Honda Research Institute EU

Honda Research Institute Europe GmbH
https://www.honda-ri.de/

Tracking Moving Vehicles Using an Advanced Grid-based Bayesian Filter Approach

Andreas Alin, Martin Butz, Jannik Fritsch

2011

Preprint:

This is an accepted article published in IEEE Intelligent Vehicles Symposium (IV). The final authenticated version is available online at: https://doi.org/[DOI not available]

2011 IEEE Intelligent Vehicles Symposium (IV)
Baden-Baden, Germany, June 5-9, 2011

Tracking Moving Vehicles Using an Advanced Grid-based Bayesian Filter Approach

Andreas Alin, Martin V. Butz and Jannik Fritsch

Abstract— **Neuroscientific research suggests that the human brain encodes spatial information in a Bayesian-optimal way by means of distributed, neural population codes. In this paper we apply this concept to Advanced Driver Assistance Systems, introducing a grid-based population code for tracking and predicting the behavior of individual vehicles. The representation encodes a spatially distributed hidden Markov model of current and future vehicle locations and velocities. Predictive information and additional sensory information are integrated over time by means of Bayesian filters. Performance of the system is compared with a Kalman Filter in an overtaking maneuver in a simulated environment. It is shown that the grid-based approach excels Kalman-Filtering performance in several situations, where the Gaussian distribution and linear system assumptions of the Kalman filter are strongly violated. Moreover, the grid-based approach allows the flexible incorporation of additional behavioral assumptions. When the approach assumes that the tracked vehicle will stay in its lane, the probability distribution can be even more favorably focused and unexpected lane changes can be detected.**

I. INTRODUCTION

To be able to warn about and avoid potentially dangerous events, Advanced Driver Assistance Systems (ADAS) should be able to both identify other objects in their surrounding and predict their location over the next few seconds. To improve and differentiate the prediction process, object-specific movement and environmental constraints should be incorporated. Humans can accomplish such tasks rather well, even though their knowledge about their surrounding is inevitably filled with uncertainties. Thus, to be able to make adequate decisions, our brain needs a way to handle uncertainties effectively. Computational neuroscience and related research areas suggest that the human brain integrates information in a Bayesian optimal way, maintaining a probabilistic model of the environment encoded in neural population codes [1]. In this paper, we investigate the advantages of population-encoded representations, building a distributed representation of anticipated object presence probabilities around a vehicle.

For realizing an ADAS with the mentioned capabilities, four major challenges have to be addressed: 1) Objects need to be detected and identified. 2) Object movements and behaviors need to be tracked and predicted. 3) A strategy for the own vehicle has to be found. And 4) the strategy has to be accomplished by either informing the driver or by automatically avoiding dangerous situations, such as collisions. This paper deals with the second challenge.

In real-world situations, detections are prone to error and therefore detection data should be filtered by incorporating predictive sources of information. To combine predictive and sensory sources of information in an optimal way, Bayesian Filters can be used. In the automotive domain, previously, Bayesian Occupancy Filters (BOFs) were proposed [2], [3], [4], which represent an occupancy grid that encodes probabilities about grid location occupancies. In contrast to BOFs, the Bayesian Histogram filter (BHF) models probabilities of the presence of individual surrounding vehicles at each node in a discrete grid, which surrounds the own vehicle. Our approach extends the BHF. Probabilities are propagated across grid nodes by means of a movement model, which reflects the dynamics of the tracked vehicle. The resulting predicted probability distribution for the next time step is fused with incoming sensory information. Due to the utilized population-encoded representation, neither the movement model nor the sensory model need to be Gaussian. Moreover, additional environmental influences, such as road boundaries, can be taken into account easily. Generally this can also be used to predict possible future positions of multiple tracked vehicles in a probabilistic way. In the long run, such a representation can therefore be used to detect and avoid dangerous areas, where a collision with other vehicles, pedestrians, or objects is highly probable. Other approaches [5] also produce a reachable set of vehicles, but we are using a less deterministic environmental model as input.

We now first relate our work to previous approaches, giving an overview over the general underlying Bayesian mechanisms and then taking a closer look at the related BOF system. Next, we detail our grid-based approach. For evaluation purposes, we compare our system with the results of a common Kalman-Bucy filter in a tracking task, where we exemplarily focus on an overtaking scenario generated by the car simulator TORCS. Finally, we summarize the achievements and conclude the paper.

II. Related Work

To track an object, different implementations of Bayesian filters exist. We first describe the basic mathematical background and then introduce some implementations of the Bayesian filter related to our approach.

In general, we may denote the state of a system by variable x within a state space X. Variable u may denote the

A. Alin is with the Department of Psychology III, University of Wuerzburg, Wuerzburg, Germany andreas.alin@uni-wuerzburg.de
M. V. Butz is with the Department of Psychology III, University of Wuerzburg, Wuerzburg, Germany butz@psychologie.uni-wuerzburg.de
J. Fritsch is with the Honda Research Institute, Offenbach am Main, Germany jannik.fritsch@honda-ri.de

(control) system input and z the (sensory) system measurement. Iterating over time, we may denote the state, input, or measurement of a system at a certain point in time by the subscript x_t, u_t, or z_t, respectively. A Bayesian filter iteratively applies a prediction and a sensory filtering step, converting the probability distribution of the previous state $p(x_{t-1})$ into the posterior distribution $p(x_t|x_{t-1}, u_t, z_t)$. First, a given control command leads to a state prediction:

$$P(x_t|z_{1:t-1}, u_{1:t}) = \int p(x_t|u_t, x_{t-1}) \cdot p(x_{t-1}|z_{1:t-1}, u_{1:t-1})\, dx_{t-1}, \tag{1}$$

in which the conditional prediction function $p(x_t|u_t, x_{t-1})$ is multiplied with the posterior distribution at iteration $t-1$. This equation may thus be understood as a probability transfer dependent on control command u_t, thus modeling the predicted system dynamics. Next, the filtering step is applied combining the incoming measurement with the predicted distribution from (1):

$$p(x_t|z_{1:t}, u_{1:t}) \propto p(z_t|x_t) \cdot p(x_t|z_{1:t-1}, u_{1:t}), \tag{2}$$

where $p(z_t|x_t)$ denotes the measurement model, which delivers the probability over all possible measurements z, observing a system with true state x by a (noisy) sensor. The resulting posterior distribution is finally normalized to 1 over all states in X.

There are different implementations of the Bayesian Filter. For non-discrete state variables, all implementable algorithms are approximate solutions of the above filtering equations. The Kalman Filter is an exception to this, if it is used on linear problems with Gaussian noise. In this case, the integral in (1) can be solved by a closed form solution. Often the Kalman filter is also used for non-linear and non-Gaussian systems, but will – like the Extended Kalman filter – produce only approximations of the real system behavior. The more the Gaussian distribution assumption is violated in the Kalman filter application, the more error prone are the results produced by the filter. Due to this problem, *particle filters* are widely used to deal with non-linear systems. Particle filters, however, have the drawback that the choice of the number of maintained particles may strongly influence the resulting system behavior [6]. Moreover, the fusion of several particle distributions can lead to disruptions. As an alternative, we thus investigate the performance of *grid-based filtering* approaches.

A. Bayesian Histogam Filter (BHF)

Bayesian Histogram Filters (BHFs) are approximating the integral in (1) with a sum over a finite number of intervals of the state space (3). These intervals are disjoint subsets of the state space. A straightforward decomposition of an m-dimensional continuous state space is an m-dimensional grid, where each subset is a grid cell [6]. A finer granularity of the cells improves accuracy, but at the same time it increases the computational effort. BHFs focus the computational effort by updating more important cells with higher probability.

Let $\hat{x}$ denote a certain point within the grid cell at which a BHF probes the probability density function. A piecewise constant histogram may be created by dividing through the cell size $|x_{k,t}|$: $p(x_t) = \frac{p_{k,t}}{|x_{k,t}|}$. The prediction and filtering processes may then be written as follows:

```
For all grid cells k do:
```

$$p(\hat{x}_{k,t}|z_{1:t-1}, u_{1:t}) = \sum p(\hat{x}_{k,t}|u_t, \hat{x}_{k,t-1}) \cdot p(\hat{x}_{k,t-1}|z_{1:t-1}, u_{1:t-1}) \tag{3}$$

$$p(\hat{x}_{k,t}|z_{1:t}, u_{1:t}) = \eta p(z_t|\hat{x}_t) \cdot p(\hat{x}_{k,t}|z_{1:t-1}, u_{1:t}) \tag{4}$$

These kinds of filters are rarely used in the automotive domain, but are common for personal localization systems [7]. Our grid-based system is essentially a modified version of a BHF.

B. Binary Bayesian Filter (BBF)

Binary Bayesian Filters (BBFs) have a state space with binary state variables and usually very high dimensional state vectors. If each dimension of the state space denotes a position in a map and the two states encode occupied and free locations, we call that filter an occupancy grid map. These representations are widely used in the robotics domain for solving the SLAM (simultaneous localization and mapping) problem (cf. e.g. [8]). In [9] a hybrid approach for object tracking in automobile scenes, which assigns to each object a number of cells in an occupancy grid and utilizes a Kalman filter for object-based tracking, is introduced.

C. The Bayesian Occupancy Filter (BOF)

On the basis of BBFs, the BOF [2], [4] was developed. Each cell's occupancy value is calculated by an individual Bayesian network. These networks are connected by the prediction model of the BOF, which assumes a constant object velocity with a Gaussian error distribution, approximating acceleration effects. Since the exact solution of a BOF cannot be solved analytically and cannot be computed in real-time, the solution is usually approximated in the implementation by using only the most informative cells. In contrast to early versions of the BOF, in the newer versions each cell has one probability value and one velocity value [3].

BOFs cannot be used directly for object tracking since no objects but only occupancy probabilities are known. This has the advantage that measurements do not need to be assigned to objects, if object are to be tracked in the application, but the assignment of the occupied cells to objects has to be done later. This was done in [3] by using a Kalman filter as an object-state-based Bayesian filter for each object together with a graph based clustering scheme. The clustering scheme is comparing two nearby grid cells by the Euclidean distance between the position, occupancy value, and velocity. Each track is associated with the objects in a probabilistic way by the common Joint Probabilistic Data Association (JPDA) filter. While the BOF creates good results for detecting occupied areas, it has problems in the object tracking task with nearby detections in liaison with the JPDA algorithm.

The BOFUM (Bayesian Occupancy Grid Filter for Dynamic Environments Using Prior Map knowledge) [10] and

BOFUG (Bayesian Occupancy Filtering using Groups) [11], are improved versions of the BOF. While the BOFUM introduces the idea that the movement of traffic participants is highly influenced by the map knowledge, the BOFUG detects the type of traffic participant, e.g. pedestrians versus vehicles, by comparing the movement history with respective, orthogonal movement models: The movement model for pedestrians is to cross a street in a right angle, while the movement model for a vehicle is to drive along the street.

The difference of the BOF approaches to our approach is the way how probabilities are handled. We use a BHF as the underlying theory, which means that the probability in a grid node represents the probability $p(x)$ that an observed object has a certain state x. Therefore, the sum over all possible states (=grid nodes) has to be 1. The BOF, on the other hand, is based on Binary Bayesian filters. The probability in the grid nodes are representing the probability that the area around that grid node is occupied by any kind of object. The sum of the probabilities of all grid nodes can consequently vary between 0 and the number of grid nodes n. While the prediction step in BOFs can be handled likewise – differing only in the way normalization takes place – a main difference can be found in the filtering step. In the sensor model of BOFs, hidden regions are explicitly modelled as unknown areas, tending towards a probability of 0.5 in a grid node without further measurements. Our approach, on the other hand, provides probability estimates about the position of a certain vehicle or object. Hidden areas would generally remain free unless evidence becomes available that an object may have moved into that area – such as a previously visible object that disappears behind an occlusion.

III. GRID MODEL

We aim to modify the BHF, to receive an algorithm that is fast enough for real-time computation but still yields an accurate prediction density and filtering model for automobile scenes. Therefor we use the kinematic model of vehicles to improve the prediction step. As a further improvement, we introduce a way to reduce the discretization error in the prediction step of a tracking application. Finally, we improve the prediction further by absorbing probability flow that crosses lane markings - thus assuming that the vehicle will stay in its lane.

A. System Overview

Our model is specified by its internal state X, which specifies the probability density function and estimated local velocity of an object or vehicle in a distributed grid of N nodes n_i. In the current work, the grid nodes are equidistantly distributed from each other in a rectangular grid, but another structure is also possible. Each node n_i has certain properties at a certain time t: its location in space $(l_x, l_y)_{i,t} = \mathbf{l}_{i,t}$, its object presence probability $p_{i,t}$, as well as the velocity estimate of that object given its presence $(v_x, v_y)_{i,t} = \mathbf{v}_{i,t}$. The system state, encoded by the properties of the N nodes, essentially specifies the internal Markov state of our model, which is used for generating predictions and filtering additional sensory information. Thus, our approach relies on the Markov assumption, that is, $P(X_{t+1}|X_{1:t}, u_{1:t}, z_{1:t}) = P(X_{t+1}|X_t)$.

Naturally, grid-based filters can only cover a limited spatial area. Because of that, border effects can occur. To avoid such effects, probabilities should be allowed to leave the grid at its borders. We use three rows of absorbing border nodes at the edges of the grid in our experiments, which showed to yield a good trade off between accuracy and performance.

During tracking, the prediction step is realized by a movement model function f_{mov}, which models the distribution of potential driving behaviors and dynamics. The resulting probability received after the prediction step is then updated with the sensor information by considering the sensor model, which encodes the knowledge about the error distribution of a certain kind of sensor. In particular, the sensor model encodes the probability that an object is located at position x, given a sensory reading of position y. Prediction and sensor-based filtering is now detailed.

B. Prediction

In the prediction step, possible object movements between time-step t and $t+1$ are calculated. To do so, we propagate the probabilities of object presences to the neighboring positions that lie in range of the local velocity estimate. For convenience, let us define a probability flow from node j to node i by:

$$flow_{i,j,t} := p_{j,t} \cdot f_{mov}(\mathbf{v}_{j,t}, \mathbf{l}_{i,t}, \mathbf{l}_{j,t}), \tag{5}$$

where f_{mov} denotes the movement model function, which is specified below. To limit the computational burden of this operation, flow values are only calculated for movement model probabilities above a threshold Θ_p – otherwise the flow is set to zero. Now, we can predict the new probability value of object presence for a node i in the next time step $t+1$:

$$P(i_{t+1}|X_t) = \sum_{j=1}^{N} flow_{i,j,t} \tag{6}$$

Note that no flow is determined from border nodes, which assures that border nodes absorb probabilities. These two equations thus accomplish the prediction step of general Bayesian filters specified in (1) and of BHFs in particular (3). However, also the velocities are propagated, as specified now. In theory, an exact BHF needs to model all possible velocities in each grid node. To make this operation computationally feasible, however, we collapse the velocity dimension to one average velocity value $\mathbf{v}_{i,t}$ in each grid node. The velocities are propagated through the grid similar to the probability values. The resulting velocity in a node, however, is inferred by the flow-averaged positional differences between the node's location and the respective flow origins. We average the direction and distance components of the velocity separately to preserve both pieces of information independently:

$$\theta(\hat{v}_i)_{t+1} = \angle\left(\frac{\sum_{j=1}^{N}(\mathbf{l}_{i,t}-\mathbf{l}_{j,t})\cdot flow_{i,j,t}}{\sum_{j=1}^{N} flow_{i,j,t}}, r\right), \tag{7}$$

$$|\hat{v}_i|_{t+1} = \frac{\sum_{j=1}^{N}|\mathbf{l}_{i,t}-\mathbf{l}_{j,t}|\cdot flow_{i,j,t}}{\Delta t \sum_{j=1}^{N} flow_{i,j,t}}, \tag{8}$$

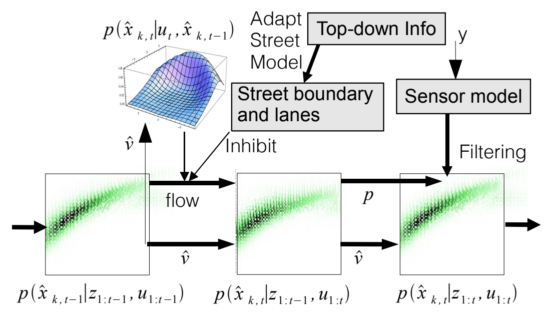

$p(\hat{x}_{k,t}|u_t, \hat{x}_{k,t-1})$
Adapt Street Model
Top-down Info
y
Street boundary and lanes
Sensor model
$\hat{v}$
Inhibit
Filtering
flow
p
$\hat{v}$
$\hat{v}$
$p(\hat{x}_{k,t-1}|z_{1:t-1}, u_{1:t-1})$
$p(\hat{x}_{k,t}|z_{1:t-1}, u_{1:t})$
$p(\hat{x}_{k,t}|z_{1:t}, u_{1:t})$

Fig. 1. A schematic view of a complete iteration of the algorithm.

whereby r is a fixed reference vector and $\angle(\mathbf{x}, \mathbf{y}) = \arccos \frac{\langle x,y \rangle}{||x|| \cdot ||y||}$. The prediction step is finalized by the trivial transformation of orientation $\theta(\hat{v}_i)_{t+1}$ and absolute velocity $|\hat{v}_i|_{t+1}$ to the Euclidean velocity $\mathbf{v}_{i,t+1}$. An overview over the prediction step is shown in Fig. 1. We now proceed with detailing the movement model applied in the prediction step, its modification when additional environmental constrains are included, and finally how the prediction step can be further optimized.

1) Movement Model: To receive a good prediction, the update function has to consider the kinematic constraints of the modeled object. This is barely possible with a Kalman Filter, since the update function has to be implemented as Gaussian noise in the system state. In our grid-based approach we use the bicycle model, also known as one-track model, as the underlying model, which calculates the reachable space of a car-like robot. We modify this model to receive a probability distribution over the reachable terrain by applying independent Gaussian noise to the velocity $||\mathbf{v}||$ and the direction $\omega(\mathbf{v})$ of a vehicle, which results in a probabilistic movement model that is shaped like a crescent moon (cf. Fig. 1). Note that we actually process the velocities in the grid relative to the observing vehicle. To process the movement model properly, these relative velocities are converted into absolute velocities by considering the ego motion of the observing vehicle.

Let us denote $\phi(x; \mu, \sigma^2)$ as the probability value derived from the Gaussian distribution with expectation value μ and standard deviation σ for value x. Let us further denote σ^2_ω as a parameter that characterizes the steering capability and $\sigma^2_{|v|}$ as another parameter that characterizes the possible velocity changes. Then, the movement model can be defined as follows:

$$\begin{aligned} & f_{mov}(\mathbf{v}, \mathbf{l}_i, \mathbf{l}_j) := \\ & \phi(\omega(\mathbf{l}_i - \mathbf{l}_j); \omega(\mathbf{v}), \sigma^2_\omega) \cdot \phi(||\mathbf{l}_i - \mathbf{l}_j||; ||\mathbf{v}||, \sigma^2_{|v|}) \\ & + \phi(\omega(\mathbf{l}_i - \mathbf{l}_j) + \pi; \omega(\mathbf{v}), \sigma^2_\omega) \cdot \phi(-||\mathbf{l}_i - \mathbf{l}_j||; ||\mathbf{v}||, \sigma^2_{|v|}) \end{aligned} \quad (9)$$

which multiplies the uncertainty due to possible speed changes with those due to possible directional changes. The second summand additionally models the probability for a speed reversal, which becomes particularly relevant given very low velocities $\mathbf{v}$.

2) Model Environment Constraints: The grid representation allows the incorporation of expected driving behavior into the prediction process. For example, a stop sign may be incorporated by reducing the predicted velocity while approaching it. In this work, we consider lane knowledge assuming that a tracked vehicle will stay in its lane.

To achieve this, we change the prediction model by introducing an absorption ratio a in (5) by reducing the velocity in a flow from node j to node i by the factor $(1 - a \cdot B(j, i))$, where $B(j, i)$ is a characteristic function, which returns 1 if there is a lane border between nodes i and j, and 0 otherwise. The higher the absorption ratio $a \in [0, 1]$ is set, the stronger a lane absorbs the probability flow. Note that this is an approximate solution, which shows good performance as evaluated below.

3) Discretization Error Reduction: Advantage and curse in BHFs are the discretization of space, where a coarser discretization has a lower computational demand but also yields higher errors. The standard point mass algorithm [12] can cause a rather high discretization error in the velocity dimension, because it simply computes the velocity by the distance between the origin node and target node. Since the velocity in our grid-approach is discretized as well and the update function depends on the velocity, an increasingly higher error can be expected to be propagated in successive predictions without filtering.

Although the filtering step counteracts this error, we used an additional improvement of the probability mass and velocity estimations in each prediction step, linearly interpolating probabilities and velocity values over the region covered by a grid node. Each grid node essentially covers a rectangular region of influence bordered by four corner nodes. Instead of determining the probability flow with respect to the location of the grid node itself, we interpolate over the four corner nodes in (5) and substitute $\mathbf{l}_{i,t}$ in (7) and (8) with

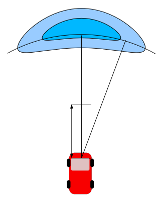
$$\mathbf{l}^{j*}_{i,t} = \frac{\sum_k \mathbf{l}^k_{i,t} f_{mov}(\mathbf{v}_{j,t}, \mathbf{l}^k_{i,t}, \mathbf{l}_{j,t})}{\sum_k f_{mov}(\mathbf{v}_{j,t}, \mathbf{l}^k_{i,t}, \mathbf{l}_{j,t})}, \quad (10)$$

where $\mathbf{l}^k_{i,t}$ denotes one of the four corner locations of node i, and $k \in \{0, 1, 2, 3\}$. Note that the increase in computational complexity is constant, because the values at the integration corners can be re-used by the four neighboring grid nodes.

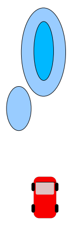
C. Filtering

The grid approach has a rather significant advantage, when it comes to filtering. Generally, any type of sensor-based noise model can be considered and distributed over the grid – as long as the grid is fine-enough.

We model a radar sensor in our simulation (Fig. 2(a)) by applying Gaussian noise to the polar coordinate with the ego-vehicle as origin and noise in the distance. The noise in angle is usually higher than the noise in distance. Furthermore, we model a camera-based sensor by adding higher noise to the distance axis and lower noise to direction-perpendicular axis (Fig. 2(b)). The filtering step in the grid then is accomplished



(a) The sensor model of a radar system applies Gaussian noise to the polar coordinate and the distance.



(b) The sensor model of a camera with an implied false-positive measurement.

Fig. 2. Tested Sensor Models

according to (4) and finally the resulting distribution over the grid is normalized to one.

IV. Evaluation

We evaluate our grid approach by modeling an overtaking situation in a simulated automotive environment. In particular, we modified the open-source car racing simulator TORCS [13] to simulate this scenario. We now first detail the employed simulation environment, test scenarios, the filter setups, and then present the achieved results.

A. TORCs Simulator

To receive test data of moving vehicles the freely available open-source car racing simulator TORCs [13] was used. We enhanced the simulator by determining simulated, local, ego-centered state information. To receive a realistic sensor feedback, noise was added to the real state information. TORCS provides a rather realistic physics engine and a modular driver design. This means that users can define their own drivers using a defined interface.

We adapted the setup that was utilized in the recent Simulated Car Racing Championships [14]. To the existing sensor system, we added sensors to detect “virtual” lanes and obstacles, which block the lane fully or partially. We use the term “virtual”, because the lanes and obstacles are not present in the simulation itself, but added to the sensory data depending on the position of the own car (car A). By means of the virtual sensors we essentially simulate a state-of-the-art lane detection algorithm, producing a polygon-shaped driving space.

B. Scenario and Setup

In our scenario, car A uses the grid to track the overtaking behavior of car B. Car A accelerates to a constant speed of 80km/h on the right lane of a simulated road. The overtaking car B starts next to A in the left lane and overtakes car A while accelerating to 100km/h. It stays in that lane for 8 timesteps = 4.0 seconds, which corresponds to 100 meters. Then, it switches the lane ahead of car A for 4 timesteps = 2.0 seconds, while reducing speed to 80km/h, and finally continues in the right lane ahead of car A for another further 8 timesteps until it enters a right curve. Fig. 3 shows a screenshot of the scenario during the lane switch. The overtaking trajectory of car B was intentionally controlled in a rather abrupt fashion to evaluate to what extent the tested filters can deal with different directional changes.

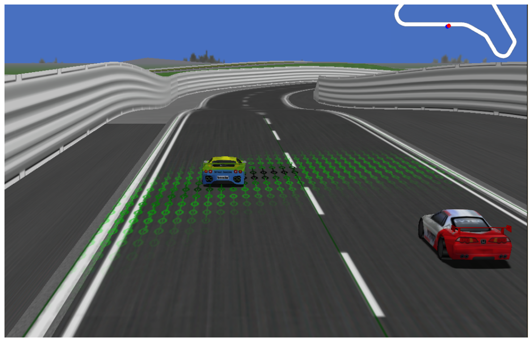


Fig. 3. A screenshot from the test scenario. The yellow car B is overtaking the red vehicle A, heading for the curve. The active grid nodes are drawn in perspective on the street.

We ran our evaluation with the following parameters. Threshold Θ_p = 0.01. Distance between nodes: 0.5 m. Node count in x direction: 60, Node count in y direction: 80. The sensors were modeled to mimic actual ones. For the radar sensor we assumed 0.218 rad as angular standard deviation. Accuracy in distance is modeled with 2% of the distance. The Kalman Filter receives an approximation of that radar sensor model, since it is not possible to use polar coordinates in the sensor model and Euclidean coordinates for the internal state without transformations. However we use an adaptive sensor model, rotating the Gaussian shape appropriately to fit the radar sensor in the following way:

$$\begin{aligned} r_{11} = & \quad \sigma_z^2\, cos(\alpha)^2 + \sigma_w^2\, sin(\alpha)^2 \\ r_{12} = r_{21} = & \quad (\sigma_z^2 - \sigma_w^2)\, cos(\alpha)\, sin(\alpha) \\ r_{22} = & \quad \sigma_z^2\, sin(\alpha)^2 + \sigma_w^2\, cos(\alpha)^2, \end{aligned}$$

where $r_{11}, r_{12} = r_{21}$ and r_{22} are the new elements of the measurement noise covariance matrix. α is the angle between the measurement and the ego-vehicle movement direction. σ_w is the noise in the distance and σ_z is the tangens of the noise in direction multiplied with the distance of the measurement. Note that we ignore the resulting small error in the mean of the distribution.

For the stereo camera model, we used a pixel width of $1.10 \cdot 10^{-5}$ m, a base line of 0.3 m and a focal length of 0.012 m to calculate the resolution in the distance. Angular standard deviation of the camera sensor is modeled with 0.0873 rad. We set the absorbtion ratio a to $\frac{19}{20}$ and the movement model parameters $\sigma_\omega = 0.16$ and $\sigma_{|v|} = 0.2$.

At time-step zero, all filters start with the same a priori distribution, which is a Gaussian around the real position. The velocity is not known at the beginning and therefore set to a Gaussian around zero.

C. Results

We compare the grid filer without and with additional lane knowledge with a basic Kalman filter approach. The grid filter with lane knowledge essentially beliefs that vehicles usually stay in their lanes. Using the lanes as additional information to adapt the form of the update function has advantages, when the observed vehicle follows its lane, which should yield higher probability values for the tracked car. However, it should yield lower values when the tracked car switches lanes – possibly providing a measure of surprise, which may indicate a potentially dangerous event.

The Kalman filter approach is used as the benchmark comparison, which would yield the optimal solution for a linear system with Gaussian noise. Since the problem is non-linear, however, due to the non-linear trajectories during overtaking and additionally due to the curve at the end of the experiment, the grid approach should have considerable advantages. The question is whether the amount of non-linearity ameliorates the disadvantage due to the grid-based discretizations.

For evaluation purposes, we ran 50 runs in which the cars followed the exact same trajectories but in which the noise added to the simulated sensors was independent but identically distributed. For these runs with the respective filtering approaches we report mean and standard deviations of the expected location (center of mass) of car B as well as of the probability mass covering the ground truth state of car B.

Fig. 4 shows the results for the expected locations generated by each filter with radar noise data, plotting mean and standard deviations of the expected locations in corridor form. It can be seen that the mean of the Kalman filter includes the ground truth well, but it also shows rather high deviations. The grid approach without lane knowledge is slightly delayed upon the lane change but yields a lower deviation of its expectations. The grid approach with lane knowledge yields even lower deviations of its expected locations. Moreover, it shows a sort-of hesitation to trust the sensory information upon the lane change, because it expects the tracked vehicle to stay in its lane. This indicates that the grid approach may be suitably used to detect unexpected events.

Table I reports the average differences of the means from the ground truths within the three sections of the simulated maneuver: driving by, changing lane, and driving in front of vehicle A. While the Kalman filter means stay rather close to the ground truth, the standard deviations of the means suggest that the grid filters noise more effectively and maintains a more focused distribution. Moreover, in the case of the grid with lane knowledge, the large difference during the lane change indicates its low likelihood due the stay-in lane assumption. For the camera sensor settings, the results are generally comparable (Table I, bottom).

Fig. 5 confirms that the grid approach with lane knowledge can be utilized to detect surprising events. In this case, we plot the probability mass that surrounds the ground truth location of the tracked vehicle for Kalman filter (integrating over the same area that is covered by a grid node) and grid approach settings. The lane change commences at timestep 9, at which point the grid approach with lane knowledge shows

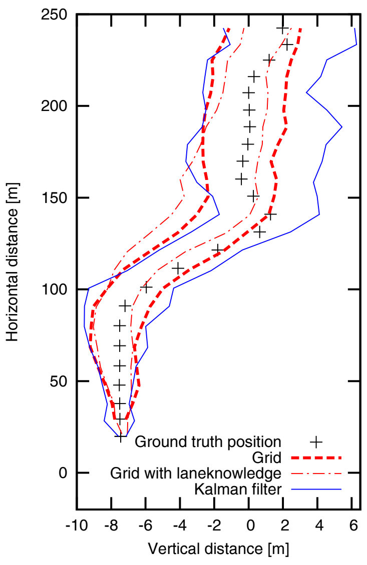


Fig. 4. Mean and standard deviations of the expectation values of the posterior distributions of the filters are compared with the actual overtaking vehicle position, plotting corridors of one standard deviation from the mean for each filter approach. Compared are the grid filter without lane knowledge, the grid filter with lane knowledge, and the Kalman filter.

TABLE I
AVERAGE EUCLIDEAN DISTANCES $dist$ BETWEEN REAL POSITION $\hat{x}$ AND ESTIMATED POSITION x AND AVERAGE STANDARD DEVIATIONS σ OF THE EXPECTED POSITIONS DURING THE DIFFERENT STAGES OF THE OVERTAKING MANEUVER

Stage	**Driving by**		**Changing Lane**		**In front**	
Radar Data	dist	σ	dist	σ	dist	σ
Grid (without lane)	0.49	1.51	1.81	2.08	0.62	2.24
Grid (with lane)	0.61	1.15	3.19	2.00	0.90	1.62
Kalman	0.60	2.83	0.84	4.57	0.72	4.50
Camera Data	dist	σ	dist	σ	dist	σ
Grid (without lane)	0.18	0.86	0.72	1.36	0.33	1.59
Grid (with lane)	0.16	0.89	1.19	1.23	0.32	1.39
Kalman	0.22	0.83	1.06	1.37	0.66	1.60

a drop in the covered probability mass, which continues for three timesteps – essentially until the grid “accepted” that the tracked car did change its lane after all. On the other hand, while the car does stay in its lane (early and late in the run) the grid approach with lane knowledge yields consistently higher probability values with lower deviations in comparison to the Kalman filter. The grid without lane knowledge yields means that are comparable to the Kalman filter but with lower standard deviations – thus yielding a more robust probability value. Table II confirms these observations, clearly pointing out the superiority of the grid with lane knowledge in all three segments – considering the higher distance during the unexpected lane change as a feature. The probability mass strongly decreases during the lane change and may thus be appropriately exploited to detect unexpected events.

In the camera sensor case (Table II, bottom), this effect cannot be observed as clearly because the high accuracy of the camera sensor in close proximity nearly overrules the lane-constraint. By adapting the influence of the lane marking a, this could be compensated. Since the angular accuracy in the camera sensor is much more accurate than in the radar model, the lane markings do not influence the results in the second and third stage of the overtaking situation as much, where the larger sensor noise is contained in the distance, but low uncertainty is maintained in the lateral direction.

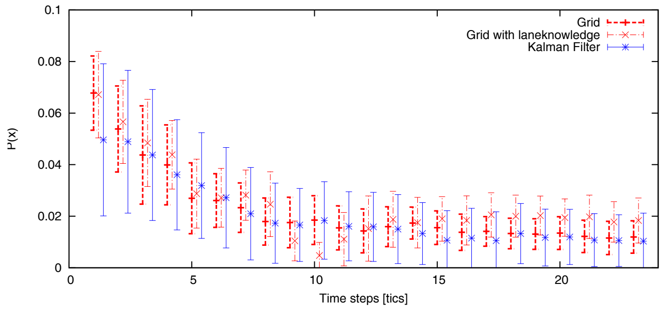


Fig. 5. The probability $P(x)$, and standard deviation, for the likelihood of the true state x during the overtaking situation with radar sensor information.

TABLE II

THE AVERAGE PROBABILITY $p(\hat{x})$ AT THE REAL POSITION $\hat{x}$ IN THE DIFFERENT STAGES OF THE OVERTAKING MANEUVER.

Stage	**Driving by**	**Changing Lane**	**In front**
Radar Data			
Grid (without lane)	0.0374	0.0168	0.0140
Grid (with lane)	0.0406	0.0121	0.0191
Kalman	0.0235	0.0163	0.0121
Camera Data			
Grid (without lane)	0.0744	0.0246	0.0151
Grid (with lane)	0.0775	0.0225	0.0159
Kalman	0.0331	0.0174	0.0104

V. CONCLUSIONS

This study has shown equal or better performance of our grid-based filter approach while tracking one vehicle over time in comparison with the standard Kalman filter approach. Due to the possibility to process any type of noise distribution, non-linear distributions can be handled more effectively with the grid-based approach. Also, multiple sources of sensory information, potentially with different noise characteristics, may be easily incorporated. Particularly highly non-Gaussian distributed sensors can be handled and integrated more effectively in comparison to a Kalman filter approach. Also, it is easily possible to merge the posterior distributions of a multi-modal grid approach, which may consist of separate grids for each sensor type (e.g. camera and radar sensors). Bayesian theory suggests that the combination of information gathered by different kinds of sensors will provide more precise information about the actual state of the system.

Moreover, we have shown that additional knowledge can be incorporated effectively and naturally, such as the assumption that cars do not change lanes usually. Of course, also an expected lane change could be modelled or also other knowledge, such as the behavior while approaching a stop-sign or a turning behavior when the turn-signal of the tracked car is activated. While similar constraints may be included in the Kalman filter approach, the Gaussian assumption would be immediately violated. In the grid-based approach, any type of constraint can be generally included. Besides the integration and combination of further sources of information, however, the grid approach also showed strong potential for identifying unexpected and thus potentially threatening events, which needs to be further evaluated.

The actual full power of the grid-based approach is expected to unfold when multiple objects – including pedestrians and obstacles – will be tracked and will interact within the grid representation. Repulsions and other interactions are then expected to occur and are being modelled at the moment.

REFERENCES

[1] K. Doya, S. Ishii, A. Pouget, and R. P. N. Rao, *Bayesian brain: Probabilistic approaches to neural coding*. The MIT Press, 2007.

[2] C. Cou, C. Pradalier, and C. Laugier, "Bayesian programming for multi-target tracking: an automotive application," in *In Proceedings of the International Conference on Field and Service Robotics, Lake Yamanaka (JP)*, 2003, pp. 99–208.

[3] C. Chen, C. Tay, K. Mekhnacha, and C. Laugier, "Dynamic environment modeling with gridmap: a multiple-object tracking application," in *Proc. of the Int. Conf. on Control, Automation, Robotics and Vision*, Singapour Singapore, 12 2006.

[4] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière, "Bayesian occupancy filtering for multitarget tracking: An automotive application," *International Journal of Robotic Research*, vol. 25, no. 1, pp. 19–30, 1 2006.

[5] M. Althoff, O. Stursberg, and M. Buss, "Erreichbarkeitsanalyse von Verkehrsteilnehmern zur Verbesserung von Fahrerassistenzsystemen," in *Proc. of 3. Tagung Aktive Sicherheit durch Fahrerassistenz*, 2008.

[6] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, USA: MIT Press, 2006.

[7] N. Sirola and S. Ali-Löytty, "Local positioning with parallelepiped moving grid," in *Proceedings of 3rd Workshop on Positioning, Navigation and Communication 2006 (WPNC'06)*, Hannover, 3 2006, pp. 179–188.

[8] D. Hähnel, D. Fox, W. Burgard, and S. Thrun, "A highly efficient FastSLAM algorithm for generating cyclic maps of large-scale environments from raw laser range measurements," in *Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, 2003.

[9] M. Bouzouraa and U. Hofmann, "Fusion of occupancy grid mapping and model based object tracking for driver assistance systems using laser and radar sensors," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2010, pp. 294 – 300.

[10] T. Gindele, S. Brechtel, J. Schröder, and R. Dillmann, "Bayesian occupancy grid filter for dynamic environments using prior map knwoledge," in *Proceedings of the IEEE Vehicles Symposium*, Xi'an China, 06 2009.

[11] S. Brechtel, T. Gindele, J. Vogelgesang, and R. Dillmann, "Probabilistisches Belegtheitsfilter zur Schätzung dynamischer Umgebungen unter Verwendung multipler Bewegungsmodelle," in *Proceedings of the 21th Fachgespräch Autonome Mobile Systeme*, 2009, pp. 49–56.

[12] S. C. Kramer and H. W. Sorenson, "Recursive bayesian estimation using piece-wise constant approximations," *Automatica*, vol. 24, pp. 789–801, 11 1988.

[13] (2007) TORCS, the open racing car simulator. [Online]. Available: http://torcs.sourceforge.net/

[14] D. Loiacono, P. L. Lanzi, J. Togelius, E. Onieva, D. A. Pelta, M. V. Butz, T. D. Lönneker, L. Cardamone, D. Perez, Y. Saez, M. Preuss, and J. Quadflieg, "The 2009 simulated car racing championship," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, pp. 131–147, 2010.