k^i = \left(\frac{1}{N} \sum_{k=1}^{N} \boldsymbol{H}(\boldsymbol{H}_s x_k^i - z_k^i)(\boldsymbol{H}_s x_k^i - z_k^i)^\top \boldsymbol{H}^\top \right), \tag{32}$$

where N_k denotes the number of cycles on which the expected value of the covariances of the objects are computed. Since the sensors provide tracked object detections, to eliminate the effect of noise filtering, N_k is set to 5, considering only the initial cycles of a track during the state estimation are not stable yet. The generic $\boldsymbol{R}_k$ measurement covariance is determined based on the average of the $\boldsymbol{R}_k^i$ object covariances, assuming that the measurements are independent as

$$\boldsymbol{R}_k = \text{diag}\left(\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{R}_k^i \right), \tag{33}$$

where n denotes the total number of objects. The variances of the sensor measurements are included in Table III.

IV. RESULTS

The simulation is validated on a different 7000-frame section of the measurement along with three aspects: the fidelity of the detection and tracking model, the similarity between the clutter density of the actual and synthetic sensor data, and the overall performance metrics of the real and simulated sensors. Since the detection and tracking model describes the detectability of the valid objects, it is evaluated by comparing the object detections provided by the simulation and the actual sensor, neglecting the clutter detections. The association between the simulated and actual detections is performed by the same GNN method used for parameter identification. Then, the performance metrics, the recall, precision, and F1 score of the detection and tracking model are computed. The recall given in (23) indicates the proportion of the object detections provided by the sensor found by the simulation too. The precision expresses how many of the simulated detections correspond to an actual sensor detection as

$$Pr = \frac{TP}{TP + FP}\,, \tag{34}$$

where FP denotes the total number of false positive detections. The comprehensive F1 score performance metric, is computed based on the precision Pr and recall Rc [84] as

$$F1 = 2\,\frac{Pr \cdot Rc}{Pr + Rc}\,. \tag{35}$$

TABLE IV
COMPARISON OF THE PROPOSED AND CONVENTIONAL DETECTION MODEL

	Camera		Radar	
Metrics	Conventional	Proposed	Conventional	Proposed
Precision	0.8265	0.8744	0.8302	0.8624
Recall	0.7922	0.8883	0.7982	0.88812
F1 Score	0.8090	0.8812	0.8142	0.8717

The performance metrics of 10 simulations provided by the proposed and a conventional Bernoulli distributed detection model are averaged and compared in Table IV. The proposed model provides false detections less frequently when the actual sensor does not detect either the object, resulting in about 3-5% higher precision for the radar and camera, respectively. According to the recall, the proposed method simulates almost 10% better when the actual sensor is able to detect the surrounding objects. Therefore, the proposed detection and tracking model provides 7.22% and 5.75% higher fidelity for the camera and radar than the conventional Bernoulli distributed model. The less significant difference in the case of the radar is due to the aforementioned reason that the parameters of the multiple scan zone detection model are more complex to be identified.

The clutter detections of the simulation do not have to match frame-by-frame perfectly the ones of the simulated sensor but their spatial distribution should be similar. Therefore, the clutter model is evaluated by comparing the $\kappa(\boldsymbol{z})$ clutter density, i.e., the frequency of the clutter occurrence in a specific part of the environment, of the proposed model, the commonly used conventional PPP, and the actual sensor detections. The $\kappa(\boldsymbol{z})$ clutter density is approximated by $\hat{\kappa}([\boldsymbol{z}_i, \boldsymbol{z}_{i+1}])$ discretized clutter map. In the evaluation, we considered the $\hat{\kappa}(\Omega_i)$ spatial distribution of the clutter density, where Ω_i denotes the grid cell given by $[d_{x,i_x}, d_{x,i_x+1}]$ longitudinal, and $[d_{y,i_y}, d_{y,i_y+1}]$ lateral position intervals. The $\hat{\kappa}(\Omega_i)$ clutter density value of the i-th grid cell is computed as

$$\hat{\kappa}(\Omega_i) = \frac{1}{N_k \cdot |\Omega_i|} \sum_{k=1}^{N_k} \sum_{\substack{\boldsymbol{c}_k \in \mathcal{C}_k: \\ \boldsymbol{c}_k \in \Omega_i}} \mathbf{1}\,, \tag{36}$$

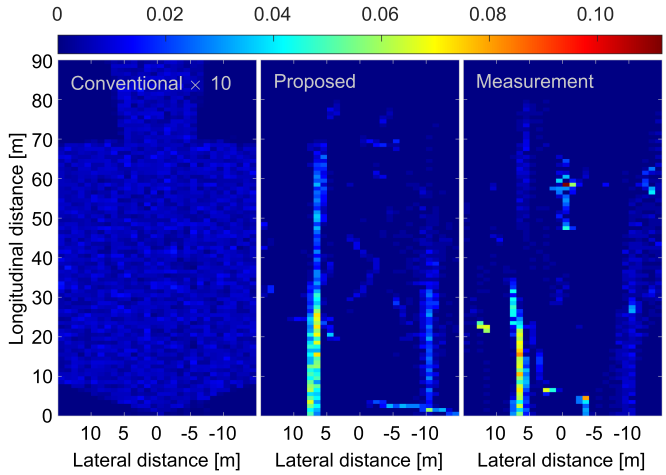


Fig. 9. The clutter density maps of the simulated (left) and the real radar detections (right).

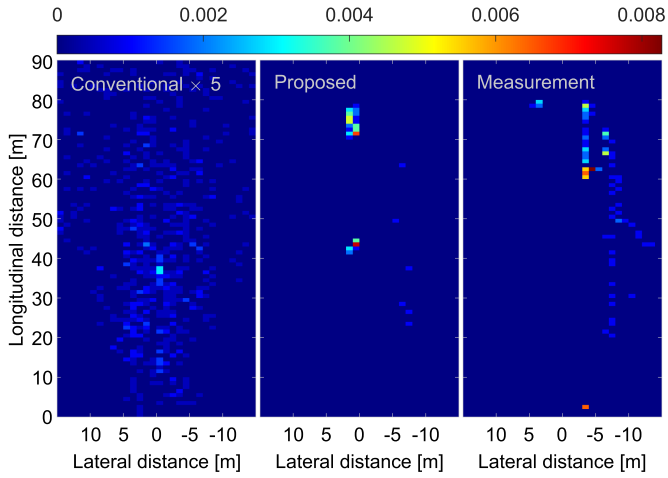


Fig. 10. The clutter density maps of the simulated (left) and the real camera detections (right).

where $\boldsymbol{c}_k \in \boldsymbol{\mathcal{C}}_k$ is a clutter detection at k-th timestamp and N_k denotes the number of frames used for the evaluation described in Section III. The density maps of the clutter detections provided by the conventional PPP (Poisson Point Process) and the proposed model are compared to the clutter density of the actual radar and camera measurements in Figs. 9 and 10. Since the density values of the conventional PPP clutter model are shallow due to the uniformly distributed detections, their density maps corresponding to the radar and camera are visualized with 10 and 5 times higher values for visibility, respectively. Fig. 9 shows qualitatively that the density map of the proposed clutter model is much more similar to the actual radar measurement than the uniformly distributed clutter detections of the conventional model. Since the camera provides few clutter detections, the clutter density values of the conventional PPP and proposed clutter model in Fig. 10 do not differ significantly. Still, the higher peaks of the proposed method seem to be more realistic than the uniform density map of the PPP model. The structural similarity index measure (SSIM) [85] is used to quantify the fidelity of the

TABLE V
STRUCTURAL SIMILARITY OF CLUTTER DENSITY MAPS

		Camera		Radar	
Range	Radius	Conventional	Proposed	Conventional	Proposed
$\max_\kappa$	1	0.8097	0.8375	0.6102	0.6065
	2	0.7576	0.7789	0.5249	0.5645
	3	0.7155	0.7357	0.4630	0.5465
	4	0.6765	0.6942	0.4174	0.5415
$[0, 1]$	1	0.9997	0.9997	0.9291	0.9384
	2	0.9998	0.9998	0.9202	0.9418
	3	0.9998	0.9998	0.9178	0.9463
	4	0.9999	0.9999	0.9174	0.9496

simulated clutter detections. The SSIM algorithm determines the quality, i.e., the similarity of digital images compared to a given reference. The clutter density maps simulated with the conventional and the proposed clutter models are compared to the density map of the actual sensors measurements as a reference. The dynamic range parameter of the SSIM defines the maximum intensity values in the input maps (e.g., 255 in the case of RGB images). If the density maps in Figs. 9 and 10 are compared as images, the dynamic range is given by the $\max_\kappa = \max(\max(\hat{\kappa}(\Omega_i^{\text{sim}})), \max(\hat{\kappa}(\Omega_i^{\text{meas}})))$ maximum simulated and measured intensity. However, this method does not consider the fact that the clutter intensity value may be greater than the dynamic range given by the highest intensity. Since the 1×1 meter size of the grid cells is smaller than the object separation distance of the sensors, the $p_c(\Omega_i) \in \{[0, 1] \subset \mathbb{R}\}$ clutter probability gives approximately the density as

$$\hat{\kappa}(\Omega_i) \approx p_c(\Omega_i) = \frac{1}{N_k \cdot |\Omega_i|} \sum_{k=1}^{N_k} \min \left(1, \sum_{\substack{\boldsymbol{c}_k \in \boldsymbol{\mathcal{C}}_k: \\ \boldsymbol{c}_k \in \Omega_i}} \mathbf{1} \right). \quad (37)$$

Therefore, the dynamic range interval [0,1] can be applied for SSIM providing a more meaningful measure of clutter model's fidelity. Furthermore, we applied four different Gaussian weighting circles in SSIM. The higher the radius of the circle, the more neighboring pixels are looked at in the similarity measure, considering the structure of the map instead of pixel-wise similarity. The similarity metrics of the conventional PPP and proposed clutters model are detailed in Table V, including the results of the two dynamic ranges and radius values of four weighting circles. Since the camera provides few clutter detections, the clutter density map of the actual camera measurement, consisting of many 0 intensity values, is easier to be simulated. Therefore, the similarity metrics in the dynamic range $[0, 1]$ show a high, more than 99% fidelity of both the conventional and the proposed clutter model. However, the clutter density map of the proposed model, considering the tracking of the false detections, is $\approx$2% more similar to the simulated camera in the $[0, \max_\kappa]$ range. The radar has a much more complex non-uniform clutter model resulting in lower similarity metrics, particularly in the case of $\max_\kappa$ range. Still, the difference between the proposed and conventional PPP clutter model is more significant, increasing

TABLE VI
PERFORMANCE METRICS OF THE REAL AND SIMULATED SENSORS

	Camera			Radar		
	Real	Simulation		Real	Simulation	
Metrics		Conv.	Prop.		Conv.	Prop.
Precision	0.9730	0.9812	0.9869	0.4724	0.5524	0.4748
Recall	0.7327	0.6824	0.7338	0.8698	0.8364	0.8593
F1 Score	0.8359	0.8050	0.8417	0.6122	0.6654	0.6114

with the circle radius. It means that the clutter density map of the proposed model is significantly closer to the actual radar data than the uniformly distributed detections of the PPP model regarding their structure. Furthermore, the similarity values of the proposed clutter model are greater than 93% for all radius values in the [0,1] range representing the fidelity of the clutter detections better than with $\max_{\kappa}$, considering the valid range of clutter intensity values.

Finally, the overall performance metrics of the simulated sensor data, namely the aforementioned recall, the precision, and the F1 score, are compared to the same metrics of the actual sensor data. The averaged metrics of 10 simulations are compared to the actual sensors in Table VI, including the proposed model and a conventional one using Bernoulli distributed detection and PPP clutter model. According to the F1 scores, both sensors are simulated with high fidelity by the proposed method since the differences to the actual sensors are less than 1% in contrast with the conventional model that under- and overestimates the camera and radar performance, respectively. The precision related to the false positive detections is more realistic for the radar simulation than for the camera since the latter considers only the clutter detections due to unknown reasons, i.e., partially uniform clutter appearance. The precision of the radar data simulated by the proposed model is much closer to the actual precision than the conventional model, indicating that the proposed clutter model is more reliable than the commonly used PPP clutter model. In contrast, the camera's recall estimation is more reliable because the radar has a complex detection model consisting of multiple scan zones. The proposed model considering the built-in tracking module of the actual sensors, achieves 5% better camera recall in simulation than the conventional model using Bernoulli distribution. Both the precision and recall of the sensors simulated by the proposed data-driven sensor model have more than 98% fidelity.

The fidelity of the radar clutter model is confirmed by Figs. 11 and 12 demonstrating the output of guardrail, lamp pole, and highway bridge simulations on a frame. Fig. 11 shows that both the spatial and cardinality distribution of the false positive objects generated by the guardrail accurately match the detections provided by the clutter model for guardrails. It also exposes the simulated radar detections corresponding to street lamp poles that do not precisely match the actual sensor measurement. This is anticipated since they are less frequently located along the road path and the radar cross-section, so the detection probability is

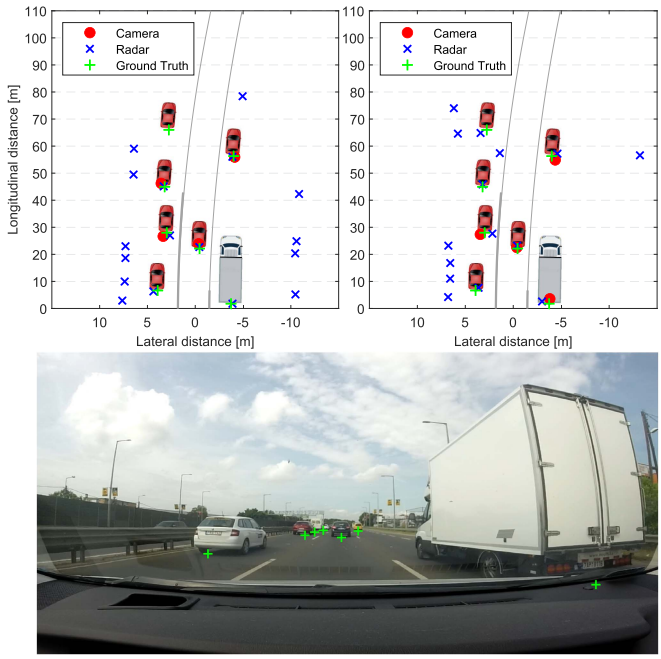


Fig. 11. Qualitative result illustrating the simulated (left) and real (right) clutter detections generated by guardrail and street lamp poles.

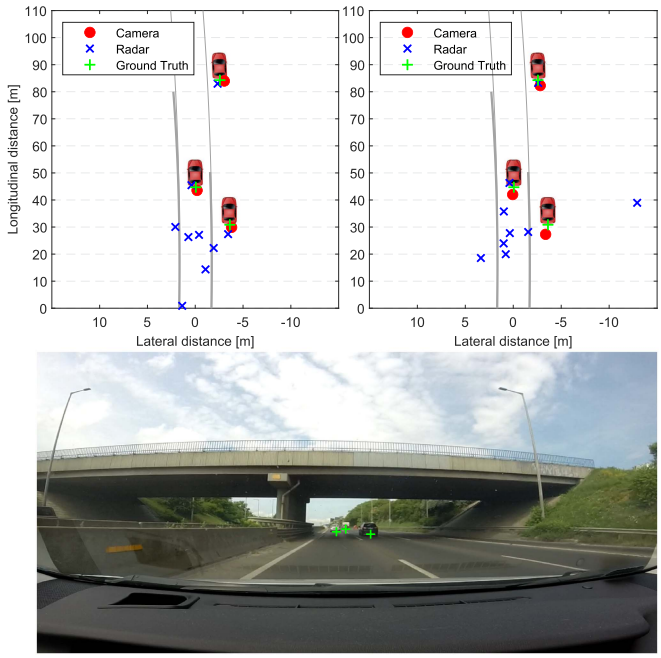


Fig. 12. Qualitative result illustrating the simulated (left) and real (right) clutter detections generated by highway bridge.

lower compared to guardrails. Fig. 12 shows an example of highway bridge clutter simulation. The number of simulated bridge detections gives a suitable estimate of the detections observed by the actual sensor. Although the position of the points reflected on the bridge does not fit precisely the measured reflections, the simulation still provides a fair estimation. Furthermore, since the clutters are simulated according to a random distribution,

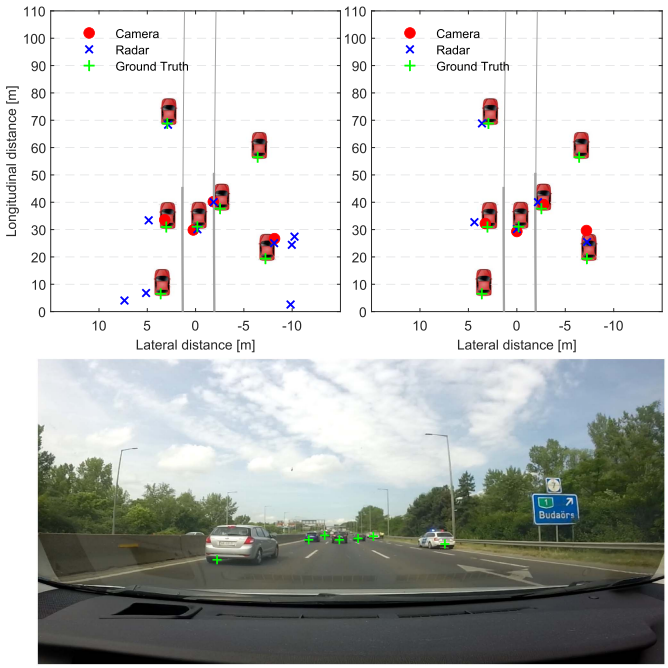


Fig. 13. Qualitative result illustrating the simulated (left) and real (right) detections generated by standstill objects in emergency lane.

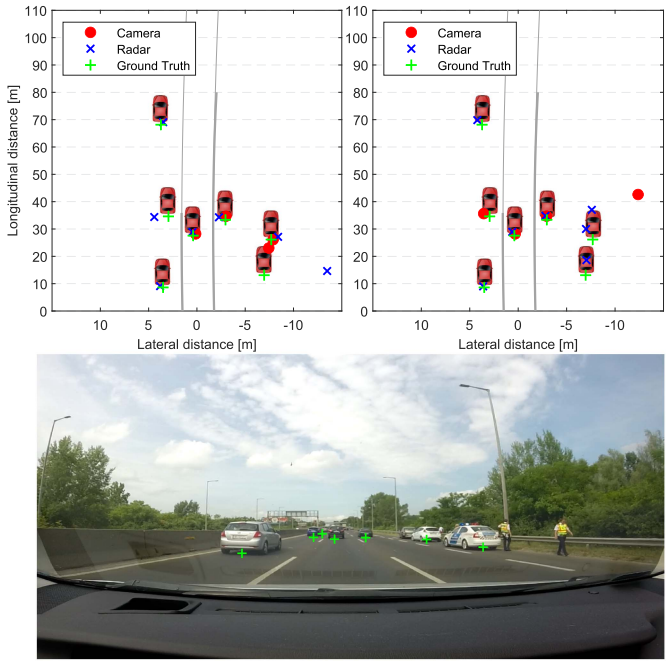


Fig. 14. Qualitative result illustrating the simulated (left) and real (right) detections generated by standstill objects in emergency lane.

the result should be evaluated statistically. The detection model performance and its limitations in a more complex scenario involving standstill objects in the emergency lane are illustrated in Figs. 13 and 14. The detectability of the police car detected by both the actual radar and camera is simulated correctly, while the farther car is missed by the real sensors and the simulator as well shown in Fig. 13. Moreover, the measurement error regarding the position of the detections is also simulated accurately. The slight difference in Fig. 13 is coming from the simulated clutter detections assumed to be generated by the lamp poles. However, as mentioned earlier, it does not distort the simulation data significantly if the clutter model does not match the detections of the actual sensors frame-by-frame, but statistically, they are similar. The limitations of the proposed detection model are illustrated in Fig. 14 showing that the radar detects both standstill cars in the emergency lane but the simulator assumes the closer one undetected. Furthermore, despite the simulated camera detecting the police car, the real sensor misses both standstill objects, and false detections occurs over the guardrail. The pedestrians appearing in Fig. 14 are not considered in the simulation nor the evaluation because of the lack of pedestrian detection in the training data used for parameter identification, which is also a limitation of the current status of the proposed method. Although the simulation does not perfectly reconstruct frame-by-frame the detections of the actual sensors due to the limitations of the proposed method, statistically, it provides high-fidelity synthetic data. Using this simulated data for testing, one can evaluate and further improve the performance of the environment perception algorithms in different scenarios, increasing the safety of the ADAS and HAD functions.

V. Conclusion

The proposed sensor model can simulate the object-level data, i.e., detections or tracked objects, of a generic smart sensor with high fidelity based on the data-driven identification of the model parameters. The simulator tackles tracked object detections as well by involving a tracking model that extends the detection model fitted to the actual sensor. The measurement model can also handle tracked objects simulating the measurement uncertainties of the sensors. The clutter model reproduces the clutters provided by the sensor with high accuracy regarding their cardinality and spatial distribution, considering different types of false detections. Since the surrounding scenario- and static environmental objects are usually provided by the commonly used 3D simulation environments on which the proposed model relies, the simulator can generate the sensor data in an arbitrary road scenario. Therefore, the simulator can support the development and testing of different environment perception modules, such as sensor data fusion, even in corner cases, increasing the safety of ADAS and HAD functions. We intend to improve the proposed model with a more detailed radar and camera clutter model considering multi-path detections and digital image processing failures that are now handled by the partially uniformly distributed clutters due to unknown reasons. Furthermore, we plan to extend the proposed simulation with the consideration of object occlusion in the detection and tracking model.

References

[1] V. Gružauskas, S. Baskutis, and V. Navickas, "Minimizing the trade-off between sustainability and cost effective performance by using autonomous vehicles," *J. Cleaner Prod.*, vol. 184, pp. 709–717, 2018.

[2] N. Krizsik and T. Sipos, "Social perception of autonomous vehicles," *Periodica Polytechnica Transp. Eng.*, vol. 51, no. 2, pp. 133–139, 2023. [Online]. Available: https://pp.bme.hu/tr/article/view/20228

[3] T. Tettamanti, I. Varga, and Z. Szalay, "Impacts of autonomous cars from a traffic engineering perspective," *Periodica Polytechnica Transp. Eng.*, vol. 44, no. 4, pp. 244–250, 2016.

[4] M. Lu, "Modelling the effects of road traffic safety measures," *Accident Anal. Prevention*, vol. 38, no. 3, pp. 507–517, 2006.

[5] J. Vargas, S. Alsweiss, O. Toker, R. Razdan, and J. Santos, "An overview of autonomous vehicles sensors and their vulnerability to weather conditions," *Sensors*, vol. 21, no. 16, 2021, Art. no. 5397.

[6] H. Zhu, K. V. Yuen, L. Mihaylova, and H. Leung, "Overview of environment perception for intelligent vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 10, pp. 2584–2601, Oct. 2017.

[7] J. Kocić, N. Jovičić, and V. Drndarević, "Sensors and sensor fusion in autonomous vehicles," in *Proc. IEEE 26th Telecommun. Forum*, 2018, pp. 420–425.

[8] W. Elmenreich, "An introduction to sensor fusion," Vienna University of Technology, Austria, Tech. Rep. 47/2001, 2002.

[9] M. T. Horváth, Q. Lu, T. Tettamanti, Á. Török, and Z. Szalay, "Vehicle-in-the-loop (VIL) and scenario-in-the-loop (SCIL) automotive simulation concepts from the perspectives of traffic simulation and traffic control," *Transp. Telecommun.*, vol. 20, no. 2, pp. 153–161, 2019.

[10] X. Ma, X. Hu, T. Weber, and D. Schramm, "Traffic simulation of future intelligent vehicles in duisburg city inner ring," *Appl. Sci.*, vol. 11, no. 1, 2021, Art. no. 29.

[11] W. Li et al., "AADS: Augmented autonomous driving simulation using data-driven algorithms," *Sci. Robot.*, vol. 4, no. 28, 2019, Art. no. eaaw0863.

[12] T. Duy Son, A. Bhave, and H. Van der Auweraer, "Simulation-based testing framework for autonomous driving development," in *Proc. IEEE Int. Conf. Mechatronics*, 2019, pp. 576–583.

[13] B. Huang, H. Xiong, J. Wang, Q. Xu, X. Li, and K. Li, "Detection-level fusion for multi-object perception in dense traffic environment," in *Proc. IEEE Int. Conf. Multisensor Fusion Integration Intell. Syst.*, 2017, pp. 411–416.

[14] B. Duraisamy, T. Schwarz, and C. Wöhler, "Track level fusion algorithms for automotive safety applications," in *Proc. IEEE Int. Conf. Signal Process., Image Process. Pattern Recognit.*, 2013, pp. 179–184.

[15] M. Wen, J. Park, Y. Sung, Y. W. Park, and K. Cho, "Virtual scenario simulation and modeling framework in autonomous driving simulators," *Electronics*, vol. 10, no. 6, 2021, Art. no. 694. [Online]. Available: https://www.mdpi.com/2079-9292/10/6/694

[16] X. Li, S. Teng, B. Liu, X. Dai, X. Na, and F.-Y. Wang, "Advanced scenario generation for calibration and verification of autonomous vehicles," *IEEE Trans. Intell. Veh.*, vol. 8, no. 5, pp. 3211–3216, May 2023.

[17] M. R. Zofka, S. Klemm, F. Kuhnt, T. Schamm, and J. M. Zöllner, "Testing and validating high level components for automated driving: Simulation framework for traffic scenarios," in *Proc. IEEE Intell. Veh. Symp.*, 2016, pp. 144–150.

[18] B. Schlager et al., "State-of-the-art sensor models for virtual testing of advanced driver assistance systems/autonomous driving functions," *SAE Int. J. Connected Automated Veh.*, vol. 3, no. 3, pp. 233–261, 2020.

[19] L. Li, W.-L. Huang, Y. Liu, N.-N. Zheng, and F.-Y. Wang, "Intelligence testing for autonomous vehicles: A new approach," *IEEE Trans. Intell. Veh.*, vol. 1, no. 2, pp. 158–166, Jun. 2016.

[20] R. Molenaar, A. v. Bilsen, R. van der Made, and R. de Vries, "Full spectrum camera simulation for reliable virtual development and validation of ADAS and automated driving applications," in *Proc. IEEE Intell. Veh. Symp.*, 2015, pp. 47–52.

[21] P. Rosenberger et al., "Towards a generally accepted validation methodology for sensor models-challenges, metrics, and first results," in *Proc. Graz Symp. Virtual Veh.*, 2019, pp. 1–13.

[22] M. O'Kelly, A. Sinha, H. Namkoong, R. Tedrake, and J. C. Duchi, "Scalable end-to-end autonomous vehicle testing via rare-event simulation," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2018, pp. 1–12.

[23] M. Herrmann and H. Schön, "Efficient sensor development using raw signal interfaces," in *Proc. Fahrerassistenzsysteme*, 2019, pp. 30–39.

[24] E. Egea-Lopez, F. Losilla, J. Pascual-Garcia, and J. M. Molina-Garcia-Pardo, "Vehicular networks simulation with realistic physics," *IEEE Access*, vol. 7, pp. 44021–44036, 2019.

[25] J. R. V. Rivero, T. Gerbich, B. Buschardt, and J. Chen, "The Effect of Spray Water on an Automotive LIDAR Sensor: A Real-Time Simulation Study," *IEEE Trans. Intell. Veh.*, vol. 7, no. 1, pp. 57–72, Mar. 2022.

[26] S. O. Wald and F. Weinmann, "Ray tracing for range-doppler simulation of 77GHz automotive scenarios," in *Proc. IEEE 13th Eur. Conf. Antennas Propag.*, 2019, pp. 1–4.

[27] U. Chipengo, A. Sligar, and S. Carpenter, "High fidelity physics simulation of 128 channel MIMO sensor for 77GHz automotive radar," *IEEE Access*, vol. 8, pp. 160643–160652, 2020.

[28] F. M. Maier, V. P. Makkapati, and M. Horn, "Environment perception simulation for radar stimulation in automated driving function testing," *E I Elektrotechnik und Informationstechnik*, vol. 135, no. 4, pp. 309–315, 2018.

[29] K. Majek and J. Bedkowski, "Range sensors simulation using GPU ray tracing," in *Proc. 9th Int. Conf. Comput. Recognit. Syst.*, 2016, pp. 831–840.

[30] Q. Yue, Y. Jia, and Z. Qiu, "Reasearch on spaceborn TDI CCD camera imaging simulation based on monte-carlo ray tracing," in *Proc. IEEE Int. Conf. Remote Sens., Environ. Transp. Eng.*, 2011, pp. 4287–4290.

[31] M. S. P. Degerman and J. Pohl, "Ultrasonic sensor modeling for automatic parallel parking systems in passenger cars," SAE Technical Paper, SAE International, 2007-01-1103, 2007.

[32] J. Thieling and J. Roßmann, "Highly-scalable and generalized sensor structures for efficient physically-based simulation of multi-modal sensor networks," in *Proc. IEEE 12th Int. Conf. Sens. Technol.*, 2018, pp. 202–207.

[33] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles," in *Field and Service Robotics*. Berlin, Germany: Springer, 2018, pp. 621–635.

[34] Vector (TESIS), "DYNA4 driver assistance: ADAS development with real-time simulation," 2023. [Online]. Available: https://www.vector.com/int/en/products/products-a-z/software/dyna4/

[35] Hexagon (VIRES), "VTD - virtual test drive," 2023. [Online]. Available: https://vires.mscsoftware.com/solutions/sensors/

[36] dSPACE, "Automotive simulation models (ASM)," 2023. [Online]. Available: https://www.dspace.com/en/inc/home/products/sw/automotive_simulation_models.cfm

[37] IPG Automotive, "CarMaker," 2023. [Online]. Available: https://ipg-automotive.com/en/products-solutions/software/carmaker/

[38] Siemens (TASS Intertational), "PreScan," 2023. [Online]. Available: https://www.plm.automation.siemens.com/global/en/products/simulation-test/active-safety-system-simulation.html

[39] Epic Games, "Unreal engine," 2023. [Online]. Available: https://www.unrealengine.com

[40] J. K. Haas, "A history of the unity game engine," Master's thesis, Worcester Polytechnic Institute, Worcester, MA, USA, 2014.

[41] J. E. Farrell, P. B. Catrysse, and B. A. Wandell, "Digital camera simulation," *Appl. Opt.*, vol. 51, no. 4, pp. A80–A90, Feb. 2012.

[42] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez, "The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3234–3243.

[43] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. Conf. Robot Learn.*, 2017, pp. 1–16 .

[44] T. Stel and J. Roßmann, "A virtual reality testbed for camera simulation in aerospace applications," in *Proc. IEEE 3rd Int. Conf. Artif. Intell., Modelling Simul.*, 2015, pp. 129–134.

[45] A. Elmquist and D. Negrut, "Methods and models for simulating autonomous vehicle sensors," *IEEE Trans. Intell. Veh.*, vol. 5, no. 4, pp. 684–692, Dec. 2020.

[46] N. Peinecke, T. Lueken, and B. R. Korn, "Lidar simulation using graphics hardware acceleration," in *Proc. IEEE/AIAA 27th Digit. Avionics Syst. Conf.*, 2008, pp. 4.D.4-1–4.D.4-8.

[47] B. Hurl, K. Czarnecki, and S. Waslander, "Precise synthetic image and LiDAR (PreSIL) dataset for autonomous vehicle perception," in *Proc. IEEE Intell. Veh. Symp.*, 2019, pp. 2522–2529.

[48] X. Yue, B. Wu, S. A. Seshia, K. Keutzer, and A. L. Sangiovanni-Vincentelli, "A LiDAR point cloud generator: From a virtual world to autonomous driving," in *Proc. ACM Int. Conf. Multimedia Retrieval*, 2018, pp. 458–464.

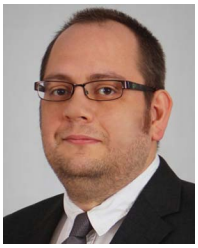
[49] Z. F. Magosi, H. Li, P. Rosenberger, L. Wan, and A. Eichberger, "A survey on modelling of automotive radar sensors for virtual test and validation of automated driving," *Sensors*, vol. 22, no. 15, 2022, Art. no. 5693.

[50] A. Ngo, M. P. Bauer, and M. Resch, "A sensitivity analysis approach for evaluating a radar simulation for virtual testing of autonomous driving functions," in *Proc. IEEE 5th Asia-Pacific Conf. Intell. Robot Syst.*, 2020, pp. 122–128.

[51] T. Machida and T. Owaki, "Rapid and precise millimeter-wave radar simulation for adas virtual assessment," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2019, pp. 431–436.
[52] D. Negrut, R. Serban, and A. Elmquist, "Physics-based sensor models for virtual simulation of connected and autonomous vehicles," Safety Research Using Simulation (SAFER-SIM) University Transportation Center, University of Wisconsin-Madison, Tech. Rep., 2020.
[53] E. Bel Kamel, A. Peden, and P. Pajusco, "RCS modeling and measurements for automotive radar applications in the W band," in *Proc. IEEE 11th Eur. Conf. Antennas Propag.*, 2017, pp. 2445–2449.
[54] S. C. Schnelle, M. K. Salaani, S. J. Rao, F. S. Barickman, and D. Elsasser, "Review of simulation frameworks and standards related to driving scenarios," Dept. Transp. Nat. Highway Traffic Safety Admin., Washington, D.C., USA, Tech. Rep. DOT HS 812 815, 2019.
[55] M. F. Holder, "Synthetic generation of radar sensor data for virtual validation of autonomous driving," Ph.D. dissertation, Technische Universität Darmstadt, Darmstadt, Germany, 2021.
[56] H. Wu., "Simulation of Radar Signal Processing Based on Matlab," in *Proc. Inf. Sci. Manage. Eng. IV*, 2016, pp. 299–304.
[57] G. Hakobyan and B. Yang, "High-performance automotive radar: A review of signal processing algorithms and modulation schemes," *IEEE Signal Process. Mag.*, vol. 36, no. 5, pp. 32–44, Sep. 2019.
[58] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3D object detection methods for autonomous driving applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3782–3795, Oct. 2019.
[59] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 4, pp. 1773–1795, Dec. 2013.
[60] S. Zhang, C. Wang, S.-C. Chan, X. Wei, and C.-H. Ho, "New object detection, tracking, and recognition approaches for video surveillance over camera network," *IEEE Sensors J.*, vol. 15, no. 5, pp. 2679–2691, May 2015.
[61] Y. Li and J. Ibanez-Guzman, "Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems," *IEEE Signal Process. Mag.*, vol. 37, no. 4, pp. 50–61, Jul. 2020.
[62] T. Hanke, N. Hirsenkorn, B. Dehlink, A. Rauch, R. Rasshofer, and E. Biebl, "Classification of sensor errors for the statistical simulation of environmental perception in automated driving systems," in *Proc. IEEE 19th Int. Conf. Intell. Transp. Syst.*, 2016, pp. 643–648.
[63] J. M. R. Velázquez, F. Mailly, and P. Nouet, "A generic model for sensor simulation at system level," in *Proc. Symp. Des., Test, Integration Packag. MEMS MOEMS DTIP*, 2018, pp. 1–4.
[64] N. Hirsenkorn, T. Hanke, A. Rauch, B. Dehlink, R. Rasshofer, and E. Biebl, "A non-parametric approach for modeling sensor behavior," in *Proc. IEEE 16th Int. Radar Symp.*, 2015, pp. 131–136.
[65] T. Hanke, N. Hirsenkorn, B. Dehlink, A. Rauch, R. Rasshofer, and E. Biebl, "Generic architecture for simulation of adas sensors," in *Proc. IEEE 16th Int. Radar Symp.*, 2015, pp. 125–130.
[66] M. Buhren and B. Yang, "Simulation of automotive radar target lists using a novel approach of object representation," in *Proc. IEEE Intell. Veh. Symp.*, 2006, pp. 314–319.
[67] S. Bernsteiner, Z. Magosi, D. Lindvai-Soos, and A. Eichberger, "Radar sensor model for the virtual development process," *ATZelektronik worldwide*, vol. 10, no. 2, pp. 46–52, 2015.
[68] M. Buhren and B. Yang, "Extension of automotive radar target list simulation to consider further physical aspects," in *Proc. IEEE 7th Int. Conf. ITS Telecommun.*, 2007, pp. 1–6.
[69] M. Bühren and B. Yang, "Simulation of automotive radar target lists considering clutter and limited resolution," in *Proc. IEEE Int. Radar Symp.*, 2007, pp. 195–200.
[70] S. Muckenhuber, H. Holzer, J. Rübsam, and G. Stettinger, "Object-based sensor model for virtual testing of ADAS/AD functions," in *Proc. IEEE Int. Conf. Connected Veh. Expo*, 2019, pp. 1–6.
[71] Y. Bar-Shalom and E. Tse, "Tracking in a cluttered environment with probabilistic data association," *Automatica*, vol. 11, no. 5, pp. 451–460, 1975.
[72] J. K. Tugnait, "Tracking of multiple maneuvering targets in clutter using multiple sensors, IMM, and JPDA coupled filtering," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 40, no. 1, pp. 320–330, Jan. 2004.
[73] F. García, A. Prioletti, P. Cerri, and A. Broggi, "PHD filter for vehicle tracking based on a monocular camera," *Expert Syst. Appl.*, vol. 91, pp. 472–479, 2018.
[74] K. Granström, L. Svensson, S. Reuter, Y. Xia, and M. Fatemi, "Likelihood-based data association for extended object tracking using sampling methods," *IEEE Trans. Intell. Veh.*, vol. 3, no. 1, pp. 30–45, Mar. 2018.
[75] O. Törő, T. Bécsi, and P. Gáspár, "PHD filter for object tracking in road traffic applications considering varying detectability," *Sensors*, vol. 21, no. 2, 2021, Art. no. 472.
[76] X. Chen, Y. Li, Y. Li, and J. Yu, "PHD and CPHD algorithms based on a novel detection probability applied in an active sonar tracking system," *Appl. Sci.*, vol. 8, no. 1, 2018, Art. no. 36.
[77] G. Hendeby and R. Karlsson, "Gaussian mixture PHD filtering with variable probability of detection," in *Proc. IEEE FUSION - 17th Int. Conf. Inf. Fusion*, 2014, pp. 1–7.
[78] S. Wang, Q. Bao, and Z. Chen, "Refined PHD filter for multi-target tracking under low detection probability," *Sensors*, vol. 19, no. 13, 2019, Art. no. 2842.
[79] D. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. Autom. Control*, vol. 24, no. 6, pp. 843–854, Dec. 1979.
[80] F. García, A. de la Escalera, and J. M. Armingol, "Joint probabilistic data association fusion approach for pedestrian detection," in *Proc. IEEE Intell. Veh. Symp.*, 2013, pp. 1344–1349.
[81] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Trans. ASME–J. Basic Eng.*, vol. 82, no. Series D., pp. 35–45, 1960.
[82] D. Simon, *Optimal State Estimation: Kalman, H. Infinity, and Nonlinear Approaches*. Hoboken, NJ, USA: Wiley, 2006.
[83] P. Konstantinova, A. Udvarev, and T. Semerdjiev, "A study of a target tracking algorithm using global nearest neighbor approach," in *Proc. Int. Conf. Comput. Syst. Technol.*, 2003, pp. 290–295.
[84] L. Kovacs, L. Lindenmaier, H. Nemeth, V. Tihanyi, and A. Zarandy, "Performance evaluation of a track to track sensor fusion algorithm," in *Proc. 16th Int. Workshop Cellular Nanoscale Netw. Appl.*, 2018, pp. 1–2.
[85] Z. Wang, A. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

László Lindenmaier received the B.Sc. and M.Sc. degrees from the Budapest University of Technology and Economics, Budapest, Hungary, in 2017 and 2020, respectively, where he is currently working toward the Ph.D. degree with the Department of Control for Transportation and Vehicle Systems. Between 2017 and 2020, he worked in the automotive industry as a development engineer in field of environment perception and sensor data fusion. His research interests include vehicle control, automotive environment perception, SLAM algorithms, object detection, Bayesian state estimation, multi-object tracking, and sensor data fusion.

Szilárd Aradi (Member, IEEE) received the M.Sc. and Ph.D. degrees from the Budapest University of Technology and Economics, Budapest, Hungary, in 2005 and 2015, respectively. Since 2016, he has been a Senior Lecturer with the Department of Control for Transportation and Vehicle Systems, Budapest University of Technology and Economics. He is currently working with the Department of Control for Transportation and Vehicle Systems, Budapest University of Technology and Economics. His research interests include embedded systems, communication networks, vehicle mechatronics, and reinforcement learning. His research and industrial works have involved railway information systems, vehicle on-board networks, and vehicle control.



Tamás Bécsi (Member, IEEE) received the M.Sc. and Ph.D. degrees from the Budapest University of Technology and Economics, Budapest, Hungary, in 2002 and 2008, respectively. Since 2005, he has been an Assistant Lecturer and since 2014, he has also been an Associate Professor with the Department of Control for Transportation and Vehicle Systems, Budapest University of Technology and Economics. His research interests include linear systems, embedded systems, traffic modeling, and simulation. His research and industrial works have involved railway information systems and vehicle control.

Olivér Törő received the M.Sc. degree from the Eötvös Loránd University, Budapest, Hungary, in 2010 and the Ph.D. degree from the Budapest University of Technology and Economics, Budapest, in 2022. Since 2018, he has been an Assistant Research Fellow with the Department of Control for Transportation and Vehicle Systems, Budapest University of Technology and Economics. His research interests include object detection and tracking in road traffic applications, multi-object state estimation, and nonlinear filtering.

Péter Gáspár received the M.Sc. and Ph.D.degrees from the Faculty of Transportation Engineering and Vehicle Engineering (KJK), Budapest University of Technology and Economics (BME), Budapest, Hungary, in 1985 and 1997, respectively, and the D.Sc. degree in control from the Hungarian Academy of Sciences, Budapest, in 2007. Since 1990, he has been a Senior Research Fellow with the Institute for Computer Science and Control and Since 2016, he has also been a Research Professor. In 2004, he became the Head of the Vehicle Dynamics and Control Research Group and then in 2017, he became the Head of the Systems and Control Laboratory, SZTAKI. He was habilitated at the BME, in 2008, and he was appointed as the University Professor. Since 2013, he has also been the Head with the Department of Control for Transportation and Vehicle Systems (KJIT), BME KJK. His research interests include linear and nonlinear systems, robust control, multi-objective control, system identification, and identification for control and artificial methods. His research and industrial works have involved mechanical systems, vehicle structures, and vehicle dynamics and control. Since 2016, he has also been a Corresponding Member of MTA. He is also a Member of the IFAC Automotive Control and Transportation Systems Technical Committee, and Chair of the International Federation of Automatic Control (IFAC) Hungary National Member Organization.