Modeling Unstructured Environments with Dynamic Persistence Grids and Object Delimiters in Urban Traffic Scenarios

Andrei Vatavu and Sergiu Nedevschi, *Member, IEEE*

***Abstract*—Modeling dynamic environments is an essential research topic in any driving assistance system. The complexity of the surrounding world, the measurement uncertainties or the unpredictable behavior of the traffic participants are the main factors that influence the detection and tracking process. In this paper we present a vision-based method for modeling and tracking unstructured dynamic environments. The proposed solution relies on raw information provided by a classified grid computed from a digital elevation map and employs two separate representation levels: a local dynamic persistence grid (DyPerGrid) that is generated as an intermediate representation level and a map of delimiters as a higher level obstacle description. A fast tracking solution is proposed by using the two models. The result is a geometrically consistent and accurate representation of the dynamic environment.**

I. Introduction

Modeling and tracking of dynamic environments is an important research topic for autonomous driving applications. The most of existing solutions consist in extracting a set of relevant features from the raw measurements that usually are acquired by laser [2][5], ultrasound[3] or vision-based sensors [4]. Typically, the dynamic properties of a moving entity are computed by associating and filtering those features that describe the same object over time. Despite the simplicity of the general idea, the dynamic environment representation remains a difficult problem. The complexity of the surrounding world, the measurement uncertainties or the unpredictable behavior of the traffic participants are the main factors that influence the modeling and tracking process. Therefore a driver assistance system should be able to address these tasks with high accuracy and time efficiency. The motion estimation approaches can be categorized by the type of used features. Some of the existing methods imply tracking of 3D point clouds [6], while other solutions are based on using high level attributes including voxels [9] , 2D boxes, 3D cuboids [12], difference fronts [14], polygonal models [13], or object contours [10][11]. Some of the existing approaches rely on using the intensity information provided by vision sensors [6]. Other solutions use only the object geometry information such as in the case of laser-based systems [2][9]. Many of the motion estimation techniques try to minimize the processing power by using intermediate representations as the input of the tracking step. The 3D information is transformed into occupancy grids [16], octrees [3][9], Stixel World [7] or digital elevation maps [15]. The tracking methods can also be separated into model based and grid based techniques [3][16]. The model based solutions are mostly used when the surrounding world is considered to be structured [12] or when the type of the tracked object is known [2]. However, in the case of unstructured environments, it is difficult to hold constrained models such as 2D boxes or 3D cuboids. A tracking algorithm may fail when the object geometry changes or when it is partially visible. Usually, the grid-based algorithms perform the tracking process before generating a model [16]. The occupancy grids is one of the most used solutions for intermediate level modeling. An occupancy grid represents a probabilistic map. Each map cell is described by a probability of being occupied or empty. One of the first occupancy grids representations was introduced by [17] to represent static environments. Later, various solutions were proposed to model dynamic grids [13][16][18]. In [16], a dynamic environment tracking method is proposed. A particle based occupancy grid is presented. The complexity of the proposed solution is linear with the grid size and with the total number of used particles. Most often, the dynamic environment representation is split into two separate tasks: simultaneous localization and mapping (SLAM), and detection and tracking of moving objects (DATMO) [13]. Usually, the Iterative Closest Points (ICP) algorithm [19] is used for data association in order to register the new measurement to the local maps and to correct the vehicle odometry. The objects that not fulfill the SLAM constraints are considered dynamic. In some, cases when the environment is fully dynamic, aligning entire maps at once may lead to erroneous results.

In this paper we present a method for modeling and tracking unstructured dynamic environments by using a dense stereo-vision system. The proposed solution (see Fig. 1) is based on information provided by a digital elevation map and employs two separate models:

- A Local Dynamic Persistence Grid (DyPerGrid): represents a 2D grid that is updated using the raw measurements, derived from the elevation map processing. Each grid cell denotes the persistence information (how often the cell has been detected as occupied).
- A map of delimiters: the blob delimiters are extracted from the DyPerGrid by considering only the cells with high persistence.

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

The proposed tracking method is organized in three main steps. The first step is performed at the blob level and consists in associating the new measurements to the existing DyPerGrid blobs. Also at this stage the object delimiters are extracted from the DyPerGrid and from the current measurements as well. At the second step, the motion information is computed by using a pairwise alignment of associated delimiters. A Kalman filter is employed in order to obtain an optimal estimate of speed vectors. The final step consists in updating the DyPerGrid representation by taking into account both the velocity vectors estimated in the previous stage and the new raw measurements. An advantage of this Blob-Delimiter-Blob approach is that it allows the extraction of more accurate obstacle models from the DyPerGrid representation. At the same time, fast motion estimation is achieved at a higher level of representation. The extracted speeds are back propagated to the blob level and are used to update the DyPerGrid. As the result a geometrically consistent representation of the dynamic environment is obtained.

The remainder of this paper is organized as follow: the DyPerGrid representation is described in the next section. Section III describes the delimiter model. The proposed dynamic environment representation approach is detailed in section IV. The experimental results and conclusions are given in the last two sections.

II. The Dynamic Persistence Grid (DyPerGrid) Model

The Dynamic Persistence Grid (DyPerGrid) model is represented by a 2D grid, mapping the world into discrete 0.1 m x 0.1 m cells. The map used in this work has a resolution of 240 rows x 500 columns (120000 cells). Unlike the occupancy grids, a DyPerGrid model determines how often a cell has been detected as occupied (the persistence of an object in that position). A multi-frame persistence map was proposed by [21] for curb detection task. Temporal filtering was used to reduce the 3D noise from dense stereo. However the proposed persistence map was considered to be static. In our case, the dynamic part of the world is also taken into a count. Thus each DyPerGrid cell m_{xz} is described by its position (x,z), speed vectors components (vx,vz) and a persistence value p_{xz}:

$$m_{xz} = (x, z, vx, vz, p_{xz}) \tag{1}$$

At each frame, the DyPerGrid is updated iteratively with the measurements received from a binary occupancy grid that is derived from the elevation map [1] projection on the horizontal plane. Each cell c_{ij} in the measurement grid is described by its position (i,j) and an occupancy value $Occ(i,j)$:

$$c_{ij} = (i, j, Occ(i, j)) \tag{2}$$

The $Occ(i,j)$ is 1 when c_{ij} is occupied and 0 otherwise. As in the case of reflection probability maps [20] the DyPerGrid map can be learned by using a so called counting model. According to this model, our goal is to

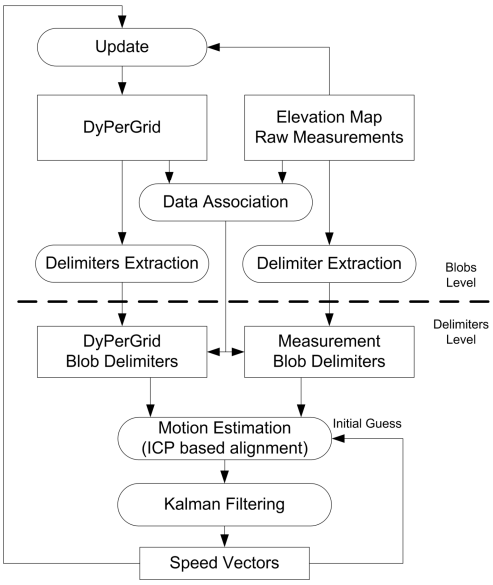

Figure 1. Dynamic Environment Modeling method overview.

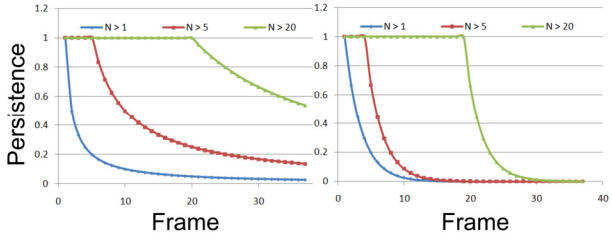

Figure 2. The persistence value converges to 0 once an occupied cell c_{ij} becomes empty. The memory effect is shown in (a) in the case when all the past observations are taken into account (the object state changes more slowly when N increases). The persistence value is calculated in (b) by using a moving average with a fixed sliding window of size N_w=3.

compute the most likely map m^* knowing the measurements $c_{1:t}$ and ego-car poses $E_{1:t}$:

$$m^* = \arg\max_m p(m \mid E_{1:t}, c_{1:t}) \tag{3}$$

By following the description in [20] and assuming that all cells m_{xz} are independent, the persistence value p_{xz} of each cell m_{xz} can be computed according to:

$$p_{xz} = \frac{N_{occ}}{N_{occ} + N_{free}} \tag{4}$$

where N_{occ} is the number of times the cell has been detected as occupied, and N_{free} is the number of times the cell has been detected as free. The persistence of a DyPerGrid cell m_{xz} can also be updated recursively from the previous value p_{xz}^{t-1} and the current occupancy $Occ(x,z)$ of the measurement cell:

$$p_{xz}^{t} = \frac{p_{xz}^{t-1} N + Occ(x,z)}{N+1} \tag{5}$$

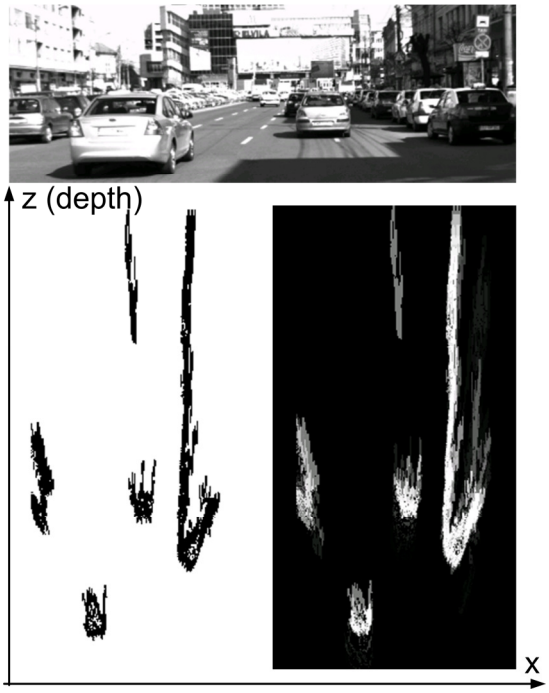

Figure 3. Bottom-left: The raw occupancy grid. Occupied cells are described with black color. Bottom-right: The updated DyPerGrid. High intensities indicates high persistence of cells.

where $N = N_{occ} + N_{free}$ is the total number of observations up to the time *t-1*.

One of the problems in the case of the updating policy described by (4) and (5) is that an object state changes very slowly when N is large (see Fig. 2). Therefore it cannot be applied for moving obstacles that are observed only for a short time. A possible solution is to use a subset with only the most recent N_w measurements (sliding window) and analyze the observations inside that subset. At each time, this subset is modified by including the new observation and excluding the oldest one. Additionally, weighting factors can be applied to each window element in order to emphasize particular observations in the subset. The persistence value can be calculated by using a moving average technique such as weighted moving average (WMA) or exponential moving average (EMA). In our proposed solution the measurement update step is performed by using a modified moving average (MMA). Fig. 3 illustrates an example with the raw occupancy grid and the corresponding DyPerGrid model for an urban scene. However, before updating the DyPerGrid model, we must take into account both the ego-vehicle movement and the motion of the other dynamic entities in the traffic scene. This is described later in this paper.

III. Object Delimiter Model

An object delimiter is defined by a set of contour points describing the object blobs. At each frame a map of delimiters is extracted from the DyPerGrid and another map is extracted from the measurement grid. The applied approach is similar in both cases. For this, we use the *BorderScanner* algorithm, described in one of our previous works [22]. The main idea of the *BorderScanner* method is to generate a better free-form obstacle model by selecting the most visible parts of the objects. This is achieved by using a scanning ray which extends from the ego-car position and moves in a radial direction with fixed increments. At each step the closest point that satisfies the selection criterion is chosen as the contour point. For each of the two models (DyPerGrid and measurement grid), different criteria are considered. For the measurement grid, the first occupied point l_i is taken into account. An extracted contour of M elements is defined as:

$$L_{measurement} = \{l_i \mid Occ(l_i) = 1, i \in [1..M]\} \quad (6)$$

In the case of the DyPerGrid the delimiter points are extracted by selecting the closest cells l_j that have the persistence value p_j greater than a given threshold τ. Thus, a DyPerGrid delimiter can be described as:

$$L_{DyPerGrid} = \{l_j \mid p_j > \tau, j \in [1..N]\} \quad (7)$$

In our experiments, for a sliding window of N_w=3, we used a persistence threshold of $\tau = 0.5$. Fig. 4 shows an example of delimiter extraction from the DyPerGrid model.

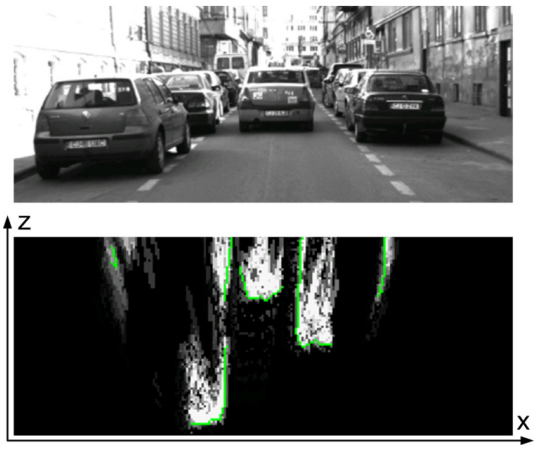

Figure 4. The delimiter extraction. Top – a traffic scene. Bottom – a part of the DyPerGrid model including the extracted delimiters (green) for the cars in the top image.

IV. The Blob-Delimiter-Blob method for object motion estimation

The proposed dynamic environment representation method could be divided into three main phases:

A. Blobs Association

The first step is performed at the blob level and consists in associating the new measurements to the existing DyPerGrid blobs. Before performing the data association, the DyPerGrid coordinate system (previous frame) must be aligned with the measurement one (current frame) in order to compensate the ego-vehicle motion. In our case the vehicle speed v and the yaw rate $\dot{\psi}$ are obtained from the car sensors through the CAN bus. We also know the time

delay Δt between two frames. These parameters are used to estimate the rotation $R_y(\psi)$ and translation $T=[t_x,t_z]^T$ of the car. By following the ego-vehicle motion model with constant yaw rate and constant speed, the t_x, t_z and ψ are estimated as:

$$t_x=\frac{v\Delta t}{\psi}(1-\cos\psi),\ t_z=\frac{v\Delta t}{\psi}\sin\psi \text{ and } \psi=\dot{\psi}\Delta t \qquad (8)$$

In order to compensate the ego-motion, each DyPerGrid cell $m(x_{t-1},z_{t-1})$ is transformed according to:

$$\begin{bmatrix}x_t\\z_t\end{bmatrix}=R_y(\psi)\begin{bmatrix}x_{t-1}\\z_{t-1}\end{bmatrix}+\begin{bmatrix}t_x\\t_z\end{bmatrix} \qquad (9)$$

Our next objective is to find blobs that identify the same obstacle in the DyPerGrid and the measurement map. We define a DyPerGrid blob as a finite collection of connected cells that have the persistence value greater than a given threshold τ:

$$A=\{a_i \mid p_i>\tau, i=[1..N_A]\} \qquad (10)$$

Similarly, a certain blob from the measurement grid is described by a set of connected cells that are occupied:

$$B=\{b_j \mid Occ(b_j)=1, j=[1..N_B]\} \qquad (11)$$

We denote with $O(a_i,b_j)$ the overlap function between two given points from A and B. This function is 1 when the two cells overlap and 0 otherwise. For each blob from the DyPerGrid and for each blob from the measurement grid we calculate an overlapping score:

$$w_{AB}=A\cap B=\sum_{i=1}^{N_A}\sum_{j=1}^{N_B}O(a_i,b_j) \qquad (12)$$

It must be noted that $w_{AB}=w_{BA}$, and $O(a_i,b_j)=O(b_j,a_i)$. As the result, a score matrix is computed: $W=\{w_{ij}\}$. Subsequently, we define the most likely association from A to B (forward association) as:

$$Assoc(A)=\arg\max_B P(B\mid A)=\arg\max_B\frac{w_{AB}}{N_A} \qquad (13)$$

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

$$Assoc(B)=\arg\max_A P(A\mid B)=\arg\max_A\frac{w_{AB}}{N_B} \qquad (14)$$

This double association allows us to consider the cases when larger blobs are split into many disjoint sets or vice versa.

Having the two collections of blobs described as $S_A=\{A_i \mid i\in[1..M]\}$ and $S_B=\{B_j \mid j\in[1..N]\}$, the result collection S of distinct associated pairs is given by:

$$S=\{(A_i,B_j)\mid Assoc(A_i)=B_j, i\in[1..M], j\in[1..N]\}\cup$$
$$\{(B_j,A_i)\mid Assoc(B_j)=A_i, Assoc(A_i)<>B_j, i\in[1..M], j\in[1..N]\}$$

Finally, the data association information is passed to the delimiter level in order to identify a set of associated contour pairs, so that the selected candidates are used for estimating the motion of objects in the traffic scene.

B. Delimiter-based motion estimation

At the second step, the speed vectors of the traffic participants are computed by using a fast pairwise alignment of the associated object delimiters. One of the most used techniques for registering two point sets in a common coordinate system is the Iterative Closest Point (ICP) algorithm. The ICP method was first proposed by Besl and McKay [19] and is widely used especially for scan matching approaches. Previously, the ICP algorithm was used by us in [8] to align objects considering only the previous and current frames. In this work we adapt the alignment method by using the contours extracted from the DyPerGrid and from the measurement grid as the input for the ICP phase. The main idea is to find an optimal rotation R and translation T by minimizing the alignment error between the two associated delimiters.

We denote by $P=\{p_1,p_2,...,p_M\}$ a model set that describes the DyPerGrid delimiter, 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 point p_i from P. Thus, our objective function can be defined as:

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

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

1. Matching. For each data point q_i from Q the closest model point from P is found:

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

In our case the closest corresponding points are determined by using a distance transform map.

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 is computed by minimizing the objective function described by the relation (15). As equation (15) is 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. This step consists in updating the final transformation matrix M_g and the position of the model contour with the estimated R and T.

5. Testing the convergence. Test if the algorithm has been achieved a minimum error by calculating the average point-to-point distance between the two corresponding sets:

$$\varepsilon_{ICP} = \frac{1}{N}\sum_{i=1}^{N}\|p_i - q_i\| \qquad (17)$$

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

In order to obtain an optimal estimate of speed vectors, a standard Kalman filter is used. For each object we track its position and the two speed components $x = (x_o, z_o, v_x, v_z)^T$. In our case the process covariance matrix Q is determined by taking into account the obstacle's acceleration while the measurement covariance matrix R is estimated by following the uncertainty model described in [16].

C. DyPerGrid update

This stage consists in a *motion update* followed by a *measurement update* of the DyPerGrid.

As the result of the delimiter alignment, each DyPerGrid blob is described by a rotation $R(\alpha)$, a translation $T = [t_x, t_z]^T$ and a speed vector $\vec{v} = [v_x, v_z]^T$. In the *motion update* step, the position of each cell $m(x,z)$ is modified according to the estimated $R(\alpha)$ and T:

$$\begin{bmatrix} x' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} - \begin{bmatrix} t_x \\ t_z \end{bmatrix} \qquad (18)$$

In the *measurement update* step the persistence of each DyPerGrid cell $m(x,z)$ is updated recursively from its previous persistence value p_{xz}^{t-1} and the current measurement $Occ(x,z)$. Unlike the classical update policy described by the equations (4) and (5) we use a modified moving average (MMA) technique with a fixed sliding window of size N_w:

$$p_{xz}^{t} = \frac{p_{xz}^{t-1}(N_w - 1) + Occ(x,z)}{N_w} \qquad (19)$$

Fig. 5.b shows an example with the evolution in time of cell persistence, given a sequence of measurements. The MMA method with different sliding window sizes N_w is compared.

V. Experimental Results

The proposed dynamic environment modeling solution has been tested on various urban traffic scenarios including cross traffic and occluded obstacles. For our experiments we used a computer equipped with an Intel Core 2 Duo CPU and 2.66GHz with 2GB of RAM. For the processing steps we used rectified and down-sampled images with 512 pixel width.

Fig. 5.c and Fig. 5.d show the difference between a static grid representation and our proposed DyPerGrid solution for dynamic environments representation. In Fig. 6, the dynamic obstacles are described by attributed polygonal models. Different traffic scenarios are tested including: approaching (yellow) and outgoing (blue) vehicles (see Fig. 6.b), occluded obstacles with different orientation (see Fig. 6. c), crossing vehicles (Fig. 6.d).

Qualitative results from various traffic scenes are shown in the Fig. 6. The orientation of a moving obstacle is described by the color hue, while its speed is given by the saturation (white color means a static object).

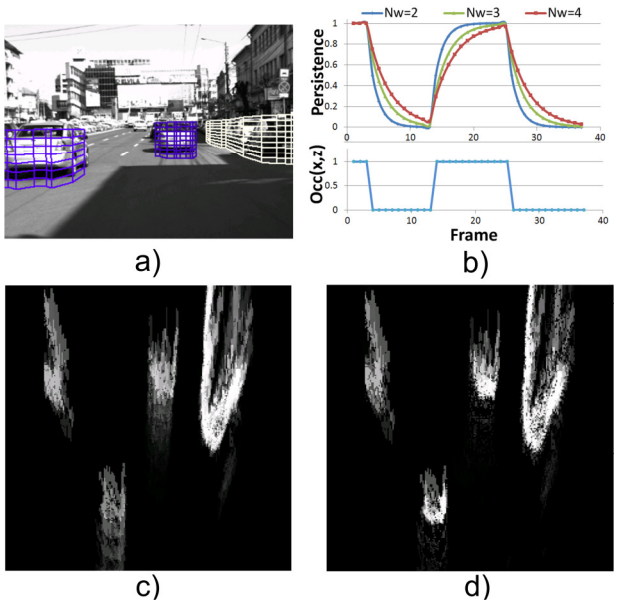

Figure 5. a) An urban traffic scene. b) The evolution in time of a cell persistence, given a sequence of measurements. The MMA method with different sliding window sizes N_w is compared. c) The resulted representation for the scene (a) in the case of Static Grid asumption (without motion update). d) A more consistent model is obtained in the case of DyPerGrid representation and Blob-Line-Blob motion estimation approach.

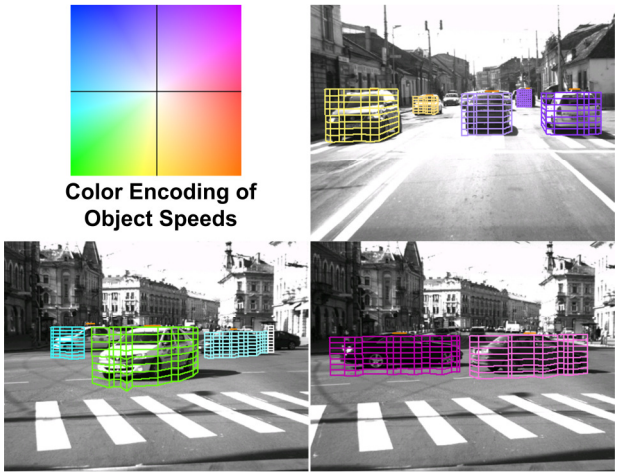

Figure 6. Different urban traffic scenarios with dynamic obstacles. The object speed vectors are color encoded as shown in top-left image. The hue is used for orientation while the saturation is for object speed (white means a static obstacle).

For the numerical evaluation we have included two cases. In both tests the proposed blob-delimiter-blob approach was compared with a solution that uses a particle based filtering mechanism [16]. The first case consists in a "follow the leader" scenario with a moving car in front of the ego vehicle. The ego-car speed represents the ground truth. The mean absolute error (MAE) is chosen as the accuracy metric. The resulted accuracy of the delimiter-based tracking solution (2.81 Km/h) is comparable to the particle filtering method (1.29 Km/h). However, our proposed method proves to be much faster (16.08 ms) than the particle based filtering approach (119.1 ms). The second test implies an incoming vehicle crossing an intersection. The speed estimation values are shown in the Fig. 7. The moving vehicle is in the field of view for only 2.5 seconds. It can be observed that the blob-delimiter-blob solution converges more quickly to the real speed of the car.

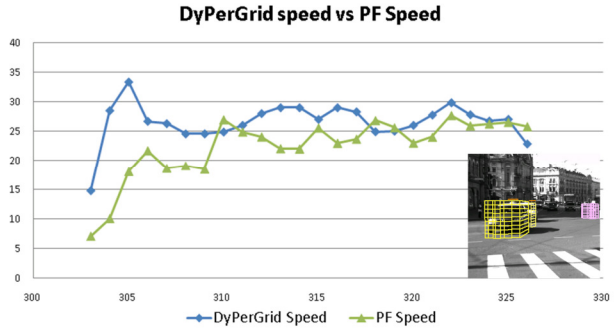

Figure 7. A comparison between the extracted speeds by using a delimiter based motion estimation (blue) and a particle-based occupancy grid (green).

VI. Conclusions

In this paper we presented a vision-based solution for modeling dynamic unstructured. The proposed method is based on information provided by a classified occupancy grid computed from a digital elevation map. Two separate representation levels are employed: a Local Dynamic Persistence Grid (DyPerGrid) and a map of delimiters. The tracking method is organized in three main steps. The first step is performed at the blob level and consists in associating the new measurements to the existing DyPerGrid blobs. At this stage the object delimiters are extracted from the DyPerGrid and from the current measurements as well. At the second step, the motion information is computed by using a pairwise alignment of associated delimiters. A Kalman filter is employed in order to obtain an optimal estimate of speed vectors. The final step consists in updating the DyPerGrid representation by taking into account both the extracted object speeds and the new measurements. The proposed method is described by a low processing-time and an accuracy that is comparable to the other complex tracking approaches.

References

[1] F. Oniga, 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.

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

[3] 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

[4] A. Barth and U. Franke, "Where will the oncoming vehicle be the next second?", in *Proc. of Intelligent Vehicles Symposium*, 2008 IEEE, pp.1068-1073, 4-6 June 2008

[5] T. de Candia, "3d tracking of dynamic objects with a laser and a camera", Technical report, Autonomous System Lab, ETH Zurich, 2010.

[6] U. Franke, C. Rabe, H. Badino, and S. Gehrig, "6d-vision: Fusion of stereo and motion for robust environment perception," in *27th Annual Meeting of the German Association for Pattern Recognition DAGM '05*, 2005, pp. 216-223.

[7] 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.

[8] 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, 16-19 Sept, 2012.

[9] 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,

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

[11] 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, pp. 271-276, 2005

[12] R. Danescu, S. Nedevschi, M.M. Meinecke, and T. Graf, "Stereovision Based Vehicle Tracking in Urban Traffic Environments", in *Proc. of the IEEE Intelligent Transportation Systems Conference (ITSC 2007)*, Seattle, USA, 2007

[13] 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.

[14] 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)*, pp.717-722, 3-7 June 2012

[15] R. Danescu, F. Oniga, S. Nedevschi, M-M. Meinecke, "Tracking Multiple Objects Using Particle Filters and Digital Elevation Maps", in *Proc. of the IEEE Intelligent Vehicles Symposium (IEEE-IV 2009)*, June 2009, Xi'An, China, pp. 88-93.

[16] R. Danescu, F. Oniga, S. Nedevschi, "Modeling and Tracking the Driving Environment with a Particle Based Occupancy Grid", in *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, No. 4, December 2012, pp. 1331-1342.

[17] A. Elfes, "A Sonar-Based Mapping and Navigation System", in *Proc. of IEEE International Conference on Robotics and Automation*, April 1986, pp. 1151-1156

[18] T. Gindele, S. Brechtel, J. Schroeder, R. Dillmann, "Bayesian Occupancy Grid Filter for Dynamic Environments Using Prior Map Knowledge", in *Proc of IEEE Intelligent Vehicles Symposium*, 2009, pp. 669-676.

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

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

[21] F. Oniga, S. Nedevschi, M.M. Meinecke, "Curb Detection Based on a Multi-Frame Persistence Map for Urban Driving Scenarios", in *Proc. of IEEE 11th International Conference on Intelligent Transportation Systems (ITSC 2008)* , pp.67-72, 12-15 Oct. 2008

[22] 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.