Stereovision-Based Multiple Object Tracking in Traffic Scenarios Using Free-Form Obstacle Delimiters and Particle Filters

Andrei Vatavu, *Member, IEEE*, Radu Danescu, *Member, IEEE*, and Sergiu Nedevschi, *Member, IEEE*

***Abstract*—In this paper we present a stereovision-based approach for tracking multiple objects in crowded environments where, typically, the road lane markings are not visible and the surrounding infrastructure is not known. The proposed technique relies on measurement data provided by an intermediate occupancy grid derived from processing a stereovision-based elevation map and on free-form object delimiters extracted from this grid. Unlike other existing methods that track rigid objects using also rigid representations, we present a particle filter-based solution for tracking visual appearance-based free-form obstacle representations. At each step, the particle state is described by two components, i.e., the object's dynamic parameters and its estimated geometry. In order to solve the high-dimensionality state–space problem, a Rao–Blackwellized particle filter is used. By accurately modeling the object geometry using the polygonal lines instead of a 3-D box and, at the same time, separating the position and speed tracking from the geometry tracking at the estimator level, the proposed solution combines the efficiency of the rigid model with the benefits of a flexible object model.**

***Index Terms*—Object tracking, particle filters, polygonal models, Rao–Blackwellization, stereovision.**

I. Introduction

THE SURROUNDING environment of a moving vehicle is filled with relevant objects of many types and shapes, all demanding the driver's attention. An advanced driving assistance system, which is designed to temporarily fulfill the driver's duties, particularly in cases when the human reaction time or the human attention span are not up to the task, needs an accurate representation of the driving environment in order to take the best decisions.

The most challenging objects of the driving environment are the dynamic ones, and for this reason, a considerable amount of research has been dedicated to the modeling and tracking of these entities [1], [4].

A dynamic entity is modeled by a set of parameters representing its geometry, its position, and its speed. A tracking mechanism estimates the value of these parameters over time, relying on features extracted from processing the raw data delivered by the available sensors. Despite the simplicity of the general idea, there are significant challenges in each step of the tracker's design. The problem of dynamic environment representation becomes a difficult task when the surrounding world is crowded, where, typically, the road lane markings are not visible and the surrounding infrastructure is not known. This may include the cases of traffic intersections, crowded urban centers, parking lots, or off-road scenarios. The tracking process must take into account multiple factors, such as the unpredictable nature of the obstacles, the measurement uncertainties or the occlusions. In addition, since the driving environment is composed of multiple static and dynamic objects that can be observed at the same time, the environment tracking system must be able to maintain and update the state of multiple objects at the same time, associating to each tracked object the proper measurement data.

There are many choices for the sensorial setup that provides the raw data to be analyzed and used as measurement in the tracking process. Most tracking techniques rely on the use of ultrasound [13], [27], laser [2], [7], [10], monocular [29], [30], or stereovision sensors [5], [6], [8], [9], [12], [14], [15]. As the imaging technology has become more reliable and cheaper, vision-based object modeling and tracking has been a very active research topic in recent years. In [1], the authors provide a survey of the past decade's progress in the vision-based vehicle detection for monocular and stereovision sensor configuration. While the monocular applications process the information in the image plane, the stereovision applications are able to process the information in 3-D space. The measurement data can be used as it is, by directly tracking, for example, 3-D point clouds [5], with each point being handled independently. Other stereovision-based tracking solutions try to reduce the computational cost by using intermediate representations, transforming the 3-D information into digital elevation maps [14], octrees [24], occupancy grids [28], [31], or stixel maps [15], [32].

When designing tracking solutions, one of the most important choices to be made is the choice of the object model, which is the state to be estimated over time. The model has to be at the same time representative for the object that is tracked, but it also has to be computationally efficient. Related work includes models such as polygonal lines [7], difference fronts [8], voxels [10], 2-D boxes [2], 3-D cuboids [6], or object contours [11], [17].

Manuscript received January 15, 2014; revised May 29, 2014 and August 15, 2014; accepted October 17, 2014. This work was supported by the Romanian Ministry of National Education, CNCS CNFIS, Project PN II PCCA 2011 3.2-0742 (SmartCoDrive) and Project PN-II-IDPCE-2011-3-1086 (MultiSens). The Associate Editor for this paper was F.-Y. Wang.

The authors are with the Computer Science Department, Technical University of Cluj-Napoca, 400114 Cluj-Napoca, Romania (e-mail: Andrei.Vatavu@cs.utcluj.ro; Radu.Danescu@cs.utcluj.ro; Sergiu.Nedevschi@cs.utcluj.ro).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2014.2366248

The most popular model for a dynamic traffic entity is the 3-D oriented box, which can be thought as a bounding box for the real obstacle in the world. This model is highly efficient, as it has a low-dimensional state vector, its evolution in time can be expressed by simple equations, and the measurements are easily associated to it. This model works well in simplified driving environments such as the highway, but it is less suitable for a complex environment, with many types of objects, which have a less cuboidal structure. Using simplified models such as the cuboid may lead the tracking process to incorrect results when the target pose estimation is affected by occlusions or by changes in its visual appearance. In order to overcome this problem, various algorithms that account for deformable object appearance have been proposed [16], [17], [30]. Typically, the model shape is implicitly represented [17], or by a set of fixed number of points. In particular, the authors in [17] describe a tracking method for slowly deforming and moving contours that are implicitly represented. Isard and Blake propose the CONDENSATION algorithm [18] for tracking parametric spline curves.

Once the target model is established, a tracking algorithm is usually developed starting from a popular probabilistic estimator. The problem of tracking is necessarily probabilistic as both the model of the object's state and the processed sensorial data are imperfect bits of knowledge about the world. The estimator should take into account the strong and the weak points of each information source and combine them for the best result.

The most popular estimators used for tracking are Kalman filters [6], particle filters [14], [16], [18], or hybrid methods [20], [21]. The traditional Kalman filter represents an optimal estimator, in which the posterior distribution is modeled by a Gaussian function. However, the classical Kalman filter solutions are only applicable to linear systems with unimodal distributions. The extended Kalman filter [25] allows nonlinear transformations for the state evolution and for the measurement mapping function, but still assumes that these transformations can be linearly approximated, at least in a reduced vicinity, and that the state is unimodal.

As an alternative, the particle filter approaches approximate the state space by a collection of N discrete samples, called "particles." Each particle represents a hypothesis about the system state. One of the main advantages of the particle filter-based solutions is the ability to handle nonlinear systems and multimodal distributions. However, particle filters are not suitable for high-dimensional state spaces as their computational complexity tends to exponentially grow with the number of state parameters. In order to handle this problem, different strategies can be found in the literature. For example, in [19], the unscented Kalman filter is used to propagate the state distribution, so that the number of sampled particles is reduced. In [20], the Rao–Blackwellized particle filter (RBPF) is introduced. The key idea of the RBPF approach is that a part of the state space can be analytically updated, whereas another part of the state is sampled. In [21], the RBPF is applied for simultaneous localization and mapping. The robot pose is estimated with a particle filter. In addition, the state vector is represented by N landmarks. Each landmark position is updated by using a 2×2 extended Kalman filter (EKF). In [2], a RBPF technique is applied for model-based vehicle tracking. For simplicity, the vehicle shape is approximated by a rectangle.

Our research team at TU Cluj-Napoca has been involved in the field of stereovision-based driving environment perception since 2001. Using sparse edge-based stereovision, the obstacles' position, size, and speed were tracked using a nonoriented cuboid model [6], as the limited 3-D information was not suitable in a more detailed perception. When real-time dense stereovision solutions became available, they were used for a much more detailed perception of the environment. The particular characteristics of the driving environment allowed us to simplify the dense 3-D information in the shape of elevation maps, whose cells could be labeled, based on the coordinates, density, and other characteristics of the associated points, into drivable (road cells), obstacle cells, intermediate (sidewalk) cells, and cells with no measurement data [12]. Significant improvements to the elevation map as a model for dynamic environments, which include modeling and tracking the speed of each map cell and additional gray-scale information that enhance the perception of the 3-D environment, are presented in [39] and [40].

The cells labeled as obstacles in the elevation map signal obstacle areas in a 2-D bird-eye view of the road environment and can be regarded as an unfiltered raw occupancy grid. From this point, two directions of research have been followed. One direction consists in grid tracking at cell level, trying to refine, for each cell, the probability that it is an obstacle cell (occupancy probability), and computing a speed vector for it, either using the raw occupancy grid directly [28], or speeding up the computation by using difference fronts between raw grids [8].

The other direction is model-based object tracking using the obstacle cells of the grid as measurement data. A particle-based solution for tracking objects modeled as nonoriented cuboids using the obstacle cells of the elevation map is presented in [14]. This model was, however, quite limited compared with the quality of the measurement information, and therefore, methods for taking advantage of the shape information were devised. This paper [9] presents a method for extracting individual objects from the occupied elevation map cells, in the shape of polygonal lines. The approach presented in [3] tracks these delimiters using the iterative closest point (ICP) for association between past and present contours. The solution proposed in [42] tries to improve the representation consistency by taking into account the persistence of occupied grid cells. Both tracking methods use Kalman filter for state update, but does not allow the shape of the object to change—the geometry of the object is assumed fixed.

The method described in [33] is able to track the object's geometry along with its position and speed, by using a set of Kalman Filters for the control points (landmarks) of the object's contour, which define the object's shape, and one Kalman filter for tracking the motion of the whole object.

There are two problems with the Kalman filter: 1) the unimodal nature of the approximation of the probability density of the target's state; and 2) the need of data association. In face of these problems, a particle filter presents itself as the natural alternative. However, modeling the whole state of the object (position, speed, and geometry) by means of particle filters

would require a large number of particles, and therefore a mixed solution, combining the efficient Kalman filter for tracking the object's geometry, and the versatile particle filter for tracking the position and speed, is the solution described in this paper, an extended version of [41].

The method presented in this paper is designed to estimate the position, the speed, and the geometry of objects from noisy stereo depth data. The modeling and tracking solution rely on measurement information provided by an intermediate occupancy grid and on free-form object delimiters extracted from this grid. The intermediate representation is derived from processing a stereovision-based elevation map, whereas the attributed polygonal object representations, are obtained by radial scanning this map by using the BorderScanner algorithm. Unlike other existing methods that track rigid objects using also rigid representations, we present a particle filter-based solution for tracking visual appearance-based free-form obstacle representations. In order to increase the object model flexibility, each particle is described by two components, i.e., the object's dynamic parameters and its estimated geometry represented by a vector of control points. Another essential contribution of the proposed approach is that the high-dimensionality state–space problem is solved by adopting a RBPF is used.

The proposed method takes into consideration the stereo uncertainties. By accurately modeling the object geometry using the free-form polygonal lines instead of a 3-D box, and, at the same time, separating the geometry tracking from position and speed tracking at the estimator level, the proposed solution combines the efficiency of a rigid model (small state space, simple state equations, simple prediction, compact representation) with the benefits of a flexible object model (a model that is able to adapt its shape according to the most recent measurement).

This paper is structured as follows: the next chapter presents the overall system architecture, the object model is described in Section III, the proposed multiple object tracking approach, and its main steps are detailed in Section IV, whereas the last two sections show the experimental results and the conclusion about this paper.

II. System Overview

The overall system architecture is based on two main modules, i.e., preprocessing and tracking (see Fig. 1).

The *preprocessing* module involves some operations that are performed before the obstacle tracking such as image acquisition or stereo reconstruction tasks. The stereo reconstruction is performed either offline or onboard. For the offline stereo reconstruction, we use an semiglobal matching (SGM) [43] technique that is implemented on GPU [35], [38]. As the onboard system is constrained by power requirements, the online stereo reconstruction is performed with a dedicated TYZX board [26]. The reconstructed stereo data is used to generate a more compact 2.5-D grid-based representation [12], in which each cell is classified based on its height value as obstacle, traffic isle, or road [see Fig. 2(b) and (d)]. The ground plane projection of this intermediate representation is used to extract free-form object delimiters [see Fig. 2(c) and (d)] and to compute a probabilistic measurement model.

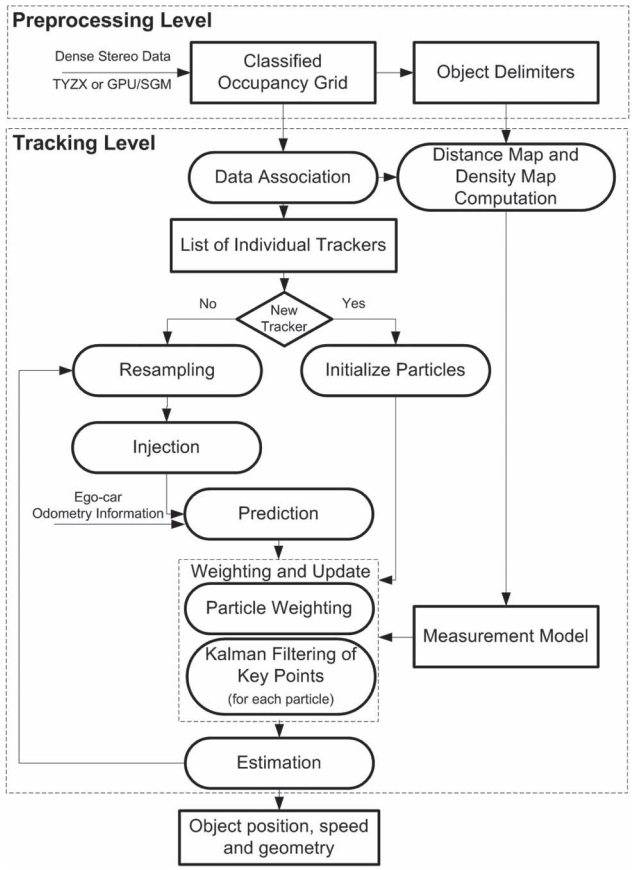

Fig. 1. Multiple object tracking. System architecture.

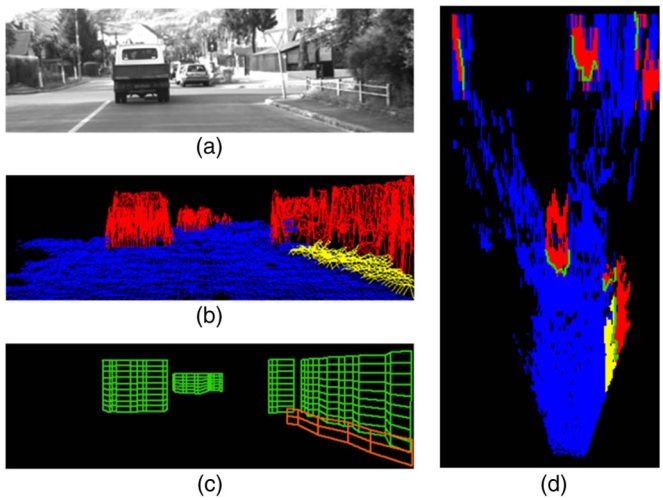

Fig. 2. (a) Gray-scale left image. (b) Digital elevation map with classified cells as (blue) road, (red) object, and (yellow) traffic isle. (c) Object delimiters. (d) The elevation map is projected on the ground plane. (Top view) The extracted delimiters are illustrated with green.

The *tracking* module performs optimal state estimation for each individual object. First, the data association is applied in order to assign new measurements to the existing tracks and to initialize new ones. Then, for each existing target, the following processing tasks are performed, i.e., state prediction, Kalman filtering of object geometry, particle weighting, estimation, resampling, and injection. These steps will be detailed in the next sections.

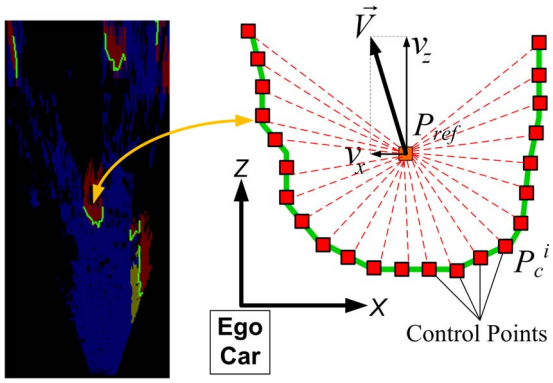

Fig. 3. (Left) Classified occupancy grid and the extracted object delimiters (top view). (Right) Object model. An object from the traffic scene is represented by N control points P_c^i (polygonal vertices), and a reference point P_{ref}.

III. Object Model

Even the most tracked obstacles in a driving environment are considered to be rigid (cars, poles, walls, side fences, etc.), their estimated visual appearance is changing over time. This is caused by several factors. For example, the same obstacle can be seen from different observation points (when the ego vehicle is moving), the obstacles gradually appear or disappear from the visibility area of the ego vehicle or because the objects are partially occluded by other crossing obstacles. Therefore, the shape of a tracked object may be adapted according to the most recent measurements. In order to provide better flexibility, each object is represented by a free-form model (see Fig. 3) with the following attributes.

1) The object position described by a reference point $P_{\text{ref}}(x_{\text{ref}}, z_{\text{ref}})$ in the coordinate system of the camera. Initially, the reference point is set to be in the center of mass of the occupancy grid blob that describes the object. Its position is recursively updated by the tracking algorithm. The reference point remains fixed in the world coordinate system when dealing with static obstacles and is updated with the estimated translations when dealing with dynamic obstacles. It must be noted that the coordinate system has its origin in front of the ego vehicle, with X-axis oriented toward the right and Z-axis pointing toward the moving direction of the host vehicle.
2) The obstacle velocity $\vec{V}(v_x, v_z)$.
3) A list of control points describing the obstacle shape $\{P_c^i(x_c^i, z_c^i) | i = [1 \dots N_c]\}$. In the initialization step, the control points are determined by choosing N points that are uniformly distributed along the object contour. The number of control points is fixed and is the same for all obstacles. Each control point $P_c^i(x_c^i, z_c^i)$ is defined by its relative position $L^i(l_x^i, l_z^i)$ to the object reference point P_{ref}. At each frame, the control points are updated by the tracking mechanism with the new measurements extracted by using the BorderScanner algorithm. More details are given in Section IV-E.

Having the parameters previously described, the overall object state at time t can be represented as

$$S_t = [x_{\text{ref}}, z_{\text{ref}}, v_x, v_z, L^1, L^2, \dots, L^N]^T \quad (1)$$

being described by two main components: the object dynamic part $X_t = [x_{\text{ref}}, z_{\text{ref}}, v_x, v_z]^T$ and its geometry component $G_t = [L^1, L^2, \dots, L^N]^T$

$$S_t = [X_t, G_t]^T. \quad (2)$$

IV. Obstacle Tracking

A Bayesian solution to the tracking problem consists in estimating, recursively in time, the current object state S_t, given all observations $Z_{1:t} = \{Z_1, Z_2 \dots, Z_t\}$ collected up to the current time t

$$p(S_t|Z_{1:t}) = \eta p(Z_t|S_t) \int_{S_{t-1}} p(S_t|S_{t-1}) p(S_{t-1}|Z_{1:t-1}) \quad (3)$$

where $p(Z_t|S_t)$ describes the *observation model*, the $p(S_t|S_{t-1})$ term denotes the *state transition probability* from S_{t-1} to S_t, and η represents the normalization constant.

A. RBPF

In a particle-based filtering solution [16], [18] the object state probability is approximated by a set of N weighted particles $p(S_t \approx \{S_t^i, w_t^i, i = [1 \dots N]\}$. Each particle S_t^i represents a hypothesis of the state of the object at a given time t. Therefore, object tracking consists in estimating the best state by evaluating the samples S_t^i and their attached weights w_t^i, given a motion model and a measurement model. A disadvantage of the classical particle filtering algorithm is that it is not suitable for high-dimensional state spaces. Usually, its computational complexity grows exponentially with the number of state parameters. The "Rao–Blackwellization" process consists in estimating a part of the object state analytically, thus reducing the number of dimensions and the computational cost of the particle filter mechanism. By dividing the full object state S_t into a dynamic component X_t and a geometry part G_t, the entire posterior density $p(S_t|Z_{1:t})$ is defined as

$$p(S_t|Z_{1:t}) = p(X_t, G_t|Z_{1:t}) \quad (4)$$

and can be factored as

$$p(S_t|Z_{1:t}) = p(G_t|X_t, Z_{1:t}) p(X_t|Z_{1:t}). \quad (5)$$

The first probability density $p(X_t|Z_{1:t})$ denotes the object position and velocity and is approximated by a set of weighted samples $\{X_t^i, w_t^i, G_t^i, i = [1, \dots, N]\}$. The second density $p(G_t|X_t, Z_{1:t})$ represents the obstacle geometry posterior distribution conditioned on its position, speed, and all observations up to the time t. Each control point in G_t is represented by a mean $\hat{L}^j$ and a covariance matrix $\sum^j$ and is estimated analytically by applying a 2×2 Kalman filter. The particle set can be now defined as

$$\left\{ q_t^i | q_t^i = \left[X_t^i, w_t^i, \left(\hat{L}^1, \sum^1 \right), \dots \left(\hat{L}^j, \sum^j \right) \right]^T \right\} \quad (6)$$

where $i = [1, \ldots, N]$ and $j = [1, \ldots, N_c]$. For each individual target, the proposed tracking solution can be decomposed into several steps. In the first phase, the tracked obstacle's dynamic parameters (position and speed) are estimated based on the new observations through the particle filtering. In the second phase, using Kalman filters, the key point positions of each particle are recursively updated by taking into account the new estimated dynamic state. In the last phase, each key point position is estimated by using a weighted average. The weights are provided by the associated particles. Next, we will present the main steps involved in our object tracking solution.

B. Data Association

In the data association step, the task is to assign new measurements to the existing individual trackers and to create new ones.

Before applying the data association, we also must take into account the ego-car motion in order to separate its speed from the independent motion of the tracked participants. In our case, the velocity v and the yaw rate $\dot{\psi}$ information are provided by the host vehicle sensors. Therefore, the position state parameters $(x_{\text{ref}}, z_{\text{ref}})$ of each particle are transformed by applying the ego-car motion model with constant speed and constant yaw rate

$$\begin{bmatrix} x_{\text{ref\_c}} \\ z_{\text{ref\_c}} \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} x_{\text{ref}} \\ z_{\text{ref}} \end{bmatrix} - \begin{bmatrix} t_x^{ego} \\ t_z^{ego} \end{bmatrix} \quad (7)$$

where Δt is the time delay between two frames and $\psi = \dot{\psi}\Delta t$ represents the vehicle rotation angle around the Y-axis and $T = [t_x^{ego}, t_z^{ego}]^T$ is the ego-car translation.

Next, the data association is performed by computing overlapping scores w_{ij} between grid blobs at consecutive frames. We define a blob as a collection of connected grid cells that are occupied. For each blob entity A from the previous frame and for each blob B from the current frame the following distance metric is calculated:

$$w_{AB} = |A \cap B| = \sum_{i=1}^{N_A} \sum_{j=1}^{N_B} O(a_i, b_j) \quad (8)$$

where $O(a_i, b_j)$ denotes the overlap function between two cells from A and B, N_A represents the number of cells in the blob A, and N_B is the number of cells in the blob B. The value of $O(a_i, b_j)$ is 1 when the two points a_i and b_j overlap and 0 otherwise.

As the result, a score matrix $\mathbf{W} = \{w_{ij}\}$ is generated. Given matrix $\mathbf{W}$, two types of association are determined: forward association (the most likely association from A to B)

$$\text{Assoc}(A) = \arg\max_B p(B|A) = \arg\max_B \frac{w_{AB}}{N_A} \quad (9)$$

and backward association (the most likely association from B to A)

$$\text{Assoc}(B) = \arg\max_A p(A|B) = \arg\max_A \frac{w_{AB}}{N_B}. \quad (10)$$

This double association allows us to consider the cases when a larger object's blobs are split into multiple disjoint sets or *vice versa*.

Having the two sets of blobs described as $S_A = \{A_i | i \in [1 \ldots M]\}$ and $S_B = \{B_j | j \in [1 \ldots N]\}$, the final list S of distinct association hypotheses is defined as

$$\begin{aligned} S = &\{(A_i, B_j) | \text{Assoc}(A_i) = B_j, i \in [1 \ldots M], j \in [1 \ldots N]\} \\ &\cup \{(B_j, A_i) | \text{Assoc}(B_j) = A_i, \text{Assoc}(A_i) \neq B_j, \\ &\quad i \in [1 \ldots M], j \in [1 \ldots N]\}. \end{aligned} \quad (11)$$

As the aim of association is to generate a set of distinct hypotheses pairs, associations from B to A must not repeat the associations from A to B, but rather gather the association pairs that have not been generated when searching from set A to set B. For this, we introduced an extra constraint in the second part of (11), $\text{Assoc}(A_i) \neq B_j$.

C. Initialization

The initialization step is applied when new association hypotheses (not tracked objects) are detected. This is achieved by comparing the list of associated blobs with the existing list of individual trackers. The object state is initialized as in the following.

1) Initializing the Object Speed: The motion parameters describing the initial state are estimated by applying a fast pairwise alignment of the associated delimiter pairs (from the previous and current frames). For this, we use the ICP algorithm, previously described in [3]. For each association hypothesis, we define two set of points: a model set $P = \{p_1, p_2, \ldots, p_M\}$ that describes the object delimiters in the previous frame, and a data set $Q = \{q_1, q_2, \ldots, q_K\}$ that describes the object delimiter in the current frame. For each point q_j from Q the corresponding closest point p_i from P is determined. Having a set of corresponding points (p_i, q_j), an optimal rotation $\mathbf{R}$ and translation $\mathbf{T}$ is computed by minimizing the alignment error

$$E(\mathbf{R}, \mathbf{T}) = \sum_{i=1}^{N} \|\mathbf{R}p_i + \mathbf{T} - q_i\|^2 \quad (12)$$

where N is the number of point-to-point correspondences. The number of corresponding points varies depending on the length of the two contours P and Q (model and data contour).

2) Initializing the Object Position: A set of initial random object hypotheses are generated around the measurement position

$$\left\{ q_0^i | q_0^i = \left[X_0^i, w_0^i, G_0^i\right]^T, i = [1 \ldots N] \right\}. \quad (13)$$

3) Initializing the Object Geometry: Each particle is initialized with the object geometry G_0 that is extracted from the measurement delimiters. It must be noted that a small amount of new particles (including new hypotheses for object position and geometry) are added in the *Injection* step.

D. Prediction

The prediction task consists in generating the new population of particles at time t from the previous set S_{t-1} given a state transition model $p(S_t|S_{t-1})$. First, the particles are moved by applying a deterministic drift based on the target dynamics.

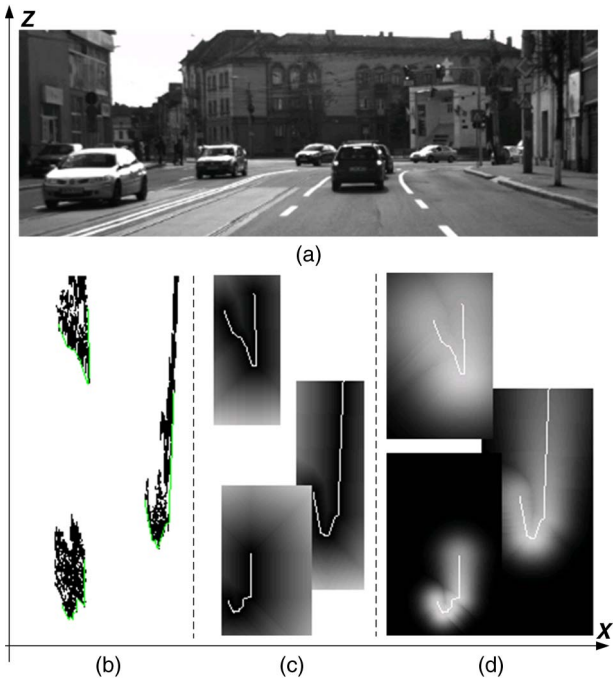

Fig. 4. (a) Left camera image. (b) Occupancy grid projected on the ground plane. The obstacle delimiters are colored with green. (c) Distance transform of the extracted delimiters. (d) The density map is generated by taking into account stereo uncertainties and distances to the closest delimiter points. High intensities indicate high measurement probability.

Then, each predicted sample state is altered according to a random noise.

Each particle's position and speed $X_t = [x_{\text{ref}}, z_{\text{ref}}, v_x, v_z]^T$ is predicted by using the standard constant velocity model

$$\begin{bmatrix} x_{\text{ref}} \\ z_{\text{ref}} \\ v_x \\ v_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{\text{ref\_c}} \\ z_{\text{ref\_c}} \\ v_x \\ v_z \end{bmatrix} + w. \tag{14}$$

The matrix multiplication describes the deterministic drift component. The stochastic part is defined by the random noise $w \sim N(0, Q)$, which is drawn from a zero mean Gaussian distribution having the covariance matrix $\mathbf{Q}$. As in our case, we use a constant velocity model, the covariance matrix Q is selected considering an experimentally adjusted covariance accounting for the obstacles' possible acceleration.

E. Measurement Update

The purpose of this stage is to assign new weights to the predicted particles and to update the object geometry for each particle. First, the raw object delimiters are extracted from the current occupancy grid. Then, the new particle weights are computed by evaluating the alignment error between the measurement and the predicted hypotheses. Finally, the key point positions of each individual particle are updated taking into account the new estimated dynamic state.

1) Object Delimiter Extraction: The obstacle delimiters are extracted from the occupancy grid, at each frame, by using the *BorderScanner* algorithm previously described in [9] (see Fig. 4). The main idea of the *BorderScanner* technique is to extract a contour C_{meas} for each object by accumulating the most visible grid cells c_i that are occupied

$$C_{\text{meas}} = \{c_i | \text{Occ}(c_i) = \text{true}, i \in [1, \ldots, M_c]\}. \tag{15}$$

This is achieved by using a virtual ray that extends from the ego-car position and traverses the grid map in a radial direction with fixed steps. The closest cells that are occupied are collected into the contour list C_{meas}. In (15), M_c is the number of extracted contour points, and $\text{Occ}(c_i)$ is the occupancy state of the measurement grid cell c_i.

2) Computing Stereo Uncertainties: The next step is to compute the stereo uncertainties. As suggested by [37], we can approximate the lateral σ_x error and the longitudinal error σ_z as

$$\sigma_z = \frac{z^2 \cdot \sigma_d}{b \cdot f}, \quad \sigma_x = \frac{\sigma_z \cdot x}{z} \tag{16}$$

where x and z are the 3-D coordinates of a point, σ_d denotes the disparity error in pixels, f is the focal distance, and b defines the distance between the left and right cameras (baseline).

3) Computing the Distance to the Measurement: The aim of this step is to determine a distance metric between any occupancy grid cell and a corresponding measurement point. First, for each obstacle, we select a region of interest (ROI) covering all generated particles around the measurement contour C_{meas}. Then, for each cell $m_{dm}(x_{dm}, z_{dm})$ in the ROI, we compute two parameters: a distance d_m to the closest measurement cell $c_j(x_{del}, z_{del})$, and its position, where dm represents the index of a given cell in the local map, and del is the index of the closest delimiter point. The resulted values are stored in a distance map.

The probability density map [see Fig. 4(d)] can be determined now by converting the distance values d_m of each point m_{dm} according to

$$\pi_{xz} = \frac{1}{2\pi\sigma_x\sigma_z} \exp\left(-\frac{1}{2}\left[\frac{(x_{dm}-x_{del})^2}{\sigma_x^2} + \frac{(z_{dm}-z_{del})^2}{\sigma_z^2}\right]\right) \tag{17}$$

where σ_x and σ_z represent the stereo uncertainties of the corresponding measurement point. As each cell has its own uncertainty, the σ_x and σ_z errors are determined for each measurement cell according to (16) considering that the disparity error is about 0.25 pixels in the case of a good stereo-reconstruction system. Both parameters define the confidence of the system in the measurement model. The confidence of the results is inversely correlated with the value of these parameters. All computed weights are stored in a density map.

4) Weighting: This step consists in assigning new weights w_t^i to the delimiter hypotheses q_t^i based on their likelihood

$$p\left(Z_t | X_t = X_t^i, G_t = G_t^i\right). \tag{18}$$

First, we need to define a distance metric between a given particle and a given observation. This is achieved by estimating an alignment error between object hypotheses and the measurement data. For each control point L^j from the particle q_t^i,

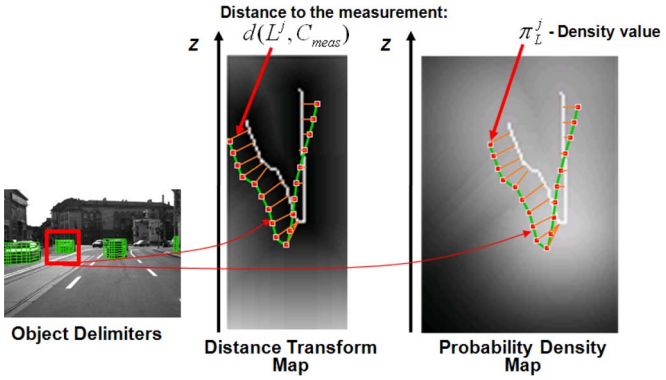

Fig. 5. Euclidean distance and the weight metrics are determined by superimposing the particle model on the two maps: distance transform map and probability density map.

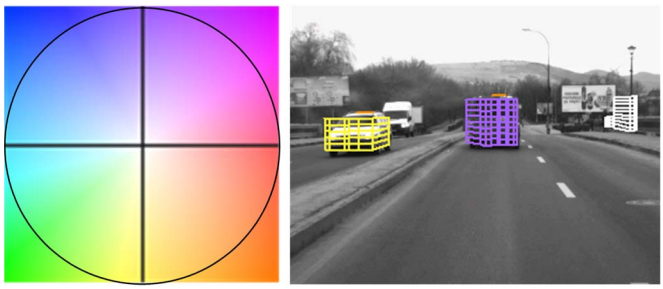

Fig. 6. (Left) Color encoding of object speed. Color hue describes the orientation of a moving obstacle, whereas the saturation describes its magnitude. (Right) Example with labeled obstacles according to their speeds.

we determine the closest corresponding point c_k from the measurement C_{meas}

$$d(L^j, C_{\text{meas}}) = \min_{k \in \{1,\dots,M_c\}} d(L^j, c_k) \tag{19}$$

where M_c is the number of measurement points in C_{meas}. In order to consider the stereo uncertainties, we also assign a density value π_L^j to each corresponding pair (L^j, c_k). The Euclidean distance $d(L^j, C_{\text{meas}})$ and the weight π_L^j metrics are determined by superimposing the particle model on the two maps estimated in the previous step (see Fig. 5). The alignment error is computed according to

$$D_{\text{alignment}} = \sum_{j=1}^{N_c} \frac{\pi_L^j \cdot d(L^j, C_{\text{meas}})}{\sum_{k=1}^{N_c} \pi_L^k}. \tag{20}$$

Finally, the overall particle weight w_t^i is computed as

$$w_t^i = \frac{1}{2\pi\sigma_D} e^{-\frac{1}{2}\frac{D^2_{\text{alignment}}}{\sigma_D^2}}. \tag{21}$$

5) Kalman Filtering: Having a population of weighted particles describing the belief about the object position and speed, we also need to update the belief about the object shape as soon as new observations are available. Given an individual particle q_t^i, its geometry component G_t^i is described by a list of control points. Therefore, for each control point, we apply a 2×2 Kalman filter to estimate its state $\hat{L}^j = [l_x^j, l_z^j]^T$ and covariance $\sum^j$. The Kalman filter input measurements are determined by choosing N equidistant points along the measurement contour extracted in the step 1. The measurement covariance matrix $\mathbf{R}$ of each control point is computed, by considering the stereo uncertainties σ_x and σ_z defined in the step 2.

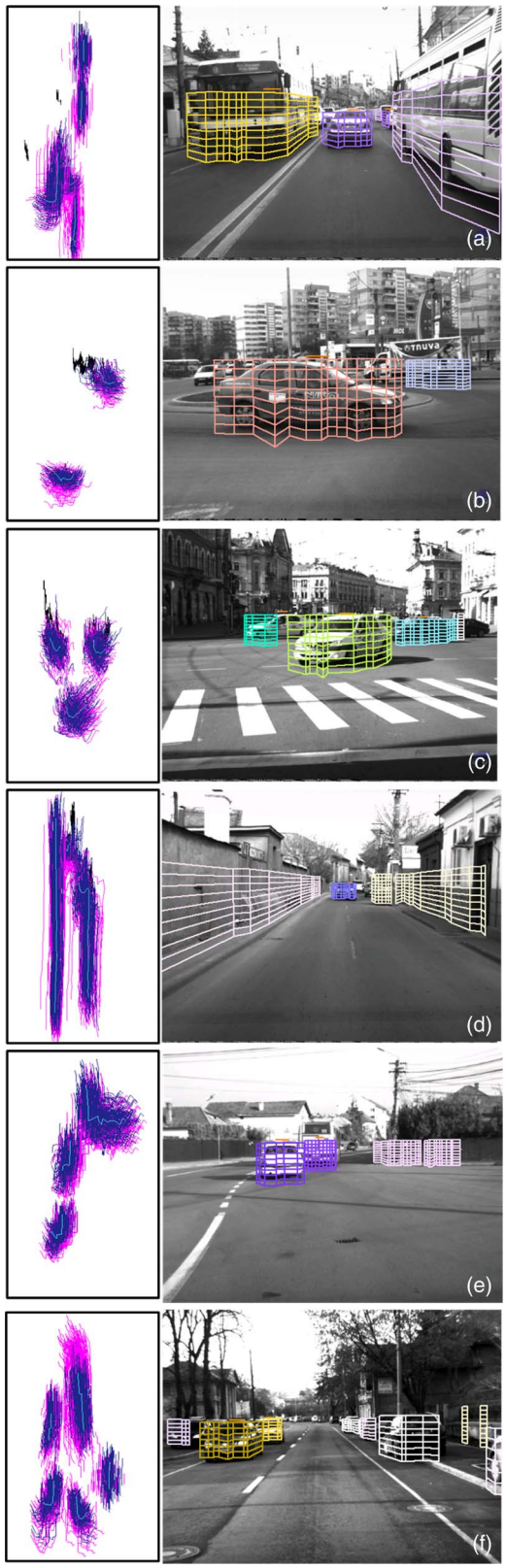

Fig. 7. Tracking multiple objects in various traffic scenarios.

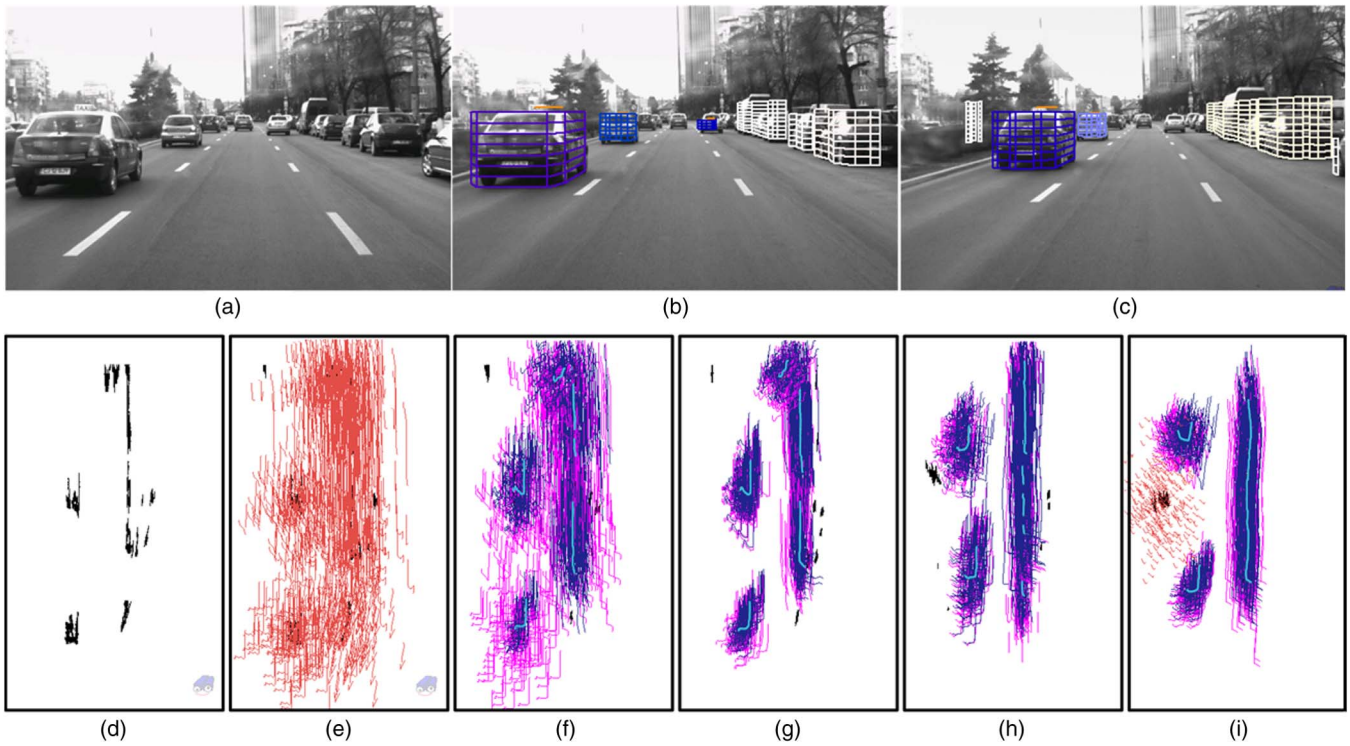

Fig. 8. Multiple object tracking. (a) Image of a traffic scene. (b) and (c) Result of the static and dynamic object representation, 10 and 25 frames later. (d) Measurement occupancy grid projected on the ground plane. (e) Particle population after initialization. (f)–(h) Individual object tracker particles, evolving in time. (i) Initialization step applied to a newly detected tracker corresponding to the tree in figure (c).

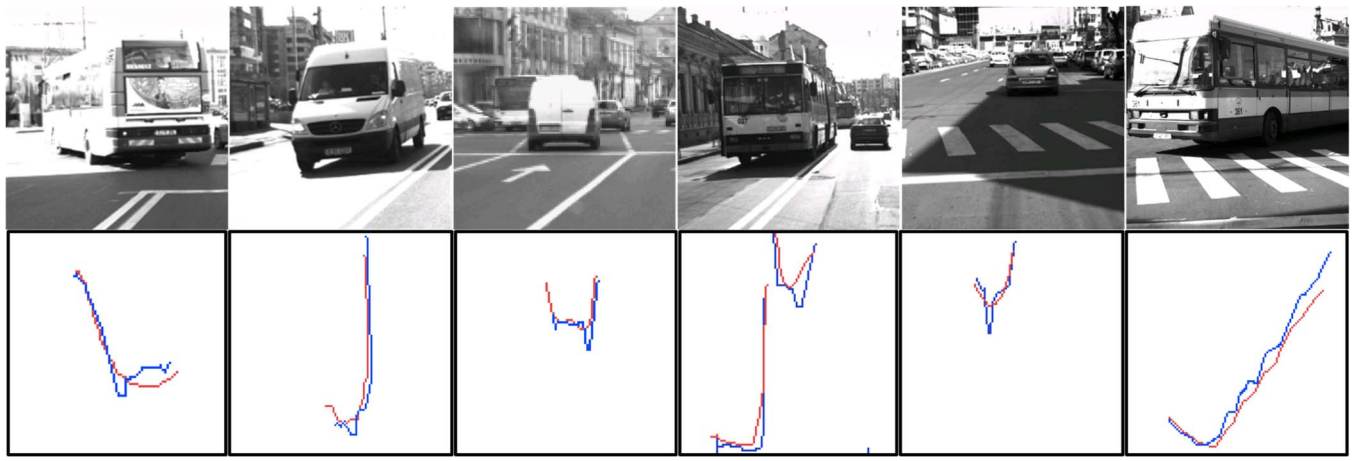

Fig. 9. Single frame object shape measurements (blue delimiters) versus tracked geometry (red).

F. Estimation

The current mean state at time t is estimated by using a weighted average of the particle states

$$\hat{S} = \sum_{i=1}^{N} w_t^i S_t^i. \tag{22}$$

G. Resampling and Injection

The resampling step consists in drawing from the previous particle set with a sampling probability proportional to the assigned weights. Thus, the particles with low importance are removed, whereas the samples with large weights are replicated.

However, there are cases when sharp changes in the traffic scene may lead to the estimation of erroneous states. This may happen due to the fact that there are no sufficient hypotheses in the vicinity of the true. This is also known as the particle deprivation problem. As a solution, we introduced an *injection* step, where a small amount of particles with low importance are replaced with new completely random samples that are drawn around the measurement, a common approach for preventing particle deprivation, as described in [36]. Through the *injection* step, we also introduce new hypotheses for object geometry.

V. Experimental Results

The proposed multiple object tracking solution has been tested on various sequences of urban traffic situations, acquired in Cluj-Napoca, Romania. We have conducted two types of evaluations: qualitative assessment and quantitative assessment. For both types of tests, we used an Intel Core 2 Duo Computer at 2.66 GHz and 4 GB of RAM. The size of the occupancy grid used in our method is 240 rows $\times$ 500 columns (0.1 m $\times$ 0.1 m cells). In order to evaluate the accuracy of the tracking method, we have performed two types of experiments. In the first case, the ground-truth measurements are provided by high-performance Global Navigation Satellite System (GNSS) receivers mounted on the ego vehicle and on the target vehicle. The second set of experiments is performed on the KITTI raw data set [34].

A. Qualitative Evaluation

The qualitative assessment was performed on various real-traffic scenarios. In order to prove the ability of the system to correctly identify the speed of the objects, each individual obstacle is labeled according to the estimated speed by following the Middlebury color coding style [22]. As presented in Fig. 6, the color hue describes the orientation of a moving obstacle, whereas the saturation describes its magnitude (e.g., blue—for outgoing objects, yellow—for incoming objects). Each static or dynamic object is represented by a free-form delimiter and a speed vector (orange). The free-form delimiter's projection in the road plane is a polygonal line, and the height of the obstacle is the maximum height of the object's associated elevation map cells (see Fig. 7).

Fig. 8 illustrates the results of the proposed multiple object tracking approach, including intermediate frames with the particle distributions and the labeled obstacles. Fig. 8(e) presents the case when all new objects' trackers are initialized for the first time. Usually, this occurs at the beginning of a sequence when the list of individual trackers is empty. It can be seen that the initial random hypotheses are clustered around each individual object [see Fig. 8(f)]. Moreover, the resulted particle distributions converge over time [see Fig. 8(g) and (h)]. The estimated mean state is colored with light blue. The predicted samples are colored with magenta. The picture also shows the influence of weighting and resampling steps (dark blue) on the predicted population of particles. Fig. 8(i) presents a particular case when the initialization step is applied to a newly activated tracker. A set of initial random object hypotheses (red) are drawn around the measurement position.

Fig. 7 presents some results from real traffic scenes, emphasizing the following specific cases:

1) dynamic objects of different size: two buses and a car [see Fig. 7(a)];
2) two objects moving in a roundabout [see Fig. 7(b)];
3) partially visible objects: moving in an intersection [see Fig. 7(c)], outgoing [see Fig. 7(e)], or incoming [see Fig. 7(f)];
4) obstacles of different types: static walls and a moving object [see Fig. 7(d)].

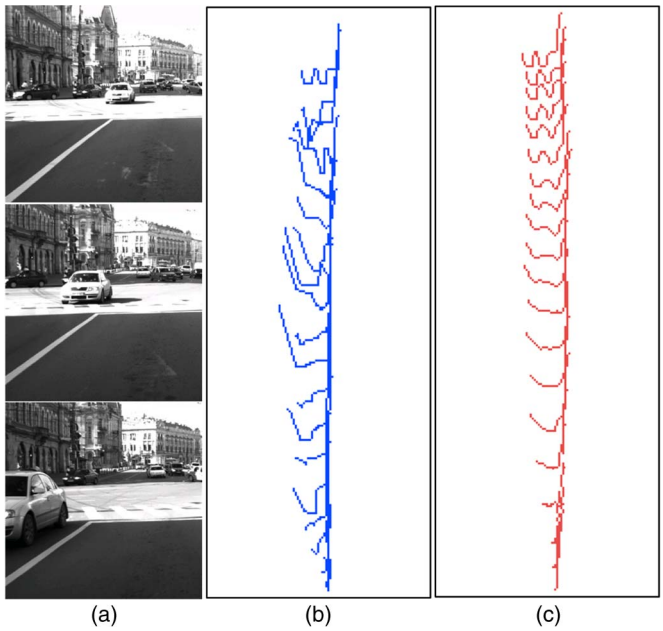

Fig. 10. Evolution of the object shape estimation in time. (a) Incoming vehicle. (b) Car trajectory (top view) described by a sequence of single frame measurements (blue color). (c) Tracked model (red). The object position and its geometry are gradually updated over time.

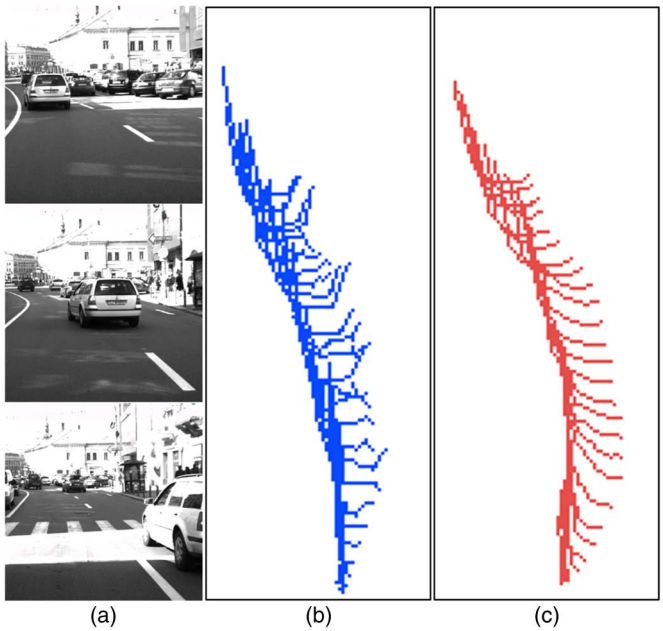

Fig. 11. Evolution of the object shape estimation in time. (a) Outgoing vehicle. (b) Car trajectory (top view) described by a sequence of single frame measurements (blue color). (c) Tracked model (red). The object position and its geometry are gradually updated over time.

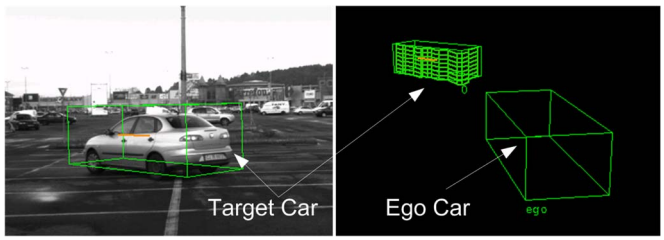

Fig. 12. Three-dimensional box is generated using ground-truth data from the high-accuracy GNSS receivers that were mounted on the Ego Car and on the Target Car. The target box is fitted over the detected obstacle (right).

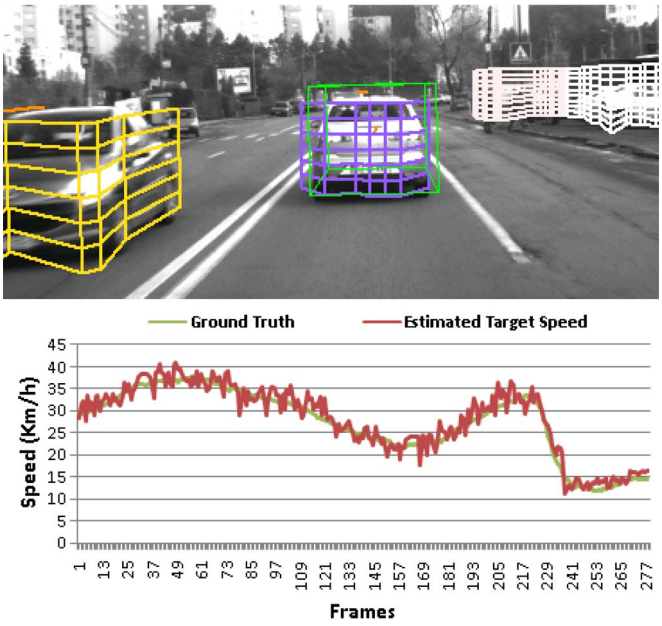

Fig. 13. (Top) Urban traffic scenario including the target car (blue, indicating an outgoing motion). The 3-D box (green) is generated by using ground-truth information and is fitted over the target vehicle. (Bottom) The estimated target speed is shown with red. The ground-truth speed is illustrated with green.

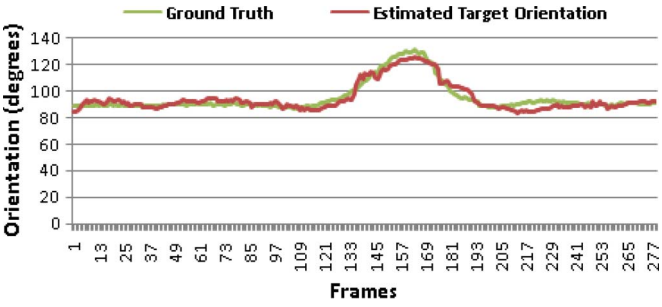

Fig. 14. Estimated target orientation is shown with red. The ground truth is illustrated with green.

It can be observed that the particle population for each object (dark blue contours) is spread out according to the uncertainties of the stereo measurement process, but the estimation generated from the particles (depicted with light blue) follows closely the real contour of the object.

Fig. 9 shows comparative results between the frame by frame measurement delimiters, obtained by applying the Border Scanner algorithm (blue) and the resulted estimated object geometry after tracking (colored with red).

Figs. 10 and 11 show how the object position and geometry are updated over time (red) by taking into account the sequence of noisy measurements (blue). It can be observed that the shape of the tracked model is gradually changing as the target vehicle is moving along its trajectory.

B. Numerical Evaluation by Using High-Accuracy GNSS Receivers

In order to be able to quantify the performance of the proposed method, we have used the ground-truth information

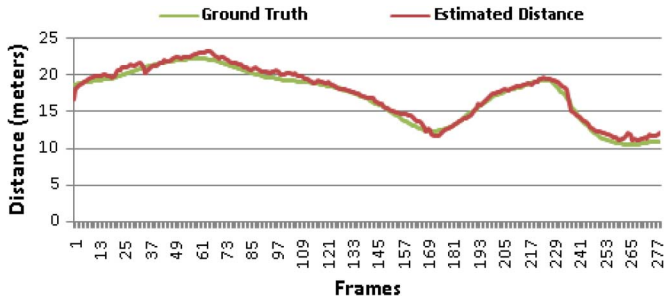

Fig. 15. Estimated distance to the target vehicle. The ground-truth distance is illustrated with green.

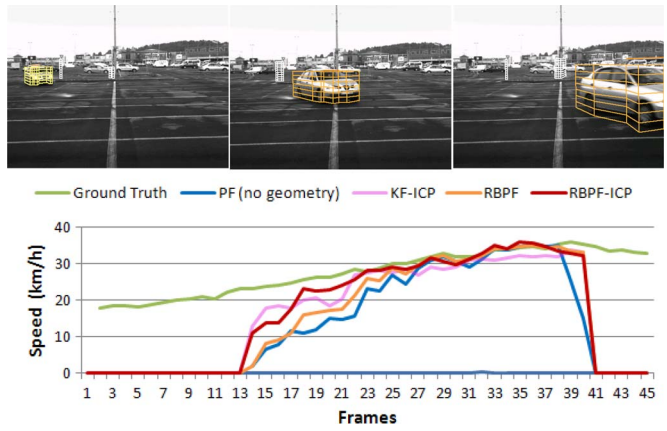

Fig. 16. Speed estimation in a controlled scenario.

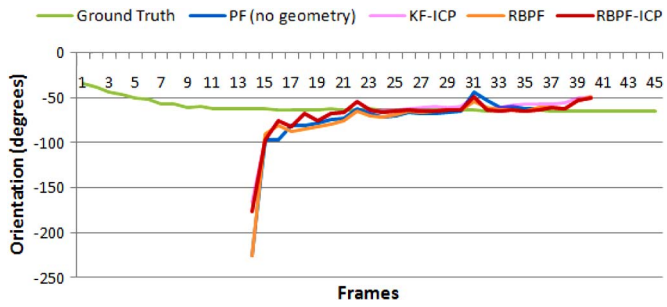

Fig. 17. Car orientation estimation in a controlled scenario.

provided by two high-accuracy Novatel GNSS receivers with RTK support [23]. We mounted one GNSS unit on the ego vehicle and one unit on the target car. The installed receivers were able to provide ground truth for the 3-D positioning and speed of the two cars with centimeter-level accuracy.

In Fig. 12, the target vehicle is represented by an oriented cuboid of fixed size. The box position and orientation are determined by using ground-truth measurements and converting them into the camera coordinate system.

For numerical evaluation, we included two types of scenarios: a real urban traffic scenario and a controlled situation. The first test case implies a sequence of an urban traffic scene, where the ego car follows a target car.

Figs. 13–15 show the results for a sample of 270 frames of city driving (at 10 frames per second), following the target

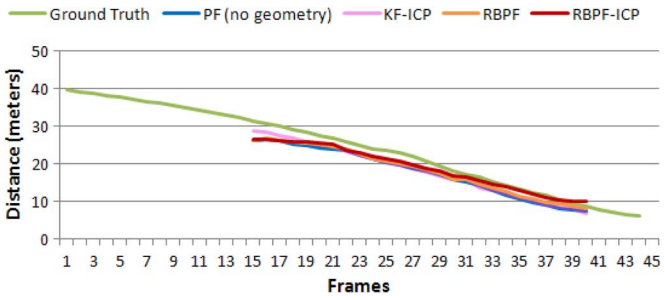

Fig. 18. Estimating the distance to the target. The target position is represented by its reference point.

TABLE I
SPEED ESTIMATION ACCURACY

Accuracy Metrics	PF (NO GEOMETRY)	KF-ICP	RBPF	RBPF-ICP
MAE (km/h)	3.42	2.25	1.43	1.33
STDEV (km/h)	5.07	1.06	1.06	0.90

TABLE II
ORIENTATION ESTIMATION ACCURACY

Accuracy Metrics	PF (NO GEOMETRY)	KF-ICP	RBPF	RBPF-ICP
MAE (deg)	5.33	4.07	3.16	1.72
STDEV (deg)	5.07	3.65	3.04	3.02

TABLE III
DISTANCE ESTIMATION ACCURACY

Accuracy Metrics	PF (NO GEOMETRY)	KF-ICP	RBPF	RBPF-ICP
MAE (km/h)	3.48	3.54	3.05	2.23
STDEV (km/h)	0.78	0.43	0.49	0.30

vehicle through several maneuvers, at a distance varying from 10 to 25 m. The target's speed ranged from 10 to 40 km/h, and the target's relative orientation to the coordinate system of the ego-vehicle changes as a left turn is executed.

The estimated speed is compared with the ground-truth speed, obtained from the high-accuracy GNSS device, as shown in Fig. 13. The mean absolute error of the speed estimation was found to be 1.85 km/h.

Fig. 14 shows the comparison between the estimated relative orientation of the target vehicle with respect to the ego vehicle's axis of elongation. The tracker correctly perceives the changes in orientation of the target, with a mean absolute error of 2.88°.

The comparison of the estimated distance between the target and the ego vehicle with the ground-truth distance (the Euclidean distance between the accurate GNSS points retrieved for the ego vehicle and the target), is shown in Fig. 15. The mean absolute error of distance estimation is 1.3 m, which includes the uncertainty related to the position of the estimated reference point for the target, as the true shape and size of the target is not known *a priori* by the tracker, and is continuously updated.

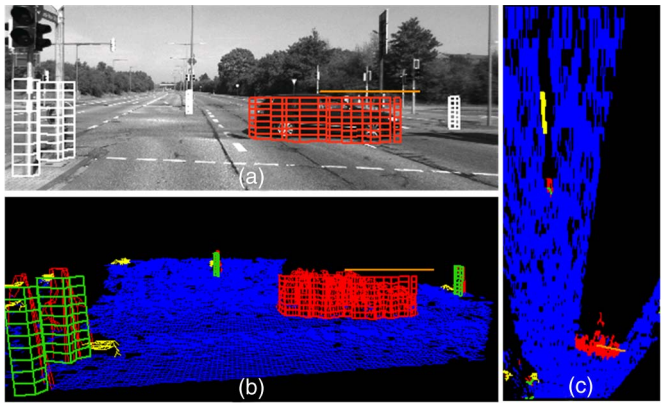

Fig. 19. Tracking results on KITTI data set. (a) The static and dynamic objects represented as free-form attributed polygonal models. The object speed vector is illustrated with yellow. The colors of the detected objects encode their speed, using the Middlebury convention (b) The 3-D Virtual view including the classified Elevation Map, the static (green) and dynamic (red) objects. (c) The results are projected on the ground plane.

The second experiment was conducted in a controlled scenario, where the target vehicle is moving in front of the ego-car and is in the field of view for only a short period of time. The target vehicle passes in front of the ego vehicle, from left to right, having an accelerated motion that varies its speed during the observation time from 20 to 30 km/h. We have evaluated the accuracy of speed, orientation, and distance estimation. Four tracking solutions are compared: a particle filter-based tracking solution that does not take into account the variable geometry of the perceived object (PF-no geometry); a Kalman Filter-based tracking technique, which uses ICP for motion extraction and providing the information of free-form delimiter position, speed, and its geometry (KF-ICP) [33]; the RBPF, including the variable geometry (RBPF); and the refined version of RBPF, the RBFP using ICP for speed initialization (RBPF-ICP).

The results of speed estimation are shown in Fig. 16. It is apparent that although all tracking solutions eventually converge to the correct speed estimation, the use of a variable geometry model helps the tracker converge faster, and the speed convergence is sped up even more by ICP initialization.

Fig. 17 shows the estimation of the target object's orientation, using the four tracking solutions. All the solutions rapidly converge to the correct orientation, the best performance being achieved again by the RBPF-ICP method.

Fig. 18 shows the evaluation of the distance measurement results relative to the ego vehicle. While the lag in estimation convergence is much lower than for speed and orientation, as the distance can be assessed directly from the stereo data, there are some initial systematic errors. The cause for these errors is that the target's reference point is initially estimated nearer to the ego vehicle than its true position, as the whole target is not yet fully observable. As the target vehicle gets closer and its true shape is estimated, the distance estimation gets significantly closer to the ground-truth data. As overall performance, the RBPF combined with ICP method again proves to be the best choice.

Tables I–III show the error assessment for the speed, orientation, and distance estimation using the four tracking methods.

TABLE IV
COMPARISON OF TRACKING APPROACHES TESTED ON KITTI DATA SET

	USED MODELS	FILTERING TECHNIQUE	STATE PARAMETERS	OBJECT SHAPE FLEXIBILITY
CUBOIDS [6]	3D Oriented boxes	Kalman Filter	Position and Speed	No
PF-GRID [28]	Dynamic Grid Cells	Particle Filter	Position and Speed	Yes
KF-ICP [33]	Attributed Polygonal Models	Kalman Filter	Position, Speed and Geometry	Yes
RBPF-ICP	Attributed Polygonal Models	Particle Filter and Kalman Filters	Position, Speed and Geometry	Yes

C. Numerical Evaluation Using a Public Data Set

Another set of experiments were performed using a publicly available and well-known benchmark database for driving assistance sensing and tracking applications, the KITTI data set compiled and maintained by the Karlsruhe Institute of Technology, and described in [34]. For our tests, we used the raw data sequences "2011_09_26_drive_0017" and "2011_09_26_drive_0018", from the "city" category. The sequences include rectified color and gray-scale image pairs suitable for stereo reconstruction, 3-D point clouds generated by a Velodyne laser scanner, Ego-Car 3-D GPS/IMU data, and object annotations, including position and occlusion status.

The rectified image pairs were processed using an SGM stereo reconstruction algorithm described in [35] and [38], and then the higher level detection and tracking algorithms were applied (see Fig. 19).

The following tracking algorithms were tested on this data set: cuboid-based tracking using Kalman Filter (CUBOIDS, [6]), particle-based occupancy grid tracking followed by cell grouping into individual objects (PF-GRID, [28]), Kalman Filter geometry tracking combined with ICP-based position tracking (KF-ICP, [33]), and the currently proposed RBPF-based tracking solution. A summary of the characteristics of each method is presented in Table IV.

The tests were aimed at assessing the accuracy of speed and orientation estimation for two types of objects, i.e., fully visible objects and partially visible (or partially occluded) objects. The results of speed accuracy estimation are shown in Table V, and the results of orientation accuracy estimation are shown in Table VI. From these results, one can see that the proposed method, RBPF-ICP, brings considerable improvement in accuracy, compared with the other methods, particularly in the case of partially visible objects.

D. Algorithm Complexity and Time Performance

The complexity of the algorithm is linear with the number of tracked objects, the number of used particles per object, and the number of control points per model. At particle level, the processing time is mostly dedicated to computing the particle's weight, which means testing the control points against the measurement data. Therefore, the particle time is linear with the number of control points. At object level, the processing time

TABLE V
COMPARISON OF SPEED ESTIMATION ACCURACY ON KITTI DATA SET

Method	FULLY VISIBLE OBJECTS		PARTIALLY VISIBLE OBJECTS	
	MAE (km/h)	STDEV (km/h)	MAE (km/h)	STDEV (km/h)
CUBOIDS [6]*	2.54	2.63	3.78	4.10
PF-GRID [28]	1.88	1.65	3.96	3.07
KF-ICP [33]*	2.59	1.18	3.55	2.56
RBPF-ICP*	1.90	1.31	2.45	2.55

* - methods working in real-time on the specified hardware configuration.

TABLE VI
COMPARISON OF ORIENTATION ESTIMATION ACCURACY ON KITTI DATA SET

Method	FULLY VISIBLE OBJECTS		PARTIALLY VISIBLE OBJECTS	
	MAE (deg)	STDEV (deg)	MAE (deg)	STDEV (deg)
CUBOIDS [6]*	5.77	4.94	9.12	5.70
PF-GRID [28]	4.57	4.70	8.95	1.84
KF-ICP [33]*	4.54	2.92	7.72	3.70
RBPF-ICP*	3.72	2.65	5.49	2.56

* - methods working in real-time on the specified hardware configuration.

of the measurement is the sum of the weighting time for each particle; thus, the object's processing time is linear with the number of particles. The same reasoning applies for the entire scene, composed of objects. The number of control points per particle is fixed, and the number of particles per object is also fixed, and therefore, the total running time is dependent only on the number of objects in the scene.

In our traffic scenario test sequences, the average number of tracked objects was 6. For each object we set up a fixed number of 80 particles and a fixed number of 20 control points per sample. The average processing time of the algorithm was about 99.83 ms/frame, or about 10 fps.

VI. CONCLUSION

In this paper, we have proposed a stereovision-based approach for tracking multiple objects in crowded urban traffic scenarios. The solution relies on measurement information provided by an intermediate occupancy grid and on free-form object delimiters extracted from this grid. In order to track visual appearance-based free-form obstacle representations, we adopted a particle filter-based mechanism, in which each particle state is described by two components, i.e., the object dynamic parameters (position and speed), and its estimated geometry (a set of key points). The high-dimensionality state-space problem is solved by using a Rao–Blackwellized solution, in which the obstacle dynamic properties are estimated by importance sampling, whereas the geometric properties are computed analytically by using a Kalman Filter for each key point. The presented probabilistic tracking approach takes into consideration the stereo uncertainties introduced by the sensorial system.

In order to evaluate the accuracy of the tracking method, we have performed two types of experiments. In the first case, the

ground-truth measurements were provided by high-accuracy GNSS receivers mounted on the ego vehicle and on the target vehicle. The second set of experiments was performed on the public raw data set. The proposed solution works in real time and, compared with the other methods, it was able to estimate with high accuracy the position, the speed and the geometry of objects from noisy stereo depth data.

As future work, we propose to improve the accuracy of our solution by including the intensity information, as in the optical flow techniques. We also intend to improve the system processing time by further optimizations.

REFERENCES

[1] 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.

[2] A. Petrovskaya and S. Thrun, "Model based vehicle detection and tracking for autonomous urban driving," *Auton. Robots*, vol. 26, no. 2–3, pp. 123–139, 2009.

[3] A. Vatavu and S. Nedevschi, "Real-time modeling of dynamic environments in traffic scenarios using a stereo-vision system," in *Proc. IEEE ITSC*, Sep. 16–19, 2012, pp. 722–727.

[4] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 5, pp. 694–711, May 2006.

[5] U. Franke, C. Rabe, H. Badino, and S. Gehrig, "6d-vision: Fusion of stereo and motion for robust environment perception," in *Proc. 27th Annu. Meet. German Assoc. Pattern Recog. DAGM*, 2005, pp. 216–223.

[6] R. Danescu, S. Nedevschi, M. M. Meinecke, and T. Graf, "Stereovision based vehicle tracking in urban traffic environments," in *Proc. IEEE ITSC*, Seattle, USA, 2007, pp. 400–404.

[7] C. C. Wang, C. Thorpe, M. Hebert, S. Thrun, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking," *Int. J. Robot. Res.*, vol. 26, no. 9, pp. 889–916, Sep. 2007.

[8] A. Vatavu, R. Danescu, and S. Nedevschi, "Real-time dynamic environment perception in driving scenarios using difference fronts," in *Proc. IEEE IV*, Jun. 3–7, 2012, pp. 717–722.

[9] A. Vatavu, S. Nedevschi, and F. Oniga, "Real time object delimiters extraction for environment representation in driving scenarios," in *Proc. ICINCO-RA*, Milano, Italy, 2009, pp. 86–93.

[10] A. Azim and O. Aycard, "Detection, classification and tracking of moving objects in a 3D environment," in *Proc. IEEE IV*, Jun. 3–7, 2012, pp. 802–807.

[11] Q. Chen, Q.-S. Sun, P.-A. Heng, and D.-S. Xia, "Two-stage object tracking method based on kernel and active contour," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 4, pp. 605–609, 2010.

[12] F. Oniga and S. Nedevschi, "Processing dense stereo data using elevation maps: Road surface, traffic isle, and obstacle detection," *IEEE Trans. Veh. Technol.*, vol. 59, no. 3, pp. 1172–1182, Mar. 2010.

[13] A. Elfes, "A sonar-based mapping and navigation system," in *Proc. IEEE ICRA*, Apr. 1986, pp. 1151–1156.

[14] R. Danescu, F. Oniga, S. Nedevschi, and M.-M. Meinecke, "Tracking multiple objects using particle filters and digital elevation maps," in *Proc. IEEE-IV*, Xi'an, China, Jun. 2009, pp. 88–93.

[15] D. Pfeiffer and U. Franke, "Modeling dynamic 3D environments by means of the stixel world," *IEEE Intell. Transp. Syst. Mag.*, vol. 3, no. 3, pp. 24, 36, 2011.

[16] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. Yezzi, "Particle filtering for geometric active contours with application to tracking moving and deforming objects," in *Proc. IEEE CVPR*, Jun. 20–25, 2005, vol. 2, pp. 2–9.

[17] J. D. Jackson, A. J. Yezzi, and S. Soatto, "Tracking deformable moving objects under severe occlusions," in *Proc. 43rd IEEE CDC*, 2004, vol. 3, pp. 2990–2995.

[18] M. Isard and A. Blake, "Condensation, 'Conditional density propagation for visual tracking'," *Int. J. Comput. Vis.*, vol. 29, no. 1, pp. 5–28, 1998.

[19] R. Merwe, A. Doucet, N. Freitas, and E. Wan, "The unscented particle filter," *Adv. Neural Inf. Process. Syst.*, vol. 8, pp. 351–357, 2000.

[20] A. Doucet, N. De Freitas, K. Murphy, and S. Russell, "Rao-Blackwellised particle filtering for dynamic Bayesian networks," in *Proc. 16th Conf. Uncertainty Artif. Intell.*, 2000, pp. 176–183.

[21] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. AAAI Nat. Conf. Artif. Intell.*, 2002, pp. 593–598.

[22] S. Baker *et al.*, "A database and evaluation methodology for optical flow," in *Proc ICCV*, 2011, pp. 1–8.

[23] NovaTel Inc., High Precision GPS & GNSS Receivers. Accessed 2013 December 11. [Online]. Available: http://www.novatel.com/products/gnss-receivers/

[24] N. Fairfield, G. A. Kantor, and D. Wettergreen, "Real-time SLAM with octree evidence grids for exploration in underwater tunnels," *J. Field Robot.*, vol. 24, no. 1–2, pp. 3–21, 2007.

[25] D. Simon, *Optimal State Estimation*. New York, NY, USA: Wiley, 2006.

[26] J. I. Woodill, G. Gordon, and R. Buck, "TYZX deepsea high speed stereo vision system," in *Proc. IEEE Comput. Soc. Workshop Real Time 3-D Sens. Their Use*, Washington, DC, USA, 2004.

[27] F. Pletzer *et al.*, "Feature-based level of service classification for traffic surveillance," in *Proc. IEEE ITSC*, Oct. 5–7, 2011, pp. 1015–1020.

[28] R. Danescu, F. Oniga, and S. Nedevschi, "Modeling and tracking the driving environment with a particle-based occupancy grid," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1331–1342, Dec. 2011.

[29] S. Sivaraman and M. Trivedi, "A general active-learning framework for on-road vehicle recognition and tracking," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 267–276, Jun. 2010.

[30] H. Tehrani Niknejad, A. Takeuchi, S. Mita, and D. McAllester, "On-road multivehicle tracking using deformable object model and particle filter with improved likelihood estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 748–758, Jun. 2012.

[31] M. Perrollaz, J.-D. Yoder, A. Nègre, A. Spalanzani, and C. Laugier, "A visibility-based approach for occupancy grid computation in disparity space," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1383–1393, Sep. 2012.

[32] F. Erbs, A. Barth, and U. Franke, "Moving vehicle detection by optimal segmentation of the dynamic stixel world," in *Proc. of IEEE IV*, Jun. 2011, pp. 951–956.

[33] A. Vatavu and S. Nedevschi, "Vision-based tracking of multiple objects in dynamic unstructured environments using free-form obstacle delimiters," in *Proc. ECMR*, Barcelona, Spain, Sep. 25–27, 2013, pp. 367–372.

[34] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[35] C. Pantilie and S. Nedevschi, "SORT-SGM: Sub-pixel optimized real-time semi-global matching for intelligent vehicles," *IEEE Trans. Veh. Technol.*, vol. 61, no. 3, pp. 1032–1042, 2012.

[36] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: MIT Press, 2005.

[37] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*. Cambridge, MA: MIT Press, 1993.

[38] I. Haller and S. Nedevschi, "Design of interpolation functions for subpixel accuracy stereo-vision systems," *IEEE Trans. Image Process.*, vol. 21, no. 2, pp. 889–898, 2012.

[39] R. Danescu and S. Nedevschi, "A particle-based solution for modeling and tracking dynamic digital elevation maps," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 3, pp. 1002–1015, Jun. 2014.

[40] R. Danescu and S. Nedevschi, "A flexible solution for modeling and tracking generic dynamic 3D environments," in *Proc. IEEE-ITSC*, The Hague, The Netherlands, Oct. 2013, pp. 1686–1692.

[41] A. Vatavu, R. Danescu, and S. Nedevschi, "Tracking multiple objects in traffic scenarios using free-form obstacle delimiters and particle filters," in *Proc. 16th IEEE ITSC*, The Hague, The Netherlands, Oct. 2013, pp. 1346–1351.

[42] A. Vatavu and S. Nedevschi, "Modeling unstructured environments with dynamic persistence grids and object delimiters in urban traffic scenarios," in *Proc. IEEE IV*, Gold Coast, Australia, Jun. 23–26, 2013, pp. 505–510.

[43] H. Hirschmüller, "Accurate and efficient stereo processing by semi global matching and mutual information," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog.*, 2005, vol. 2, pp. 807–814.

Andrei Vatavu (M'12) received the M.S. degree in computer science from Technical University of Cluj-Napoca, Cluj-Napoca, Romania, in 2008, where he is currently working toward the Ph.D. degree in computer science, specializing in stereovision systems for intelligent vehicles.

He is a Ph.D. Student with the Faculty of Automation and Computer Science, Technical University of Cluj-Napoca. His research interests include stereovision-based environment representation, occupancy grid analysis, and object tracking.

Radu Danescu (M'11) received the Diploma Engineer, M.S., and Ph.D. degrees in computer science from Technical University of Cluj-Napoca (TUCN), Cluj-Napoca, Romania, in 2002, 2003, and 2009, respectively.

He is an Associate Professor with the Computer Science Department, TUCN, teaching image processing, pattern recognition, and design with microprocessors. His main research interests include stereovision- and probability-based tracking, with applications in driving assistance. He is a member of the Image Processing and Pattern Recognition Research Laboratory, TUCN.

Sergiu Nedevschi (M'99) received the M.S. and Ph.D. degrees in electrical engineering from Technical University of Cluj-Napoca (TUCN), Cluj-Napoca, Romania, in 1975 and 1993, respectively.

From 1976 to 1983 he was with the Research Institute for Computer Technologies, Cluj-Napoca, as a Researcher. In 1998 he was appointed Professor in computer science and founded the Image Processing and Pattern Recognition Research Laboratory, TUCN. From 2000 to 2004 he was the Head of the Computer Science Department, TUCN, and from 2004 to 2012, the Dean of the Faculty of Automation and Computer Science. He is currently Vice President in charge of scientific research with TUCN.

Prof. Nedevschi has published hundreds of scientific papers and has edited multiple volumes, including books and conference proceedings. His research interests include image processing, pattern recognition, computer vision, intelligent vehicles, signal processing, and computer architecture.