Vision-based Tracking of Multiple Objects in Dynamic Unstructured Environments using Free-Form Obstacle Delimiters

Andrei Vatavu and Sergiu Nedevschi, *Member, IEEE*

***Abstract*—Modeling and tracking of dynamic objects is a challenging research problem in the field of driving assistance systems. Typically, the environment to be tracked is heterogeneous and unstructured. As a consequence, the tracking system must deal with measurement uncertainties, occlusions or deformable objects. In this paper we propose a real-time object tracking solution for dynamic unstructured environments. This method relies on stereo vision-based 3D information that is mapped into an intermediate digital elevation map. We apply a recursive Bayesian approach for estimating both the obstacle dynamic parameters and its geometry. In order to compute the obstacle motion we use an Iterative Closest Points-based registration technique that takes into consideration the stereo uncertainties. In our case, the object model is represented by a reference point and *N* delimiter landmarks. For each target we apply a Kalman filter in order to track the obstacle position and speed. In addition, the object geometry is updated by using an independent 2x2 Kalman filter for each delimiter landmark. The proposed method works in real-time and takes into consideration the stereo uncertainties.**

I. Introduction

Modeling and tracking of dynamic obstacles is an essential research topic in the field of driving assistance systems. Usually, the tracking module relies on extracting a set of key features from the scene and estimating their state over time. Existing research solutions use vision [5][6], ultrasound [4] or laser sensors [2][3]. Despite the simplicity of the main idea, the modeling and tracking of the dynamic objects remains a challenging task. Typically, the environment to be tracked is heterogeneous and unstructured and the tracking system must deal with measurement uncertainties, occlusions, deformable objects or unpredictable behavior of the traffic entities. In such complex situations, a driving assistance system must be able to represent and track multiple obstacles in real-time, with high confidence and high accuracy.

The motion estimation approaches can be applied at different representation levels. Some of the existing techniques imply directly tracking 3D point clouds [5], while other solutions try to minimize the computational cost by using intermediate representations. The 3D information is transformed into occupancy grids [13], octrees [4][10], Stixel maps [15], voxels [10] or digital elevation maps [14]. Most of the tracking methods use high level attributes such as object contours [11][12], difference fronts [8], polygonal models [7], 2D boxes or 3D cuboids [2][6]. Usually, the existing approaches work well in structured environments, where the obstacle's geometry is known. Typically, the traffic entities are represented by simplistic models such as bounding boxes and the obstacle's position is determined by its center of mass. However it is difficult to use constrained models when the surrounding world is unstructured. The target pose estimation may be affected by occlusions or by changes in its geometry and the tracking mechanism may lead to erroneous results. In order to overcome this problem, various algorithms for moving and deforming objects are proposed [16][17]. Most often, the model shape is represented implicitly [17], or by a set of fixed number of points. For example, the authors in [17] present a tracking solution for slowly deforming and moving contours that are represented implicitly. Isard and Blake [18] introduce the CONDENSATION algorithm for tracking parametric curves. In [23], the authors propose a real-time tracking approach by using level sets. Usually, the obstacle tracking techniques rely on variants of Kalman filters [6], particle filters [16][18] or hybrid solutions [2][20][21]. In [6], a Kalman filter is used to track objects that are modeled as cuboids. In [2], a Rao-Blackwellized Particle Filter is applied to estimate both dynamic and geometric properties of the tracked cars. For simplicity the vehicle shape is approximated by a rectangle. The classical Kalman filter based estimators are suitable for linear systems in which the posterior distribution is modeled by a Gaussian function. As an alternative, the particle filter based methods approximate the state space by a collection of N discrete samples. Their advantage is the ability to handle non-linear systems and multi-modal distributions. However, particle filters are not suitable for high-dimensional states as their computational complexity tends to grow exponentially with the number of state parameters.

The existing tracking methods also differ by the way the correct correspondences are computed. Direct scan matching approaches such as Iterative Closest Points (ICP) algorithm [1] are used for aligning new measurements to the existing models. However, in the case of laser-based systems, the object registration in subsequent scans is hard to be achieved when the traffic objects or the ego-car moves at high speeds or when the measurement uncertainties are not taken into consideration.

In this paper we propose a real-time object tracking solution for dynamic unstructured environments. Our method relies on stereo vision-based 3D information that is mapped into an intermediate digital elevation map. The resulted grid of elevations is used to perform the data association and to extract a set of free-form object delimiters. We propose a recursive Bayesian approach for estimating both the obstacle

Andrei Vatavu and Sergiu Nedevschi are with the Technical University of Cluj-Napoca, Computer Science Department (e-mail: Andrei.Vatavu@cs.utcluj.ro, Sergiu.Nedevschi@cs.utcluj.ro). Address: Computer Science Department, Str. Memorandumului Nr. 28, Cluj-Napoca, Romania. Phone: +40 264 401484.

dynamic parameters and its geometry. In order to compute the obstacle motion we apply an Iterative Closest Points-based alignment between the extracted delimiters and the associated trackers. Unlike other solutions where objects are represented by fixed templates, we have adopted an approach in which we consider that the obstacle shape may gradually change over time. In our case, the object model is represented by a reference point and *N* delimiter landmarks (control points). For each target we apply a Kalman filter in order to track the obstacle position and speed. In addition, the object geometry is updated by using an independent 2x2 Kalman filter for each delimiter landmark. The proposed method works in real-time and takes into consideration the stereo uncertainties.

The paper is structured as follows: the next section presents the system architecture, the preprocessing stage is described in chapter III, the object model is presented in chapter IV while the proposed obstacle tracking algorithm is detailed in chapter V. The last two sections present the experimental results and conclude this work.

II. System Overview

The system architecture (see Fig. 1) comprises of three main components: Intermediate Representation, Preprocessing and Tracking.

The first component acquires image pairs from the two cameras. Next, stereo reconstruction is applied with a dedicated TYZX board [22]. The raw dense stereo information is then used to compute an intermediate classified grid [14]. Each grid cell is labeled based on its height value as: obstacle, traffic isle or road (see Fig. 2.b and c).

The Preprocessing component performs a set of tasks prior to object tracking stage. In this step, the resulted classified grid is used to assign new measurements to the existing trackers and to extract obstacle delimiters.

The Tracking level consists in estimating the optimal state parameters. For each existing individual tracker, the following processing steps are applied: prediction, motion estimation, updating the object dynamic parameters (position and speed) and updating the object geometry. We will detail each of these steps in the following sections.

III. Preprocessing

The Preprocessing module performs a set of tasks prior to object tracking stage.

A. Data Association

The data association step assigns new observations to the existing targets. In our case we perform the data association by computing overlapping scores between the occupancy grid blobs at consecutive time steps. We define a blob *A* in the current frame as a collection of connected cells that are occupied:

$$A = \{a_i \mid Occ(a_i) = true, i = [1..N_A]\} \quad (1)$$

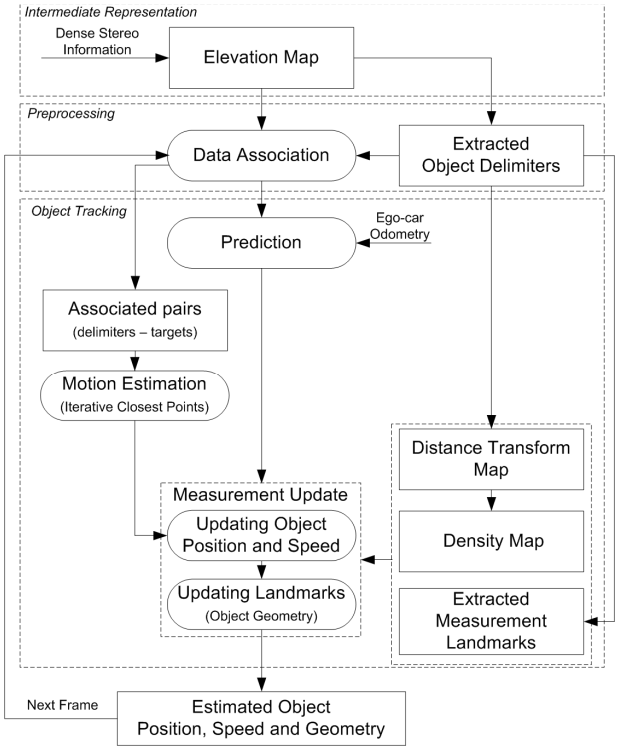

Figure 1. System Overview

where $Occ(a_i)$ is *true* when a cell a_i is occupied and *false* when not. Similarly, a certain blob from the previous frame is described as:

$$B = \{b_i \mid Occ(b_i) = true, i = [1..N_B]\} \quad (2)$$

We denote with $O(a_i, b_j)$ the overlap function between two points from *A* and *B*. This function is 1 when the two occupied cells overlap and 0 otherwise. For each blob from the previous frame *A* and each blob from the current frame *B* we compute an overlapping score:

$$w_{AB} = A \cap B = \sum_{i=1}^{N_A} \sum_{j=1}^{N_B} O(a_i, b_j) \quad (3)$$

As the result, a score matrix $W = \{w_{ij}\}$ is formed. In order to consider the cases when larger blobs are split into many disjoint parts and vice versa, we define two cases: the most likely association from *A* to *B* (forward association):

$$Assoc(A) = \arg\max_B P(B \mid A) = \arg\max_B \frac{w_{AB}}{N_A} \quad (4)$$

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

$$Assoc(B) = \arg\max_A P(A \mid B) = \arg\max_A \frac{w_{AB}}{N_B} \quad (5)$$

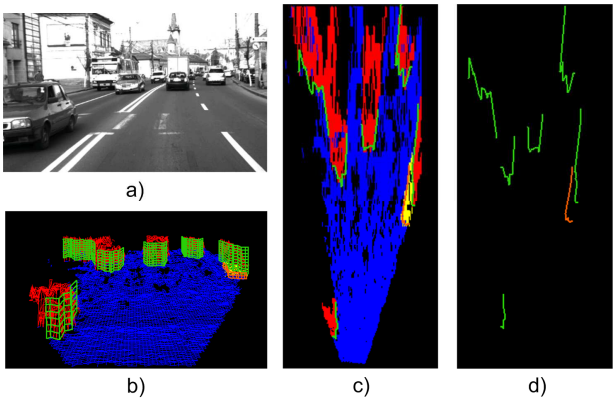

Figure 2. a) Left camera image b) A virtual view with the resulted grid of elevations and the extracted delimiters. The grid cells are classified as road (blue), obstacle (red) and traffic isle (yellow). c) The elevation map is projected on the ground plane. d) The top view of the extracted delimiters. The obstacle delimiters are represented with green while the traffic isle delimiters are colored with orange.

B. *Extracting Object Delimiters*

At each time step, the object delimiters are extracted from the occupancy grid by using the *Border Scanner* algorithm that is described in [9]. The main idea of the *Border Scanner* approach is to generate an object contour C_{object} by selecting the most visible object cells c_i. This is achieved by considering a virtual ray which extends from the observation point and moves in a radial direction with fixed increments. At each step the closest occupied grid cell $Occ(c_i) = true$ is chosen as the delimiter point (see Fig. 2.d).

$$C_{object} = \{c_i \mid Occ(c_i) = true, i \in [1..M_c]\} \tag{6}$$

IV. Object Model

In order to deal with deformable obstacles we consider that the object shape may gradually change over time due to factors such as the dynamic nature of the environment, noisy measurements or occlusions.

In our solution, an object is described by the following parameters (see Fig. 3):

- A local reference point P_{ref} denoting the obstacle position in the camera coordinate system. The reference point is initially set to the object center of mass, and is subsequently estimated by the tracking mechanism.
- The object speed vector $\vec{V}(v_x, v_z)$
- A set of landmarks $K = \{P_c^i(x_c^i, z_c^i) \mid i = [1..N_c]\}$ specifying the object shape, and defining the vertices of a polygonal line. Initially, the model landmarks (control points) are set by selecting *N* equidistant points along the object contour. Then, these landmarks are updated as new observations become available. For each landmark

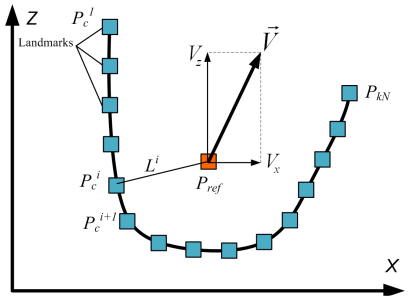

Figure 3. Object Model. An object is represented by a reference point P_{ref}, and a set of *N* landmarks P_c^i.

$P_c^i(x_c^i, z_c^i)$ we also compute its relative position $L^i(l_x^i, l_z^i)$ to the reference point $P_{ref}(x_{ref}, z_{ref})$:

$$\begin{bmatrix} l_x^i \\ l_z^i \end{bmatrix} = \begin{bmatrix} x_c^i - x_{ref} \\ z_c^i - z_{ref} \end{bmatrix} \tag{7}$$

In our case the coordinate system has its origin in front of the ego-vehicle. The *X* axis points to the right while the *Z* axis points towards the ego-car direction. Considering the notations above, at a time *t* the obstacle state is defined as:

$$S_t = [x_{ref}, z_{ref}, v_x, v_z, l_x^1, l_z^1, l_x^2, l_z^2, ..., l_x^N, l_z^N]^T \tag{8}$$

V. Object Tracking

From a Bayesian perspective, the tracking problem consists in estimating the current object state S_t, from a set of noisy measurements $Z_{1:t} = \{Z_1, Z_2 ..., Z_t\}$ up to the time *t*:

$$p(S_t \mid Z_1, Z_2 ... Z_t) = \eta p(Z_t \mid S_t) \int_{S_{t-1}} p(S_t \mid S_{t-1}) p(S_{t-1} \mid Z_1, ..., Z_{t-1}) \tag{9}$$

where η represents a normalization constant. The $p(Z_t \mid S_t)$ term describes the *measurement model* at a time *t* and $p(S_t \mid S_{t-1})$ denotes the *state transition model* (motion model) from S_{t-1} to S_t.

In our approach, we split the obstacle state S_t into two parts:

$$S_t = [X_t, G_t]^T \tag{10}$$

where the first component $X_t = [x_{ref}, z_{ref}, v_x, v_z]^T$ denotes the object position and speed while the component $G_t = [l_x^1, l_z^1, l_x^2, l_z^2, ..., l_x^N, l_z^N]^T$ defines the obstacle geometry. The overall posterior distribution $p(X_t, G_t \mid Z_{1:t})$ defined by (9) is decomposed as:

$$p(X_t, G_t \mid Z_{1:t}) = p(X_t \mid Z_{1:t}) p(G_t \mid X_t, Z_{1:t}) \tag{11}$$

The first distribution $p(X_t \mid Z_{1:t})$ describes the object dynamic parameters, while the second term $p(G_t \mid X_t, Z_{1:t})$ denotes the object geometry posterior conditioned on X_t.

Both terms are estimated analytically by using Kalman filters. Therefore, we first estimate the obstacle position and speed. Then, each landmark in G_t being described by a mean value and a covariance matrix $(\hat{L}^j, \sum^j)$, is updated by using a 2x2 Kalman filter. In the next sections we detail the main steps of our tracking solution.

A. Prediction

This step predicts the current state S_t at time *t* given the previous information S_{t-1} and the motion model $p(S_t \mid S_{t-1})$. First, based on the ego-car motion we extract its velocity from the independent dynamics of the tracked objects. In our case, we obtain the vehicle speed *v* and the yaw rate $\dot{\psi}$ from the car sensors. By following the ego-vehicle motion model with constant yaw rate and constant speed, the object position (x_{ref}, z_{ref}) is transformed according to:

$$\begin{bmatrix} x_{ref\_c} \\ z_{ref\_c} \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} x_{ref} \\ z_{ref} \end{bmatrix} - \begin{bmatrix} \frac{v\Delta t}{\psi}(1-\cos\psi) \\ \frac{v\Delta t}{\psi}\sin\psi \end{bmatrix} \quad (12)$$

where Δt is the time delay between two frames and $\psi = \dot{\psi}\Delta t$ denotes the ego-vehicle rotation angle around the *Y* axis. The position and velocity $X_t = [x_{ref}, z_{ref}, v_x, v_z]^T$ of each tracked object is predicted according to:

$$\begin{bmatrix} x_{ref} \\ z_{ref} \\ v_x \\ v_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{ref\_c} \\ z_{ref\_c} \\ v_x \\ v_z \end{bmatrix} + w \quad (13)$$

In addition, each landmark $L^i = \left[l_x^i, l_z^i\right]^T$ is described by the following state transition equation:

$$\begin{bmatrix} l_x^i \\ l_z^i \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} l_x^i \\ l_z^i \end{bmatrix} + v \quad (14)$$

where $w \sim N(0, Q_d)$ and $v \sim N(0, Q_g)$ are Gaussian random noises with zero mean and known covariances Q_d and Q_g. The matrix Q_d is determined based on a certain covariance for the obstacle acceleration. The covariance Q_g is estimated by considering that the obstacle geometry may gradually change over time.

B. Motion Estimation

To estimate the motion we perform a fast pairwise alignment of the extracted delimiters in the current frame to the existing individual trackers (represented by a set of landmarks). One of the most used approaches for aligning two point sets in a common coordinate system is the Iterative Closest Point (ICP) algorithm. First introduced by Besl and McKay [1] this method is typically used in scan matching approaches. A detailed description of our ICP-based motion estimation algorithm is given in [19]. The main idea is to find an optimal rotation *R* and translation *T* by minimizing the alignment error between the extracted delimiters and the associated object models. Additionally, this approach includes the stereo uncertainties by assigning a weight w_i to each point-to-point correspondence pair. We denote by $P=\{p_1,p_2,..., p_M\}$ a model set that describes the target, and by $Q=\{q_1,q_2, ..., q_K\}$ a data set containing the points of a measurement contour. Each point q_j from *Q* is paired with the closest landmark p_i from *P*. Thus, our objective function can be defined as:

$$\varepsilon(R,T) = \sum_{i=1}^{N} \left\| w_i (Rp_i + T - q_i) \right\|^2 \quad (15)$$

In order to converge to a local minimum the following main steps are repeated:

1. Matching: For each measurement point q_i from *Q* the closest landmark point from *P* is found:

$$d(q_i, P) = \min_{j \in \{1..N_p\}} d(q_i, p_j) \quad (16)$$

First, we define a region of interest that covers both the object model and the extracted delimiter $C_{objectt}$. Then we compute a modified Distance Transform (see Fig. 4.c). Each cell $m_{dm}(x_{dm}, z_{dm})$ from the distance map will be described by two values: a distance d_m^2 to the nearest delimiter point $c_j(x_{del}, z_{del})$, and a position of the respective correspondence (x_{del}, z_{del}). The probability density map (see Fig. 4.d) can be determined now for each cell m_{dm} by converting the distance values according to:

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

where σ_x and σ_z represent the stereo uncertainties of the corresponding measurement point. If we consider that the stereo-vision system is rectified, then for each grid cell the depth error σ_z and the lateral error σ_x can be approximated as:

$$\sigma_z = \frac{z^2 \cdot \sigma_d}{b \cdot f}, \quad \sigma_x = \frac{\sigma_z \cdot x}{z} \quad (18)$$

where *x* and *z* are the real world coordinates of a point, σ_d represents the disparity error, *b* is the stereo system baseline and *f* denotes the focal length. In order to make use of information provided by the stereo uncertainties we assign a density value w_i to each corresponding pair (q_i, p_i).

2. Outliers Rejection*:* Two strategies are used for filtering the erroneous correspondences: removing the pairs with a large point-to-point distance and rejecting many-to-one correspondences.

3. Error Minimization: In this step, an optimal rotation *R* and translation *T* are computed by minimizing the objective function described in (15). As equation (15) represents a

least-square optimization problem, we estimated the unknown coefficients by setting the partial derivatives to zero and solving the resulted system of equation.

4. Updating: The total transformation matrix M_g and the position of the model contour are updated with the computed R and T.

5. Testing the convergence: Test if the algorithm has been achieved a minimum error by calculating the average point-to-point alignment error according to:

$$D_{error} = \sum_{i=1}^{N} \frac{w_i \cdot d(q_i, p_i)}{\sum_{k=1}^{N} w_k} \qquad (19)$$

The algorithm stops when the change in the error is below a given threshold or when a maximum iteration number is reached.

C. State Update

This step consists in updating the obstacle dynamic parameters X_t and its geometry G_t. First, the new object position and speed are estimated by using the predicted state (13) and the new motion information computed in the previous step. Next, the geometry component G_t is updated as follows: for each landmark we use an independent 2x2 Kalman filter to estimate its state $\hat{L}^i = [l_x^i, l_z^i]^T$ and covariance $\sum_L^i$. The new observations are generated by selecting N equidistant points along the extracted delimiter in the current frame. We use the same error model (18) to compute measurement covariance matrices for both object dynamics X_t and its geometry G_t (one covariance per landmark).

VI. Experimental Results

We tested our system in multiple urban traffic scenarios. The experiments were performed on an Intel Core 2 Duo E6750 CPU at 2.66GHz and 4GB of RAM. The elevation map used in our approach has a resolution of 240 x 500 cells.

Fig. 5. illustrates how the obstacle geometry is inferred (green) after applying ICP-based motion estimation and landmark filtering. The model delimiter (extracted in the previous frame) is colored with red while the measurement delimiter (extracted in the current frame) is represented with blue.

Fig. 6. Shows, for an individual target, the estimated speeds with the proposed tracking approach.

Fig. 7. presents some examples in real traffic situation (left) and the virtual view of the scene (right). The object speed is encoded in the hue and saturation. The virtual scene shows how the obstacles are classified as static (green) and dynamic (red).

The objects with a speed greater than 9km/h are considered dynamic. The processing time of our solution is dependent on the number of tracked obstacles and object landmarks.

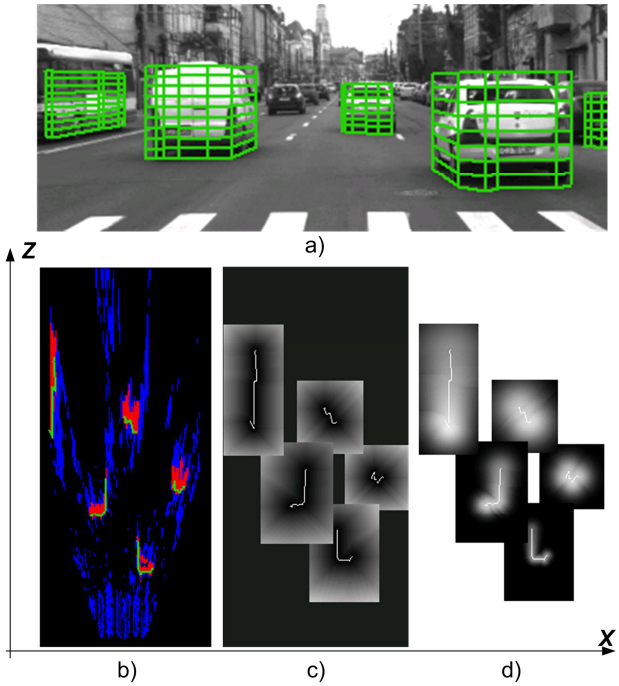

Figure 4. a) A traffic scenario with the extracted object delimiters (green). b) The classified elevation grid (top view) and the projected obstacle delimiters. c) The Distance Transform. d) The density map, generated for each object. High intensities means high measurement probability.

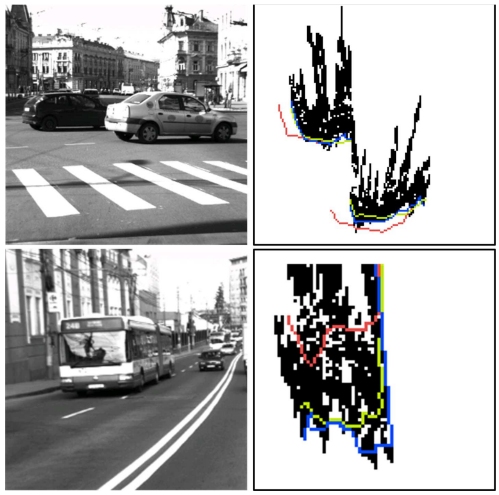

Figure 5. The obstacle geometry. Left: examples of traffic scenes. Right: The tracked object model (Top view). The model delimiters are colored with red. Current measurements are drawn with blue. The filtered geometry is represented with green color.

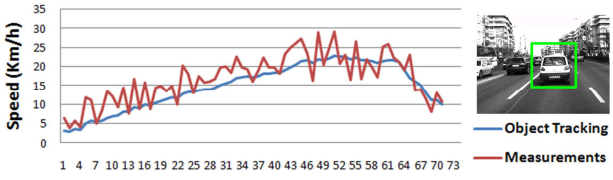

Figure 6. Estimated Speeds. Object tracking (blue) vs Measurements (red). The measurements are obtained using only the ICP-based motion estimation.

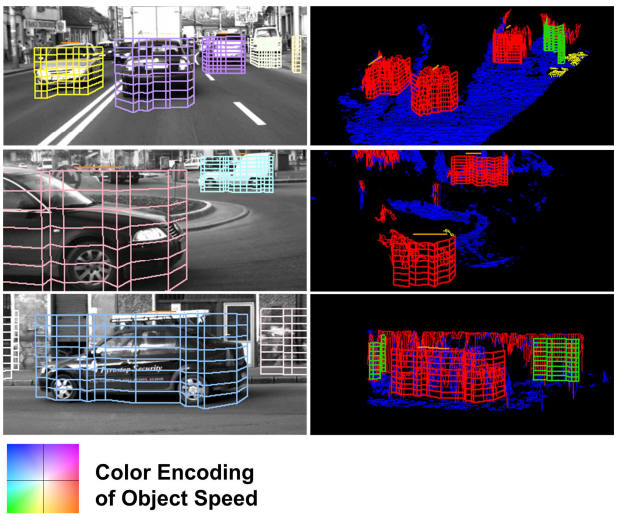

Figure 7. Multiple Object Tracking. Left: real traffic scenarios. Speeds are color encoded. Right: The virtual view of the scene with dynamic (red) and static (green) objects. The color hue describes the orientation of a moving obstacle, while the color saturation encodes its speed

The average number of tracked obstacles in our experiments was 5. In addition, to describe each target we fixed the number of landmarks to 100. In these conditions the average running time required by the tracking algorithm was 61.45 ms per frame. In order to estimate the accuracy of our method we used a "follow the leader" scenario, consisting of a single moving target in front of the ego-car with known speed. The resulted mean absolute error (MAE) was 5.23Km/h.

VII. Conclusions

In this paper we presented a real-time method for multiple-object tracking in unstructured environments. Our solution is based on information provided by a stereovision-generated elevation map. We propose a recursive Bayesian approach for estimating both the obstacle dynamic parameters and its geometry. In order to compute the obstacle motion we apply an Iterative Closest Points-based alignment between the extracted delimiters and the associated trackers. Unlike the other existing techniques where objects are represented by fixed templates, we have adopted an approach in which we consider that the obstacle shape may gradually change over time. In our case, the object model is represented by a reference point and *N* delimiter landmarks (control points). For each target we apply a Kalman filter in order to track its position and speed. In addition, the object geometry is updated by using an independent 2x2 Kalman filter for each delimiter landmark. The proposed method works in real-time and takes into consideration the stereo uncertainties. As future work we propose to improve the system accuracy by integrating measurements provided by other motion estimation techniques such as optical flow.

References

[1] P. Besl and N. McKay, "A method for registration of 3d shape", in *Trans. Pattern Analysis and Machine Intelligence*, 12(2), 1992.

[2] A. Petrovskaya and S. Thrun, "Model based vehicle tracking for autonomous driving in urban environments". In *RSS*, 2008.

[3] L. Spinello, R. Triebel, and R. Siegwart, "Multiclass multimodal detection and tracking in urban environments", in *IJRR*, 29.12, pp 1498-1515, 2010.

[4] N. Fairfield, G. A. Kantor, and D. Wettergreen, "Real-Time SLAM with Octree Evidence Grids for Exploration in Underwater Tunnels", in *Journal of Field Robotics*, 2007

[5] U. Franke, C. Rabe, H. Badino, and S. Gehrig, "6d-vision: Fusion of stereo and motion for robust environment perception," in *DAGM '05*, 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. of the IEEE ITSC 2007*, Seattle, USA, 2007

[7] C.C. Wang, C. Thorpe, M. Hebert, S. Thrun, and H. Durrant-Whyte, "Simultaneous localization, mapping and moving object tracking", in *The International Journal of Robotics Research*, 26(6), June 2007.

[8] A. Vatavu, R. Danescu, S. Nedevschi, "Real-time dynamic environment perception in driving scenarios using difference fronts", in *Proc of IEEE Intelligent Vehicles Symposium (IV 2012)*, 2012, pp.717-722.

[9] A. Vatavu, Sergiu Nedevschi, and Florin Oniga, "Real Time Object Delimiters Extraction for Environment Representation in Driving Scenarios", In *Proc. of ICINCO-RA 2009*, Milano, Italy, 2009, pp 86-93.

[10] A. Azim, O. Aycard, "Detection, classification and tracking of moving objects in a 3D environment," in *Proc. of IEEE Intelligent Vehicles Symposium (IV)*, 3-7 June 2012, pp.802-807

[11] S. Prakash, S. Thomas, "Contour tracking with condensation / stochastic search", In *Dept. of CSE. IIT* Kanpur, 2007.

[12] M. Yokoyama and T. Poggio, "A Contour-Based Moving Object Detection and Tracking", in *Proc. of the IEEE 14th International Conference on Computer Communications and Networks (ICCCN '05)*, Washington, DC, USA, 2005, pp. 271-276.

[13] A. Elfes, "A Sonar-Based Mapping and Navigation System", in *Proc. of IEEE ICRA*, April 1986, pp. 1151-1156

[14] F. Oniga and S. Nedevschi, "Processing Dense Stereo Data Using Elevation Maps: Road Surface, Traffic Isle, and Obstacle Detection", in *IEEE Transactions on Vehicular Technology*, Vol. 59, No. 3, March 2010, pp. 1172-1182.

[15] D. Pfeiffer and U. Franke, "Efficient Representation of Traffic Scenes by Means of Dynamic Stixels", in *Proc. of IEEE Intelligent Vehicles Symposium (IEEE-IV)*, 2010, pp. 217-224.

[16] Rathi, Y.; Vaswani, N.; Tannenbaum, A.; Yezzi, A., "Particle filtering for geometric active contours with application to tracking moving and deforming objects", in *Proc. of CVPR 2005*, vol. 2, 2005, pp. 2-9.0

[17] Jackson, J.D.; Yezzi, A.J.; Soatto, S., "Tracking deformable moving objects under severe occlusions", *in Proc. of 43rd IEEE Conference on Decision and Control, CDC*, Vol.3, 14-17 Dec. 2004, pp.2990-2995

[18] M. Isard and A. Blake. Condensation, "Conditional density propagation for visual tracking", in *International Journal of Computer Vision*, 29(1):5–28, 1998.

[19] A. Vatavu and S. Nedevschi, "Real-time modeling of dynamic environments in traffic scenarios using a stereo-vision system", in *Proc. of IEEE 15th International Conference on Intelligent Transportation Systems (ITSC 2012)*, pp.722-727, 2012.

[20] A. Doucet, N. De Freitas, K. Murphy, S. Russell, "Rao–Blackwellised particle filtering for dynamic Bayesian networks" in *Proc. of the Sixteenth conference on Uncertainty in artificial intelligence*, 2000, pp. 176–183.

[21] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem", in *Proc. of the AAAI National Conference on Artificial Intelligence*, pp. 593–598, 2002.

[22] J. I. Woodill, G. Gordon, and R. Buck, "Tyzx deepsea high speed stereo vision system". In *Proc. of IEEE Computer Society Workshop on Real Time 3-D Sensors and Their Use*, Conference on Computer Vision and Pattern Recognition, 2004.

[23] C. Bibby, I. Reid, "Real-time tracking of multiple occluding objects using level sets", *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on* , vol., no., pp.1307,1314, 13-18 June 2010.