An Overtaking Decision Algorithm for Networked Intelligent Vehicles based on Cooperative Perception

Milos Vasic   Gael Lederrey   Iñaki Navarro   Alcherio Martinoli

Abstract— This paper presents an overtaking decision algorithm for networked intelligent vehicles. The algorithm is based on a cooperative tracking and sensor fusion algorithm that we previously developed. The ego vehicle is equipped with lane keeping and lane changing capabilities, as well as a forward-looking lidar sensor. The lidar data are fed to the tracking module which detects other vehicles, such as the vehicle that is to be overtaken (leading) and the oncoming traffic. Based on the estimated distances to the leading and the oncoming vehicles and their speeds, a risk is calculated and a corresponding overtaking decision is made. We compare the performance of the overtaking algorithm between the case when the ego vehicle only relies on its lidar sensor, and the case in which it fuses object estimates received from the leading car which also has a forward-looking lidar. Systematic evaluations are performed in Webots, a calibrated high-fidelity simulator.

Fig. 1.   Example of an overtaking scenario.

I. INTRODUCTION

Overtaking other vehicles accounts among the leading causes of motor vehicle accidents. They are known to have a high risk for a fatal outcome. In 2013, there was a total of 81000 accidents while overtaking in the USA [1]. These accidents can be, among other reasons, due to a wrong driver's judgment or impaired visibility conditions.

It is well known that distances to remote objects and their relative velocities can be estimated more accurately using sensors than by bare eyes. Vehicles' built-in intelligence can therefore be a useful aid to human drivers in these situations. However, there exist situations in which individual intelligent vehicles cannot perform very well: no matter what sensor technology is used, all sensors suffer from limited range, Field Of View (FOV), or non line-of-sight conditions to some extent. Hence, cooperation of vehicles by the means of exchanging the sensing data or track estimates can be advantageous.

In this paper, we consider a sample scenario depicted in Fig. 1. It introduces three types of cars: the ego ($\mathbb{E}$), the leading ($\mathbb{L}$) and the oncoming ($\mathbb{O}$) cars. The car $\mathbb{E}$ is an intelligent vehicle which contains sensors enabling it to perceive the environment and estimate its state. It processes the information and, once it encounters a slower car ahead, assesses the risk of overtaking and makes an overtaking decision. The car $\mathbb{L}$ is any car that finds itself driving in front of the car $\mathbb{E}$, on the same lane at lower speed. It may be equipped with sensors and may share its information with the car $\mathbb{E}$ through communication. The car $\mathbb{O}$ is any car that drives
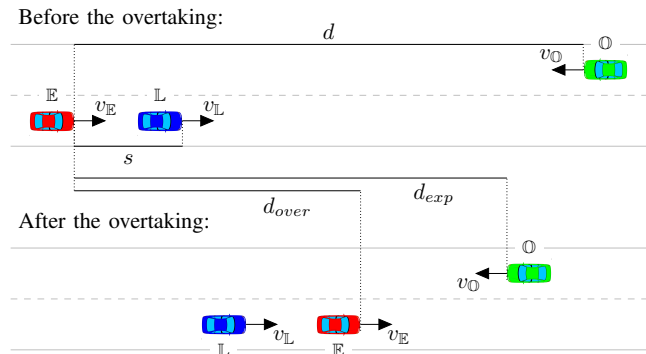
on the passing lane of the car $\mathbb{E}$, in the opposite direction. It possesses no senors nor communication equipment, and it represents a potential hindrance to the overtaking maneuver.

In this paper we present an algorithm which assesses the overtaking risk and decides if the car $\mathbb{E}$ can overtake the car $\mathbb{L}$ safely. The algorithm relies on information provided by a tracking and fusion algorithm previously presented in [2]. The cars $\mathbb{E}$ and $\mathbb{L}$ run the tracking algorithm, each using a forward-looking lidar sensor. The car $\mathbb{E}$ additionally fuses the information received from the car $\mathbb{L}$, if available. We consider two cases: the case in which the car $\mathbb{L}$ cooperates with the car $\mathbb{E}$ by communicating its information, and the case in which it does not.

The rest of the paper is organized as follows: Sec. II presents related work in the field. Sec. III summarizes the cooperative tracking and fusion algorithm, whereas Sec. IV presents the overtaking algorithm. Experimental evaluation is provided in Sec. V, while Sec. VI concludes the paper.

II. RELATED WORK

Cooperative overtaking assistance systems are readily studied in the literature. In [3] a system which predicts when a potential lane-change is going to be performed by the driver of the ego-vehicle, cooperatively exchanges information with vehicles traveling in the opposite direction via the cellular network, and provides the driver with an estimate of the overtaking maneuver risk. It assumes that all oncoming vehicles are cooperative. Another assistance system is based on real-time video transmission [4]. This approach requires a high communication bandwidth. A cooperative overtaking assist system using an intelligent road surface is introduced in [5]. It requires investments in the infrastructure.

In our approach overtaking assistance is based on the

information obtained from on-board sensors, which is fed to the target tracking algorithm. The cooperative fusion algorithm is designed to minimize the communication bandwidth requirements (instead of raw data, tracks are communicated). One possible approach for multi-target tracking is a Probability Hypothesis Density (PHD) filter, a generalization of the single-target Bayes filter. Its two well known implementations are based on a Gaussian Mixture (GM) model [6], and a Sequential Monte Carlo (SMC) model [7]. Works on fusing together the PHD intensities originating from different sources exist: an approach for GM-PHD intensities is given in [8] and for SMC-PHD intensities in [9]. Common for these works is the assumption that there exist a unique FOV (area) that is covered by all participating sensors simultaneously.

III. COOPERATIVE GAUSSIAN MIXTURE PHD FILTER

To perform cooperative tracking of multiple targets (i.e., cars) using laser sensors, we use the Cooperative Gaussian Mixture PHD (C-GM-PHD) filter described in [2]. We summarize here the algorithm for completeness.

A. Multiple target tracking using a lidar sensor

The tracking algorithm works in two phases. The first phase pre-processes the laser point cloud with the aim to detect objects that resemble cars. First, the clustering of the point cloud is performed using the DBSCAN algorithm [10]. Then, lines and corners are fitted to clusters, and the one with smaller RMS error is preserved for each cluster. Finally, rectangles are fitted to lines and corners, and feature points are extracted. The set of object measurements is generated, where an object measurement is characterized using the center and orientation of a fitted rectangle, $z = [x, y, \theta]^\top$.

The second phase represents the tracking algorithm based on the Gaussian Mixture Probability Hypothesis Density (GM-PHD) filter [6]. We model a target state using a vector $\mathbf{x} = [x, y, \theta, v, \omega]^\top$, where x and y represent the center of the tracked vehicle, θ its orientation, and v and ω its linear and rotational speed. Targets' state hypotheses are called intensity and are modeled as a Gaussian Mixture. At time $k-1$ the intensity containing J_{k-1} components with weights w_{k-1}, means m_{k-1} and covariances P_{k-1}, where the weight of a Gaussian component represents the number of targets that are represented using that component, has the form

$$D_{k-1}(\mathbf{x}) = \sum_{i=1}^{J_{k-1}} w_{k-1}^{(i)} \mathcal{N}(\mathbf{x}; m_{k-1}^{(i)}, P_{k-1}^{(i)}) \quad (1)$$

The filter contains the predict and the update step, in which we use an Unscented Kalman Filter to deal with non-linearities. The motion model used in the predict step is the constant turn-rate and the constant velocity model. To obtain the predicted intensity, the prior is multiplied by the probability of survival p_S, which is a function of the hypothesis state, and a birth intensity γ_k is added. It is given by

$$D_{k|k-1}(\mathbf{x}) = \sum_{j=1}^{J_{k-1}} p_{S,k}(m_{k|k-1}^{(j)}) w_{k-1}^{(j)} \mathcal{N}(\mathbf{x}; m_{k|k-1}^{(j)}, P_{k|k-1}^{(j)}) +$$
$$\gamma_k(\mathbf{x}) \quad (2)$$

The update step utilizes the set of measurements Z_k and yields a posterior intensity

$$D_k(\mathbf{x}) = \sum_{i=1}^{J_{k-1}} [1 - p_{D,k}(m_{k|k-1}^{(i)})] w_{k|k-1}^{(i)} \mathcal{N}(\mathbf{x}; m_{k|k-1}^{(i)}, P_{k|k-1}^{(i)})$$
$$+ \sum_{\mathbf{z} \in Z_k} \sum_{j=1}^{J_{k|k-1}} w_k^{(j)}(\mathbf{z}) \mathcal{N}(\mathbf{x}; m_{k|k}^{(j)}(z), P_{k|k}^{(j)}) \quad (3)$$

where

$$w_k^{(j)}(\mathbf{z}) = \frac{p_{D,k}(m_{k|k-1}^{(j)}) w_{k|k-1}^{(j)} q_k^{(j)}(\mathbf{z})}{\kappa_k(\mathbf{z}) + \sum_{l=1}^{J_{k|k-1}} p_{D,k}(m_{k|k-1}^{(l)}) w_{k|k-1}^{(l)} q_k^{(l)}(\mathbf{z})} \quad (4)$$

$$q_k^{(j)}(\mathbf{z}) = \mathcal{N}(\mathbf{z}; H_k^{(j)} m_{k|k-1}^{(j)}, R_k + H_k^{(j)} P_{k|k-1}^{(j)} [H_k^{(j)}]^\top) \quad (5)$$

$$m_{k|k}^{(j)}(\mathbf{z}) = m_{k|k-1}^{(j)} + K_k^{(j)}(\mathbf{z} - H_k m_{k|k-1}^{(j)}) \quad (6)$$

$$P_{k|k}^{(j)} = [I - K_k^{(j)} H_k] P_{k|k-1}^{(j)} \quad (7)$$

$$K_k^{(j)} = P_{k|k-1}^{(j)} H_k^\top (H_k P_{k|k-1}^{(j)} H_k^\top + R_k)^{-1} \quad (8)$$

The parameters used in the update step are the clutter level $\kappa_k(\mathbf{z})$, the probability of detection $p_{D,k}(m_{k|k-1}^{(i)})$ dependent on the mean of the Gaussian component i (as a target can be occluded or leave the sensor FOV), the observation model H_k and the observation noise covariance R_k. The first sum in (3) represents missed targets, and the second updated targets.

After the update step, the number of Gaussian components increases quadratically. Therefore, all Gaussian components with a very low weight are removed. Moreover, components that are close to each other are merged together and approximated by a single component.

B. Cooperative fusion of PHD intensities

Vehicles located within the communication range of the car $\mathbb{E}$ can share their PHD intensities, i.e., their estimates about targets in their FOV (in this work, this is only the car $\mathbb{L}$). In the next paragraph, we explain how the car $\mathbb{E}$ can fuse received PHD intensities with its own intensity, hence increasing its FOV beyond the one of its sensors, and decreasing uncertainty in the areas of overlapping FOVs.

Before fusion, we need to translate the received intensities (states and covariances of the target hypothesis) to the coordinate frame of the ego vehicle. This is done using the Approximate Transformation method [11], which adds uncertainties of frame poses to targets' covariances. In order to avoid data incest problem, we use a General Covariance Intersection (GCI) algorithm, which offers a conservative way of fusing two Gaussian mixtures [12]. It is shown in [8] that GCI can be approximated to applying Covariance Intersection (CI) pairwise to components from the two intensities. To address the fact that the FOVs may not overlap entirely, we apply CI intersection only to components whose Mahalanobis distance from each other is less than T_F. We take the Gaussian components i and j from intensities D_1 and D_2 respectively; the fused component would have the following mean $m_{ij}^{(12)}$, covariance $P_{ij}^{(12)}$ and weight $w_{ij}^{(12)}$:

$$m_{ij}^{(12)} = P[\mathcal{W}(P_i^{(1)})^{-1} m_j^{(2)} + (1 - \mathcal{W})(P_i^{(1)})^{-1} m_j^{(2)}] \quad (9)$$
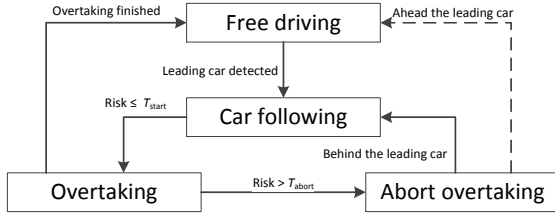
Fig. 2. A FSM defining the behavior of the ego car.

$$P_{ij}^{(12)} = \left[\mathscr{W}(P_i^{(1)})^{-1} + (1 - \mathscr{W})(P_j^{(2)})^{-1} \right]^{-1} \quad (10)$$

$$w_{ij}^{(12)} = \left(w_i^{(1)}\right)^{\mathscr{W}} \left(w_j^{(2)}\right)^{1-\mathscr{W}} \kappa(\mathscr{W}, P_i^{(1)})\kappa(1 - \mathscr{W}, P_j^{(2)})$$

$$\cdot \mathcal{N}\left(m_i^{(1)} - m_j^{(2)}; 0, \frac{P_i^{(1)}}{\mathscr{W}} + \frac{P_j^{(2)}}{1 - \mathscr{W}} \right) \quad (11)$$

where

$$\kappa(\mathscr{W}, P) = \frac{[\det(2\pi P \mathscr{W}^{-1})]^{\frac{1}{2}}}{[\det(2\pi P)]^{\frac{\mathscr{W}}{2}}} \quad (12)$$

and $\mathscr{W}$ is the fusion weight whose value we set to 0.5. We keep track of components that have been fused, and we copy the non-fused components to the result set (e.g., target that are in the FOV of only one vehicle).

IV. Overtaking Decision Algorithm

The behavior of the car $\mathbb{E}$ from Fig. 1 is defined by a Finite State Machine (FSM) shown in Fig. 2. The car $\mathbb{E}$ drives freely on the road at its target speed until the tracking module detects a car driving on the same lane at a lower speed (leading car $\mathbb{L}$). The state of the car $\mathbb{E}$ is then changed to *car following*, where the adaptive cruise controller is engaged. As described in the remainder of this section, a risk for overtaking is continuously computed as a function of the presence and position of the car $\mathbb{O}$, and the decision whether to start (or eventually abort) the overtaking is based on a threshold.

A. State description

1) Free driving: the car $\mathbb{E}$ drives freely on the road at target speed v_0 while keeping centered in its lane.

2) Car following: The car $\mathbb{E}$ follows the car $\mathbb{L}$ (which is driving on the same lane), maintaining a desired distance and time headway.

3) Overtaking: the car $\mathbb{E}$ has determined that it is safe to overtake the car $\mathbb{L}$ (the risk that there is an oncoming car $\mathbb{O}$ on that lane is low enough). It changes lane and performs overtaking at its desired speed v_0.

4) Abort overtaking: the risk that the car $\mathbb{O}$ occupies the overtaking lane has become significant, thus the car $\mathbb{E}$ aborts the overtaking maneuver.

B. Decision algorithm (FSM state transitions)

The initial FSM state is *free driving*. Once the car $\mathbb{L}$ is detected, the state is changed to *car following*.

In the *car following state*, the overtaking risk is constantly computed. The risk is defined between 0 and 1, and it depends on two factors: the probability that there is a car $\mathbb{O}$ on the

passing lane, and the comparison between the distance to overtake and the distance traveled by the car $\mathbb{O}$. The two distances are estimated using the time to overtake and are computed by taking into account the position and the velocity of the ego, leading and oncoming vehicle. If the risk is lower or equal to the defined threshold T_{start} for at least five consecutive simulation steps, the overtaking is initiated.

The risk is constantly computed in the *overtaking* state as well. If the risk exceeds the T_{abort} thresholds for at least two consecutive simulation steps, the state is changed to *abort overtaking*. Otherwise, if the overtaking is successfully finished (the car $\mathbb{E}$ gets ahead of the car $\mathbb{L}$ plus the safety distance d_{safe}), the state is changed back to *free driving*.

In the *abort overtaking* state, the positions of cars $\mathbb{E}$ and $\mathbb{L}$ are compared. If the car $\mathbb{E}$ is ahead of the car $\mathbb{L}$, it accelerates and changes lane back to its driving lane without taking into consideration the safety distance. The state is then changed back to *free driving*. Otherwise, it brakes and pulls behind the car $\mathbb{L}$, and the state is consequently changed to *car following*.

The subsections that follow provide details on how the cars $\mathbb{L}$ and $\mathbb{O}$ are detected among all tracked vehicles and how the overtaking risk is computed.

1) Detection of leading and oncoming vehicle: The tracking and fusion module (see Sec. III) feeds a list of tracked cars to the controller. In order to detect in which lane the tracked car is driving, a temporary coordinate system is placed at the lane in which the car $\mathbb{E}$ is driving (see Fig. 3). The position of the tracked car is expressed in that coordinate system, by taking into account the lateral position of the car $\mathbb{E}$ in its lane (cf. Sec. IV-C). Then, a number of samples are taken at random from the normal distribution centered in the position of the tracked car, with the covariance that is provided by the tracker. This set of samples reflect the probability distribution of the position of the tracked vehicle.

The number of samples falling on one lane over the total number of samples defines the probability that the center of the car is in that lane. However, when considering to overtake, it is important to assess whether *any part* of the car is in the lane. Thus, the probability $P(i, l)$ of a tracked car i to be on lane l is computed by taking into account all the particles that fall in one lane extended by half of the vehicle width on each of the sides. It is important to note that lane occupancy probabilities defined in this way do not sum up to one (i.e., they do not represent a probability mass function).

Finally, in order to determine the direction of travel of the tracked car, we use its velocity, as well as the car $\mathbb{E}$ velocity.

2) Overtaking risk computation: We define the risk to overtake a vehicle given an oncoming vehicle i as a piecewise function

$$r_i = \begin{cases} 0 & \text{if } d_{exp,i} - d_{over} > d_{margin} \\ 1 - \frac{d_{exp,i} - d_{over}}{d_{margin}} & \text{if } d_{exp,i} - d_{over} > 0 \\ 1 & \text{otherwise} \end{cases} \quad (13)$$

where d_{margin} is a margin distance, $d_{exp,i}$ is the expected distance between the ego and the oncoming car i after completed overtaking, and d_{over} is the distance to be traveled by the ego vehicle until the overtaking completion. The
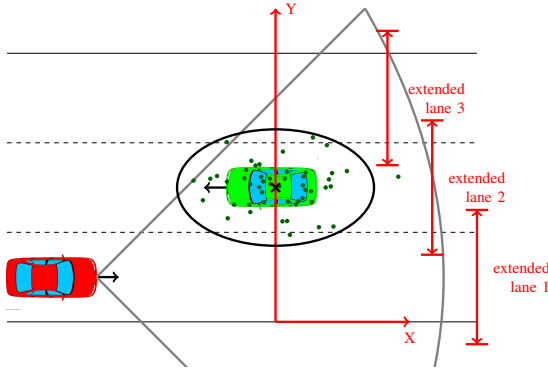
Fig. 3. Lane occupancy probability. The center and covariance of the tracked car is shown in black. For illustration purposes, there are 50 sample points (in green) estimating the position of the car center point. All 50 samples are contained within the *extended* lane 2, 6 samples are falling on *extended* lane 1, and 7 points on *extended* lane 3. Respective lane occupancy probabilities are $\frac{50}{50} = 1.0$, $\frac{6}{50} = 0.12$, and $\frac{7}{50} = 0.14$. The leading car is not shown for clarity reasons.

expected distance is easily computed from the tracked distance to the oncoming vehicle d_i and its tracked velocity $v_{\mathbb{O},i}$, as well as the time that the car $\mathbb{E}$ needs to complete the overtake:

$$d_{exp,i} = d_i - v_{\mathbb{O},i} \cdot t_{over} \qquad (14)$$

The overtaking distance is computed as

$$d_{over} = \frac{1}{2} a_{\mathbb{E}} \cdot t_{over}^2 + v_{\mathbb{E}} \cdot t_{over} \qquad (15)$$

The time to overtake t_{over} can be computed from the quadratic equation for the overtaking length L in the frame of the car $\mathbb{L}$:

$$L = \frac{1}{2} a_{\mathbb{E}} \cdot t_{over}^2 + \Delta v \cdot t_{over} \qquad (16)$$

where $\Delta v = v_{\mathbb{E}} - (v_{\mathbb{L}} + \sigma_{v_{\mathbb{L}}})$ considers the standard deviation of the leaders velocity, and $L = d + l_{\mathbb{L}} + d_{safe}$ is composed of the distance between cars $\mathbb{L}$ and $\mathbb{E}$, length of the car $\mathbb{L}$, and safety distance that is required between the car $\mathbb{E}$ and the $\mathbb{L}$ before the car $\mathbb{E}$ can return back on its lane after overtaking.

Finally, the overtaking risk is defined considering all N_{cars} cars detected by the tracker and probability of them occupying the overtaking lane $l_{\mathbb{E}} + 1$

$$R = \max_{1 \le i \le N_{cars}} (P(i, l_{\mathbb{E}} + 1) \cdot r_i) \qquad (17)$$

The overtaking decision is solely dependent on the risk. We define a threshold T_{start} to start overtaking if the risk is lower, as well as T_{abort} to abort the overtaking if the risk is higher than the threshold.

C. Lateral controller

To enable cars to drive on a particular lane (required in FSM states *free driving* and *car following*), as well as perform lane changes (states *overtaking* and *abort overtaking*), we use pre-defined trajectories. Each lane is composed of waypoints, and a vehicle implements a PI controller to compute the steering angle based on the coordinates of itself and the next waypoint. Changing lane is achieved by simply swapping the used trajectory. While this approach is a simplification of a potentially more realistic lane following and changing method,

it enables us to leverage a perfect positioning information of the car in the lateral controller computations. The lateral controller is the same for each state of the FSM.

D. Longitudinal controllers

Different FSM states use different controllers, as described below.

In the *free driving* and *overtaking* states, the car $\mathbb{E}$ is driven at maximum acceleration a until the desired speed v_0 is reached, and with the constant speed afterwards.

In the *abort overtaking* state, we brake at half of the maximum deceleration if the car $\mathbb{E}$ is positioned behind the car $\mathbb{L}$ (overtaking is aborted and the car $\mathbb{E}$ pulls behind the car $\mathbb{L}$). Otherwise, the car $\mathbb{E}$ accelerates at maximum acceleration and finishes the overtake ahead of the car $\mathbb{L}$ (though at high risk).

The controller employed in the *car following* state is based on the Adaptive Cruise Control (ACC) model. The ACC provides an acceleration for a vehicle following another vehicle in the lane, taking into account the desired velocity and time gap. It was introduced by Kesting et al. in [13] and it represents an improvement to the Intelligent Driver Model (IDM) [14], which is known to brake too hard in some dense traffic situations, for example due to lane changes of other cars. The acceleration of the car given by the ACC model a_{ACC} can be computed as

$$a_{ACC} = \begin{cases} a_{IDM} & \text{if } a_{IDM} \ge a_{CAH} \\ (1-c)a_{IDM} + \\ c[a_{CAH} + b\tanh(\frac{a_{IDM} - a_{CAH}}{b})] & \text{otherwise} \end{cases} \qquad (18)$$

where b and c are the deceleration and coolness parameters as defined in Table I. a_{CAH} is a Constant-Acceleration Heuristic (CAH) which assumes that the velocity of the leading vehicle will not abruptly change and is defined by

$$a_{CAH} = \begin{cases} \frac{v_{\mathbb{E}}^2 \tilde{a}_{\mathbb{L}}}{v_{\mathbb{L}}^2 - 2s\tilde{a}_{\mathbb{L}}} & \text{if } v_{\mathbb{L}}(v_{\mathbb{E}} - v_{\mathbb{L}}) \le -2s\tilde{a}_{\mathbb{L}} \\ \tilde{a}_{\mathbb{L}} - \frac{(v_{\mathbb{E}} - v_{\mathbb{L}})^2 \Theta(v_{\mathbb{E}} - v_{\mathbb{L}})}{2s} & \text{otherwise} \end{cases} \qquad (19)$$

where Θ is the Heaviside step function, s is the distance to the car $\mathbb{L}$, $v_{\mathbb{E}}$ and $v_{\mathbb{L}}$ the velocities of cars $\mathbb{E}$ and $\mathbb{L}$ (cf. Fig. 1), $a_{\mathbb{L}}$ the maximum acceleration of the car $\mathbb{L}$, and $\tilde{a}_{\mathbb{L}} = \min(a_{\mathbb{L}}, a_{\mathbb{E}})$ the effective maximum acceleration. The IDM acceleration function is given by

$$a_{IDM} = a \left[1 - \left(\frac{v_{\mathbb{E}}}{v_0} \right)^\delta - \left(\frac{s^*(v_{\mathbb{E}}, \Delta v)}{s} \right)^2 \right] \qquad (20)$$

$$s^* = s_0 + v_{\mathbb{E}} T + \frac{v_{\mathbb{E}} \Delta v}{2\sqrt{ab}} \qquad (21)$$

where the approaching rate is $\Delta v = v_{\mathbb{E}} - v_{\mathbb{L}}$. Other parameters and their respective values used for the $\mathbb{E}$ car are summarized in Table I.

V. EXPERIMENTAL EVALUATION

A. Experimental setup

A systematic experimental evaluation has been carried out in Webots, a high-fidelity robotic simulator containing

Parameter	Value
Desired speed v_0	90 km/h
Free acceleration exponent δ	100
Desired time gap T	0.1 s
Jam distance s_0	5.0 m
Maximum acceleration a	2.7 m/s^2
Desired deceleration b	6.0 m/s^2
Coolness factor c	0.99

Parameter	Value
Lidar range	140 m
Lidar FOV	110 deg
GNSS noise σ_x, σ_y	0.1 m
GNSS noise σ_v	0.01 m/s
Compass noise $(\sigma_\theta, \sigma_\omega)$	(0.1 deg, 0.1 deg/s)
Communication radius	100 m
Communication delay	80 ms

automotive modules developed in our laboratory.[1] Three Citroën C-ZERO cars have been placed on an infinitely long, straight road with four lanes. The car $\mathbb{E}$ is placed behind the car $\mathbb{L}$, both facing one direction, whereas the car $\mathbb{O}$ faces the opposite direction and finds itself on the passing lane of the car $\mathbb{E}$. The longitudinal positions of the three cars are random in each experimental run, but their configuration remains as explained.[2] Our algorithm could easily be adapted for curved roads by computing the distance in the road (curvilinear) coordinate system.

The cars $\mathbb{E}$ and $\mathbb{L}$ are traveling with target speeds v_0 of 90 km/h and 50 km/h, respectively. They are equipped with a forward facing Ibeo LUX lidar sensor, generic GNSS and compass sensors achieving RTK performance, and a communication transceiver. Communication is achieved by simple messages implemented in the simulator environment using constant delay and message loss. The characteristics of integrated devices are listed in Table II. The car $\mathbb{O}$ travels at target speed of 30 km/h and is not equipped neither with sensing nor communication devices. Using higher traveling speeds would require using sensors with larger range.

The vehicle model implemented in Webots uses the throttle, brake, and steering angle as inputs. The input steering angle (provided by the lateral controller) is transferred to the wheels using a Webots built-in PID controller. We use the Citroën C-ZERO engine model[3] to map the required acceleration (provided by the longitudinal controller) to the throttle input of the Webots *car library*, and we assume a linear brake response in the case of deceleration. The simulation step is 80 ms, equivalent to the sampling frequency of the lidar.

---

[1]For more details see http://disal.epfl.ch/RO2IVSim.

[2]A video showcasing different experiments can be viewed at http://disal.epfl.ch/NetworkedIV

[3]Provided by PSA Groupe.

B. Tracking parameters

In the C-GM-PHD filter, we empirically determine the sensor standard deviation to be $\sigma_z = [\sigma_x, \sigma_y, \sigma_\theta]^\top = [1\,\mathrm{m}, 1\,\mathrm{m}, 45\,\mathrm{deg}]^\top$ (this includes the point cloud pre-processing noise). To make our filter conservative, the ego state estimation noise needed for the Approximate Transformation is intentionally not set exactly to the values used in the simulator (cf. Table II). Instead, it equals to $[\sigma_x, \sigma_y, \sigma_\theta, \sigma_v, \sigma_\omega]^\top = [1\,\mathrm{m}, 1\,\mathrm{m}, 0.5\,\mathrm{deg}, 1\,\mathrm{m/s}, 0.5\,\mathrm{deg/s}]^\top$. The clutter model is assumed to be Poisson with mean of 10 clutter measurements per sensor surveillance area. The probability of detection and survival are respectively set to 0.98 and 0.99. The occlusion model, as well as the values of other parameters intrinsic to the C-GM-PHD filter are the same as in our previous work [2].

C. Experimental results

The experiments are conducted using six different parameter settings. While keeping the value of T_start at some small, positive value (we chose 0.01), we vary $T_\mathrm{abort} \in \{0.2, 0.5, 0.8\}$. For each value of T_abort, we launch the experiment with and without cooperative fusion. Without fusion, the car $\mathbb{E}$ tracks targets using only its lidar, thus having shorter range and more occluded FOV. For each of the six settings we perform 500 simulation runs with different random initial positions to obtain statistically significant results.

An overtaking attempt is defined as an event in which the car $\mathbb{E}$ starts changing the lane. It can be finished by a successful overtake, an abort or a crash.

Fig. 4a shows the duration of overtaking maneuver, measured from the moment the first overtaking attempt is made (excluding experiments that resulted in crashes). Long durations represent overtaking maneuvers whose first attempts have been aborted, as in these cases the car $\mathbb{E}$ had to wait for the risk to diminish before starting and successfully completing the second overtaking attempt. It can be seen that the duration decreases with increased risk threshold T_abort. This is due to overtakes that are less likely to be aborted, for the price of accepting more risk. We can also see that experiments which use fusion outperform the ones without fusion algorithm: since, when using fusion, the car $\mathbb{E}$ suffers less from occlusions and can see further away, it aborts the overtaking attempt less frequently and therefore saves time.

The percentage of aborted overtakes is given in Table III. Aborts are split in two categories, the ones in which the car $\mathbb{E}$ brakes and pulls behind the car $\mathbb{L}$, and the ones in which the car $\mathbb{E}$ keeps accelerating and successfully finishes the risky maneuver. There are significantly less aborts of the second type in the case of fusion than in the case of no fusion. Their number also decreases with the increasing T_abort, as the car $\mathbb{E}$ allows for higher risk before aborting. Due to larger sensor range in the fusion case, there are slightly more aborts of the first type than in cases without fusion. These facts lead us to the conclusion that cooperative fusion might contribute to road safety, as it reduces the number of occasions in which the overtaking car enters risky situations. Fig. 4b supports this conclusion, by showing the actual risk at which the overtaking
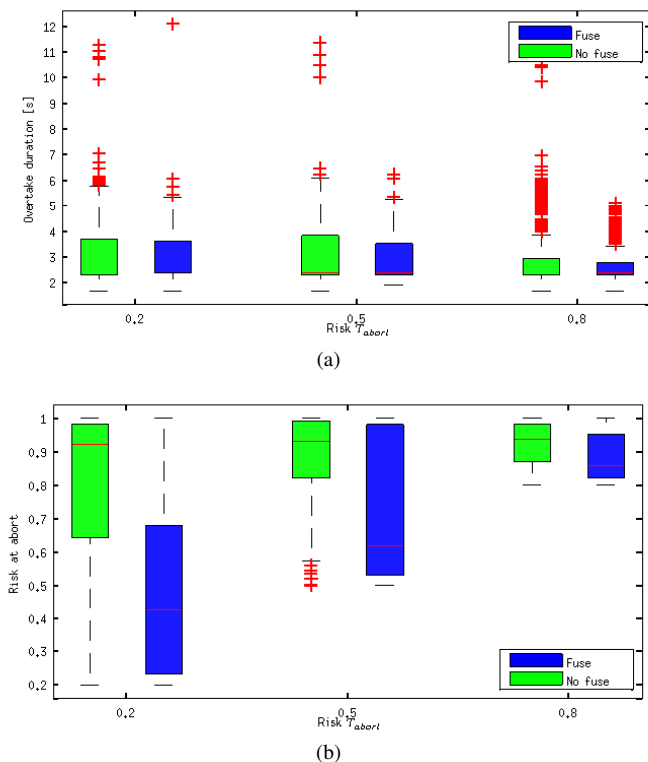
Fig. 4. Boxplots showing for different values of T_{abort} (a) overtake duration from the moment of the first overtaking attempt, (b) overtaking risk at abort (note: the risk is always higher or equal than the abort threshold T_{abort}). Each box aggregates 500 runs minus the number of crashes and represents the upper and lower quartiles, the red line in the box marks the median, the bars extend to the most extreme data points not considered outliers, and the red crosses show outliers.

TABLE III

PERCENTAGE OF ABORTED AND CRASHED OVERTAKES.

| | Risk | | | | | |
| | 0.2 | | 0.5 | | 0.8 | |
	No fuse	Fuse	No fuse	Fuse	No fuse	Fuse
Abort behind $\mathbb{L}$ (%)	2.47	3.58	2.24	4.25	2.93	3.18
Abort in front of $\mathbb{L}$ (%)	16.29	0.42	9.39	0.21	7.53	0.21
Crash (%)	1.62	0.00	1.01	0.00	2.65	0.81

maneuver was aborted. It is always significantly lower in the case of fusion, independent from the chosen value of T_{abort}.

Overtakes resulting in a crash between the car $\mathbb{E}$ and the car $\mathbb{O}$ are listed in Table III as well. Data show that cooperative fusion successfully reduces the number of crashes with the oncoming car, which is indeed its intended purpose. As expected, the highest number of crashes are recorded for $T_{\text{abort}} = 0.8$.

VI. CONCLUSION

We presented a novel overtaking decision algorithm for intelligent vehicles which relies on the C-GM-PHD filter. By using information provided by the filter, the algorithm assesses the risk of overtaking (i.e., the risk of an oncoming car being on the passing lane within the collision distance

from the ego car). We compared the cases in which the leading car cooperates with the ego car by sharing its track estimates, and in which it does not. The overtaking algorithm showed good performance and the added value of cooperative fusion was evident from the experimental data. The number of risky maneuvers was significantly reduced when the ego car benefited from the higher sensor range, less occlusions and higher tracking confidence. Due to lower risk, the number of crashes between the ego and the oncoming car was also lower. When the cooperative fusion algorithm was used, a smoother and safer vehicle operation was noticeable.

As future work, one could consider more complex evaluation scenarios, such as those characterized by additional traffic traveling on multiple lanes in different directions, or having different traffic participants in addition to cars (e.g., trucks, bikes). Moreover, the tracking algorithm would need to be extended to address the object detection problem in more complex scenarios.

REFERENCES

[1] "Traffic Safety Facts 2013: A compilation of motor vehicle crash data from the fatality analysis reporting system and the general estimates system," NHTSA, DOT HS 812 139, 2013.

[2] M. Vasic and A. Martinoli, "A collaborative sensor fusion algorithm for multi-object tracking using a Gaussian mixture Probability Hypothesis Density filter," in *IEEE Intelligent Transportation Systems Conference*, 2015, pp. 491–498.

[3] R. Toledo-Moreo, J. Santa, and M. A. Zamora-Izquierdo, "A cooperative overtaking assistance system," in *Iros 2009 3rd Workshop: Planning, Perception and Navigation for Intelligent Vehicles*, 2009.

[4] A. Vinel, E. Belyaev, K. Egiazarian, and Y. Koucheryavy, "An overtaking assistance system based on joint beaconing and real-time video transmission," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 5, pp. 2319–2329, Jun. 2012.

[5] W. Birk, E. Osipov, and J. Eliasson, "iRoad - cooperative road infrastructure systems for driver support," in *16th World Congress and Exhibition on Intelligent Transport Systems*, 2009. ISBN 978-1-61738-589-6

[6] B.-N. Vo and W.-K. Ma, "The Gaussian mixture probability hypothesis density filter," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, Nov. 2006.

[7] B. Ristic, D. Clark, and B.-N. Vo, "Improved SMC implementation of the PHD filter," in *13th Conference on Information Fusion*, Jul. 2010. doi: 10.1109/ICIF.2010.5711922

[8] G. Battistelli, L. Chisci, C. Fantacci, A. Farina, and A. Graziano, "Consensus CPHD filter for distributed multitarget tracking," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 3, pp. 508–520, Jun. 2013.

[9] M. Uney, D. Clark, and S. Julier, "Distributed fusion of PHD filters via exponential mixture densities," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 3, pp. 521–531, Jun. 2013.

[10] M. Ester, H.-P. Kriegel, J. S, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *International Conference on Knowledge Discovery and Data Mining*, Portland, Oregon, USA, 1996, pp. 226–231.

[11] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *The International Journal of Robotics Research*, vol. 5, no. 4, pp. 56–68, Dec. 1986.

[12] R. P. S. Mahler, "Optimal/robust distributed data fusion: a unified approach," in *Signal Processing, Sensor Fusion, and Target Recognition IX*, vol. 4052, 2000, pp. 128–138.

[13] A. Kesting, M. Treiber, and D. Helbing, "Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity," *Philosophical Transactions of the Royal Society A*, vol. 368, pp. 4585–4605, 2010.

[14] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Physical Review E*, vol. 62, no. 2, pp. 1805–1824, 2000.