Real-Time Modeling of Dynamic Environments in Traffic Scenarios using a Stereo-Vision System

Andrei Vatavu and Sergiu Nedevschi, *Member, IEEE*

Abstract—**Real-time modeling of dynamic environments is one of the most demanding research problems in the field of driving assistance systems. The representation module may be affected by several factors such as occlusions, unpredictable nature of the traffic participants, wrong associations or noisy measurements. In this paper we propose two different methods for real-time modeling of dynamic environments. Both motion estimation techniques are vision-based and rely on information provided by a Digital Elevation Map. The first approach consists in determining the differences between the previous and current frames. These differences are then used for computing the speeds of the traffic participants. The second motion estimation technique consists in using a fast pairwise alignment of object delimiters that are extracted by radial scanning of the Elevation Map. The final result is a more compact polygonal map with associated static and dynamic features.**

I. Introduction

In the context of Advanced Driver Assistance Systems (ADAS), real-time modeling of dynamic environments is one of the most demanding research problems. The detection and tracking of moving traffic entities is an indispensable intermediate step for higher level in-vehicle applications such as parking assistance, path planning or collision detection. Unlike the clearly structured traffic scenarios where the objects are represented by 2D bounding boxes or 3D cuboids, modeling the dynamic entities becomes a difficult task when the environment to be tracked is a busy urban center or an intersection. The representation module may be affected by several factors such as occlusions, unpredictable nature of the traffic participants, wrong associations or noisy measurements.

A common solution for tracking dynamic obstacles consists in extracting a set of relevant object attributes from the scene and estimating their motion over time. Current approaches can directly use 3D points[1] or they can track high level features such as stixels [3], object contours [5] [9], 2D bounding boxes or 3D cuboids [2], free form polygonal models [4] etc. The dynamic environment representation techniques can be divided depending on the type of used sensors. Current systems includes laser [7][8][20], sonar [10], radar[20] or vision based sensors [2]. The motion estimation methods can also be categorized by the level at which the dynamic features extraction is applied. Some of the existing approaches are based on estimating the motion before computing the object model [11][13], while other solutions rely on extracting some relevant features and subsequently estimating their dynamic properties [2][4][5]. Many of the techniques for movement detection use intermediate representations as primary information. A common solution is mapping 3D information into octrees [10], digital elevation maps [12] or occupancy grids [11].

Dynamic obstacle detection approaches also differ by the data association and the way the correct correspondences are determined. Some methods try to detect the appearance of objects in successive frames. The motion is inferred by measuring different displacement of the same object over time. In [20] a difference map is computed by determining the object changes in two consecutive virtual scans. The obstacle's center of mass and its rectangular shape are used as inputs for the tracking process.

One of the most used methods for model fitting is the RANSAC algorithm [14]. The RANSAC approach has the advantages of being robust against outliers. However, its accuracy depends on the number of used samples and may lead to a high computational cost.

Direct matching techniques, such as Iterative Closest Point (ICP) [15] method, are widely used in vehicle localization and mapping [4]. Several ICP variants are compared in [16]. In [4] a map with moving objects is segmented by considering that dynamic attributes do not fulfill the constraints of the SLAM. The most of ICP methods are adapted for laser systems and do not take into account the odometry information. However, the data association in consecutive scans is difficult to be achieved when the entities from the traffic scene or the ego vehicle moves at high speed or when the measurement uncertainties are not considered.

In this paper we propose two different methods for real-time modeling of dynamic environments. Both motion estimation techniques rely on information provided by a Digital Elevation Map. Unlike the intensity based registration techniques, such as optical flow, that rely on identifying the corresponding features based on the intensity of points, the proposed methods in this work are based on finding object correspondences by using only the object geometry information. The first approach consists in computing the differences between the previous and current frames and generating an evidence space called Difference Map. For each moving object three types of areas are defined: a direction front indicating the direction of the moving obstacles, a shadow front that is located behind the moving obstacles, and an object's core area that remains unchanged in the consecutive frames. The speeds of the

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

traffic participants are directly computed from the moving object differences. The second motion estimation technique consists in using a fast pairwise alignment of object delimiters that are extracted by radial scanning of the Elevation Map. Unlike the other existing approaches that consist in aligning the whole local maps at once, and then separating the dynamic entities from the static ones, we first associate the objects at the blob level and then apply the ICP for each associated candidate. In order to stabilize the results, the speed vectors extracted by the two methods are subjected to a Kalman filtering. The final result is a more compact polygonal map with associated static and dynamic features.

The remaining of the paper is structured as follows: Section 2 presents the architecture of the proposed dynamic environment modeling system. The preprocessing tasks that are necessary for the movement detection are detailed in the Section 3. Section 4 describes the two motion estimation techniques: based on Difference Map analysis and based on pairwise alignment of the object delimiters. Section 5 describes the filtering. The experimental results and conclusions are presented in the last two sections.

II. System Architecture

The dynamic environment modeling module has been conceived for urban driving scenarios by extending our previous Dense Stereo-Based Object Recognition System (DESBOR) [6]. The extended system (Figure 1) is composed by the following main components:

Image acquisition and 3D reconstruction – in this step the images are acquired from the two cameras at a full resolution. The images are undistorted, rectified and down sampled to a width of 512 pixels. Then, 3D reconstruction is performed by using a TYZX hardware board [18].

Intermediate representation – at this stage the reconstructed 3D points are used to compute a Digital Elevation Map (Figure 2.b and 2.c). This intermediate representation is described by a grid of heights. Each grid cell is classified as road, obstacle and traffic-isle. A detailed description about the Elevation Map is presented in [19].

Delimiters Extraction – the obstacle delimiters are extracted by radial scanning of the Elevation Map. With this approach, the erroneous results caused by the occlusions are minimized. A list of static polygonal models is generated as the result (Figure 2.b). Each delimiter inherits the type, position and height properties of the associated Elevation Map blobs. More details about delimiters extraction are presented in [17].

Preprocessing – the preprocessing module consists in performing a set of tasks that are necessary for the motion estimation component. First, the ego-vehicle motion is compensated in order to separate its speed from the motion of the other obstacles in the traffic scene. Then, the data association is computed at the object level by overlapping the Elevation Map blobs. A list of associated delimiter pairs is passed to the motion estimation step.

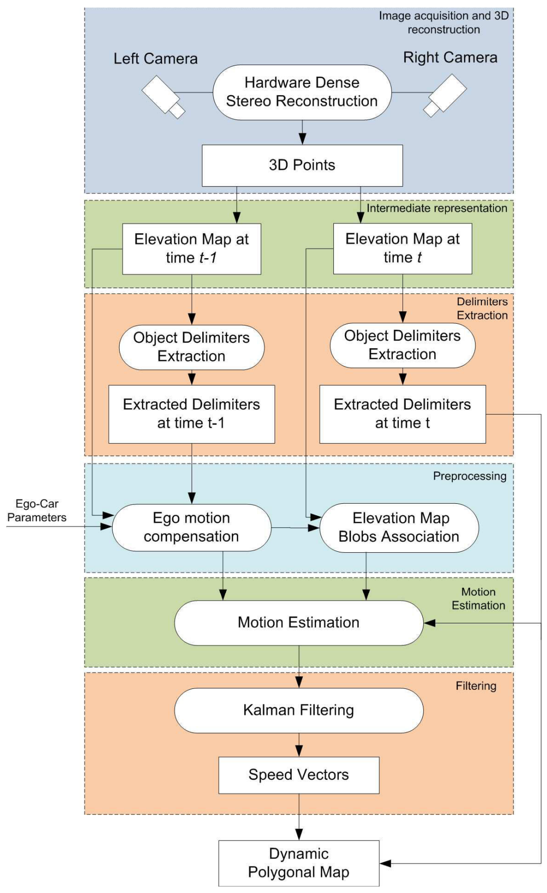
Image acquisition and 3D reconstruction
Left Camera
Right Camera
Hardware Dense Stereo Reconstruction
3D Points
Intermediate representation
Elevation Map at time t-1
Elevation Map at time t
Delimiters Extraction
Object Delimiters Extraction
Object Delimiters Extraction
Extracted Delimiters at time t-1
Extracted Delimiters at time t
Preprocessing
Ego-Car Parameters
Ego motion compensation
Elevation Map Blobs Association
Motion Estimation
Motion Estimation
Filtering
Kalman Filtering
Speed Vectors
Dynamic Polygonal Map

Figure 1. System Architecture

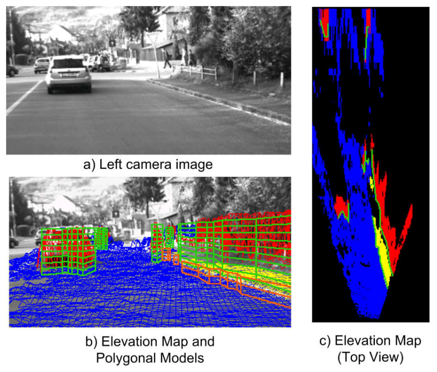
a) Left camera image
b) Elevation Map and Polygonal Models
c) Elevation Map (Top View)

Figure 2. a) Left camera image. b) Elevation Map and the extracted polygonal models are projected on the left camera image (green – obstacles, yellow – traffic isles). c) The Elevation Map - Top View. The Map's cells are classified as road (blue), traffic isle (yellow) and obstacle (red).

Motion Estimation – two different methods are used for the motion estimation. The first method is based on using the Elevation Map information from the previous and current frame in order to compute a map of differences. Based on this map of differences we extract three types of areas called difference fronts, which serve for the speed vectors estimation. The second approach uses the associated delimiter pairs as the input data. The velocity information is computed by performing a pairwise alignment of the associated candidates.

Filtering – a standard Kalman filter is employed in order to compute the objects' filtered position and speed.

A dynamic polygonal map is generated as the result. Each polyline element is characterized by its static and dynamic features (size, position, height, type and speed).

III. Preprocessing

The preprocessing level consists in performing a set of tasks necessary for the subsequent stages.

A. Ego-Motion Compensation

The ego-motion is compensated in order to separate the ego-vehicle speed from the motion of other traffic entities. For a given point $P_{t-1}(x_{t-1}, y_{t-1}, z_{t-1})$ from the previous frame, the corresponding position $P_t(x_t, y_t, z_t)$ in the current frame is estimated with the following relation:

$$\begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix} = R_y(\theta)\begin{bmatrix} x_{t-1} \\ y_{t-1} \\ z_{t-1} \end{bmatrix} + \begin{bmatrix} t_x \\ 0 \\ t_z \end{bmatrix} \quad (1)$$

Where θ is the rotation angle around the Y axis, and t_x, t_z represent the ego-vehicle translation components on the X respectively Z axis. It must be noted that the rotation and translation parameters are estimated by using the information provided by the onboard sensors (speed and yaw rate).

B. Data Association

This step consists in associating the object delimiters in consecutive frames. We use the spatial overlap of objects as the similarity measure. For each object O_i from the previous frame and for each object C_j from the current frame we compute an overlapping score S_{ij}. This score is given by the number of points that overlap. The results are stored into a score matrix $S=\{S_{ij}\}$. The associated pairs are determined by extracting candidates with the highest score from the S matrix. Two types of associations are used: a forward association, that estimates best overlapping entities in the current frame for all blobs in the previous frame, and a backward association that finds corresponding objects in the previous frame for all blobs from the current frame. A list with all distinct pairs associated in the two steps is generated as the result. It must be admitted that the association step is limited by the quality of reconstruction as well as by the speed of the traffic objects. However, considering the velocity of 50km/h as the speed limit in urban traffic scenarios, and the camera frame rate of 20fps, the overlapped area of the same car in two consecutive frames is about 75%. Therefore, this would be sufficient that the object association to be achieved.

IV. Motion Estimation

We have used two different approaches for the motion estimation stage.

A. Motion Estimation using Difference Maps

A map of differences is computed by overlapping the two consecutive Elevation Map representations. Unlike the intensity based registration techniques, such as optical flow, that rely on identifying the corresponding features based on the point intensity in a image space, our method is based on extracting dynamic features by taking into account the 3D object geometry. For each cell in the previous frame we check its occupancy at the same position in the current frame. The resulted Difference Map points may belong to the following classes:

Direction – the cells that are empty in the previous frame and are occupied in the current frame.

Shadow – the cells that are occupied in the previous frame and are empty in the current frame.

Core – the cells that are occupied in the both frames at the corresponding position.

Based on the Difference Map information, a moving object is described by the three types of fronts (Figure 3, left): a direction front (the direction of the moving obstacle), a shadow front (located behind the moving obstacle) and a core area (the unchanged object part in the consecutive frames).

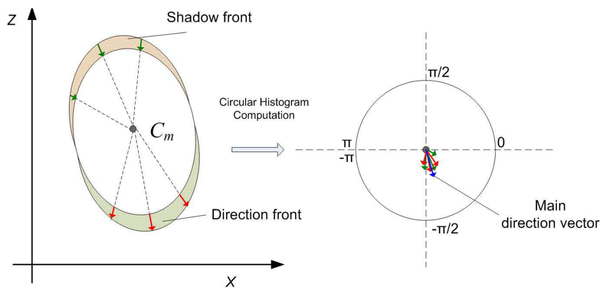

Figure 3. The main direction vector is computed by using a circular histogram. The histogram data samples are extracted from difference fronts.

For each dynamic entity we calculate a dominant direction vector by gathering a set of circular measurements for difference fronts and storing them into a circular histogram (Figure 3, right). Thus, for a cell situated at the object boundary we accumulate all the difference points into the histogram by moving along a virtual ray towards the object center of mass C_m. Each computed data sample is described by an angle θ_i and a magnitude M_i (the magnitude is directly proportional to the number of accumulated difference points). The mean direction of a moving object is computed by using vector addition for all components accumulated into the histogram.

$$\overline{\theta} = atan2(V_z, V_x). \quad (2)$$

where:

$$V_x = \sum_{i=1}^{N} M_i \cos(\theta_i), V_z = \sum_{i=1}^{N} M_i \sin(\theta_i) \quad (3)$$

The mean vector magnitude is defined by:

$$r = \frac{1}{N}\sqrt{{V_x}^2 + {V_z}^2} \quad (4)$$

where N is the number of individual vectors in the histogram.

B. *Motion Estimation based on ICP technique*

Unlike the previous method, this approach is based on the information provided by the list of associated contour pairs. For each distinct pair we compute correspondences between the two object delimiters and estimate an optimal rotation R and a translation T that minimize the alignment error. For the contour alignment we use the Iterative Closest Point (ICP) approach [15]. For each pair that identifies the same object in the consecutive frames we define two set of points: a model set $P=\{p_1, p_2, ..., p_M\}$ that describes the object contour in the previous frame, and a data set $Q=\{q_1, q_2, ..., q_K\}$ that describes the object contour in the current frame. For each point q_j from Q the corresponding closest point p_i from P is found. We want to find an optimal rotation R and translation T that minimize the alignment error. The objective function is defined:

$$E(R,T) = \sum_{i=1}^{N} \left\| Rp_i + T - q_i \right\|^2 \quad (5)$$

where p_i and q_i are the corresponding point pairs of the two sets and N is the total number of correspondences. The proposed alignment method consists in the following main steps:

1) Matching: for each point q_i from data set Q, the closest point from the model set P is found.

$$d(q_i, P) = \min_{j \in \{1..N_p\}} d(q_i, p_j) \quad (6)$$

A list of correspondent pairs is generated. Usually this task is the most computationally extensive in the ICP algorithm. The classical brute force search approach has a complexity of $O(N_q \cdot N_p)$. In order to reduce the complexity to $O(N_q \cdot \log N_p)$ many ICP methods employ KD-Tree data structures [22]. In our case, for finding closest points, we use a modified version of Chamfer based Distance Transform [21]. For each separate model contour, two maps (Figure 4) are computed: a distance map that stores the minimum distances to the closest points, and a correspondence map, storing the positions of the closest points.Thus, for each point from the data set, a corresponding point is found by superimposing the data contour on the two masks.

2) Outliers Rejection: The purpose of this stage is to filter erroneous correspondences that could introduce a bias in the estimation of translation and rotation. Two types of rejection strategies are used: distance based rejection, and boundary based rejection. The first strategy consists in eliminating the pairs whose point-to-point distance $d(q_i, p_j)$ is larger than a given threshold D_t. In addition the stereo-system uncertainties are taken into account. As suggested by [11], if we assume that the stereo-vision system is rectified, then the z error is computed:

$$\sigma_z = \frac{z^2 \cdot \sigma_d}{b \cdot f} \quad (7)$$

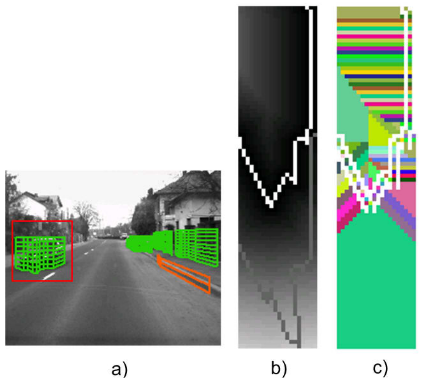

Figure 4. An example of Distance Transforms (b) and Corresponding Mask (c) that are computed for a dynamic obstacle (a). Data delimiters (gray color) and model delimiters (white color) are superimposed on the Distance Transform image. The Corresponding Mask's colors (c) identify uniquely the closest point from the model set.

Where f is the focal length, z is the depth distance, b is the stereo system baseline and σ_d denotes the disparity error. For each corresponding pair (p_i, q_i) from the two sets, the rejection is made according to the following relation:

$$d(q_i, p_j) > D_t + \sigma_z \quad (8)$$

The second type of rejecting consists in removing the point correspondences caused by incomplete overlap. Usually, these situations occur when one of the two contours is incompletely extracted due to occlusions.

3) Error Minimization: in this step we determine an optimal transformation M by minimizing the objective function defined by Equation (5).The rotation matrix R around the Y axis is linearized by approximating $\cos\alpha$ by 1 and $\sin\alpha$ by α:

$$R_y(\alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \approx \begin{bmatrix} 1 & 0 & \alpha \\ 0 & 1 & 0 \\ -\alpha & 0 & 1 \end{bmatrix} \quad (9)$$

The rotation R and the translation T are combined into a single transformation matrix M:

$$M = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \alpha & t_x \\ 0 & 1 & 0 & t_y \\ -\alpha & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

The Equation (5) can be rewritten as:

$$E(R,T) = \sum_{i=1}^{N} \left\| \begin{bmatrix} 1 & 0 & \alpha & t_x \\ 0 & 1 & 0 & t_y \\ -\alpha & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_{i,x} \\ p_{i,y} \\ p_{i,z} \\ 1 \end{bmatrix} - \begin{bmatrix} q_{i,x} \\ q_{i,y} \\ q_{i,z} \\ 1 \end{bmatrix} \right\|^2 \quad (11)$$

The $E(R,T)$ is minimized with respect to α, t_x, t_y, and t_z by setting the partial derivatives to zero:

$$\begin{cases} \dfrac{\partial E(R,T)}{\partial \alpha} = 2\sum_{i=1}^{N}\begin{bmatrix} \alpha(p_{i,x}^2 + p_{i,z}^2) + t_{i,x}p_{i,z} \\ -t_{i,z}p_{i,x} - q_{i,x}p_{i,z} + q_{i,z}p_{i,x} \end{bmatrix} = 0 \\ \dfrac{\partial E(R,T)}{\partial t_x} = 2\sum_{i=1}^{N}\left(t_{i,x} + p_{i,x} + \alpha p_{i,z} - q_{i,x}\right) = 0 \\ \dfrac{\partial E(R,T)}{\partial t_y} = 2\sum_{i=1}^{N}\left(t_{i,y} + p_{i,y} - q_{i,y}\right) = 0 \\ \dfrac{\partial E(R,T)}{\partial t_z} = 2\sum_{i=1}^{N}\left(t_{i,z} + p_{i,z} - \alpha p_{i,x} - q_{i,z}\right) = 0 \end{cases} \qquad (12)$$

The unknown coefficients α, t_x, t_y, and t_z are determined by solving the system of equations (12).

4) Updating: assuming that we have estimated new *R* and *T* parameters in the previous step, a new target set is computed by applying the new transformation *M* to the model set. A global transformation M_G is updated:

$$M_G = M_G \cdot M \qquad (13)$$

5) Convergence testing: at this step an error metric is estimated by computing the average point-to-point distance between the measurement set and the transformed model:

$$Err = \frac{1}{N}\sum_{i=1}^{N}\left\| p_i - q_i \right\| \qquad (14)$$

If the error is greater than a given threshold, the process continues with a new iteration. The algorithm stops when the computed error is below the selected error threshold or when a maximum iteration number has been exceeded.

V. FILTERING

Each object is tracked in order to compute its filtered position and speed. A standard Kalman filter is used. The state of each obstacle is described by the coordinates of the obstacle center of mass $C_m(X_m, Z_m)$ and by the mean direction vector speed components $\vec{r} = (V_x, V_z)$:

$$x = (X_m, Y_m, V_x, V_y)^T \qquad (15)$$

For each new object a new tracker is initialized. If an obstacle has an associated object in the previous frame, then the associated tracker is updated with the current frame measurements and the obstacle state is estimated based on the current measurements. The measurement covariance matrix *R* is computed based on the stereo uncertainty model described in [11]. The process covariance matrix *Q* is estimated considering a certain covariance for the obstacles' acceleration.

VI. EXPERIMENTAL RESULTS

We tested our dynamic environment representation approaches on 23 traffic scenarios from urban environments. For our experiments we used a 2.66GHz Intel Core 2 Duo Computer with 2GB of RAM. The data acquisition was made at a maximum frame rate of 24fps. The acquisition system is composed by two grayscale cameras including 2/3" CCD sensors with a fixed focal length of 6.5mm. Our processing steps are based on undistorted, rectified and down-sampled images (512 pixel width). Figure 5 presents intermediate results obtained with the two motion estimation methods for the same traffic scene. The dynamic obstacles with a speed greater than 15km/h are colored with red. The resulted difference map is presented in Figure 5.b. We used blue color for the direction fronts and magenta color for the shadow fronts. Figure 5.c presents the pairwise alignment of the two associated object delimiters. The model delimiter (extracted in the previous frame) is colored with yellow, while the data contour (extracted in the current frame) is drawn with blue. The ICP alignment result is represented with red.

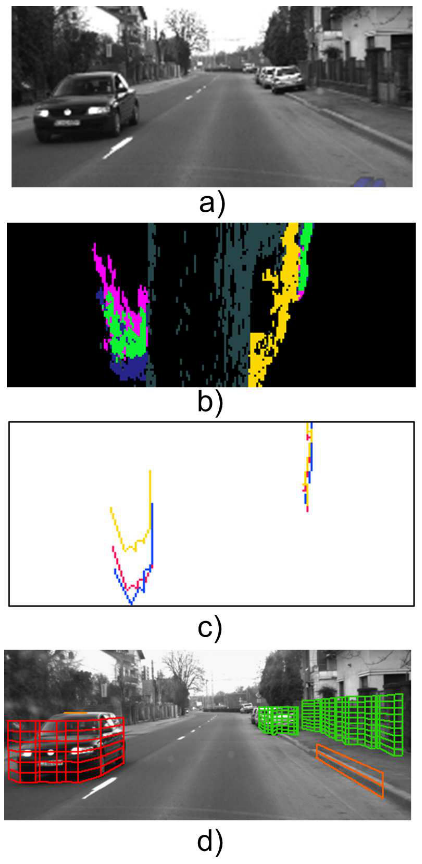

Figure 5. The motion estimation with intermediate results: a) A traffic scene b) Resulted Difference Map. Direction fronts are colored with blue, shadow fronts are colored with magenta and core area is represented with green. The yellow color is used for traffic isles. c) The ICP alignment result (red color) between a model delimiter that was extracted in the previous frame (yellow) and a data delimiter that was extracted in the current frame (blue). d) the environment representation result with dynamic (red) and static (green) obstacles.

In Figure 6 the two motion estimation methods are compared in terms of their extracted speeds.

Table 1 shows a comparative result obtained in the case of the two motion estimation techniques. In order to compute the speed estimation accuracy we used a stationary test vehicle with a target speed of zero as the ground truth for our measurements. For the ICP based motion estimation we set the maximum number of iteration to 10. We obtained a better accuracy in the case of ICP approach. However its accuracy depends on the number of iterations used to align the two object delimiters.

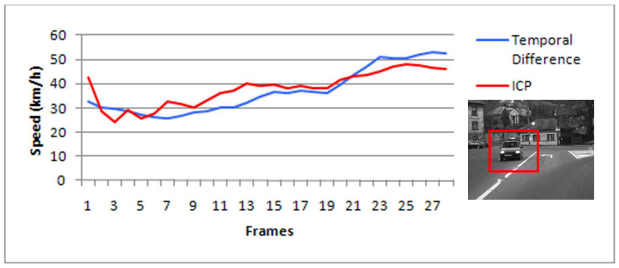

Figure 6. The estimated speeds in the case of the motion estimation method based on difference maps (blue color) and the mootion estimation based on ICP alignment (red color).

TABLE I. SPEED ESTIMATION

	Speed Estimation Methods	
	Difference Maps	***ICP***
Accuracy (MAE)	9.8Km/h	7.5Km/h
Average processing time per frame	8.4ms	28.3ms

VII. CONCLUSIONS

In this paper we propose two different methods for real-time modeling of dynamic environments. Both motion estimation techniques are based on information provided by a Digital Elevation Map. Our first approach consists in computing the differences between the previous and current frames and generating an evidence space called Difference Map. For each moving object three types of areas are defined: a direction front indicating the direction of the moving obstacles, a shadow front that is located behind the moving obstacles, and an object's core area that remains unchanged in the consecutive frames. The speeds of the traffic participants are directly computed from the moving object differences. The second motion estimation technique consists in using a fast pairwise alignment of object delimiters that are extracted by radial scanning of the Elevation Map. Unlike the other approaches that align the whole local maps at once, and then extract the dynamic features, we first associate the objects at the blob level and then apply the ICP for each associated candidate. In order to stabilize the results, the extracted speed vectors are filtered by using a standard Kalman filter. The results show that the ICP algorithm usually produce more accurate results but at a higher computational cost. As the future work we propose to improve our approaches by using also the intensity information as in the optical flow methods.

REFERENCES

[1] 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.

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

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

[4] 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.

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

[6] S. Nedevschi, R. Danescu, T. Marita, F. Oniga, C. Pocol, S. Sobol, C. Tomiuc, C. Vancea, M. M. Meinecke, T. Graf, T. B. To, and M. A. Obojski, "A sensor for urban driving assistance systems based on dense stereovision". In proceedings of Intelligent Vehicles 2007, pp. 278-286, Istanbul, 2007.

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

[8] R. Madhavan, "Terrain aided localization of autonomous vehicles". In proceedings of Symposium on Automation and Robotics in Construction, Gaithersburg, 2002

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

[10] 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

[11] R. Danescu, F. Oniga, and 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.

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

[13] J. M. Hess, "Characterizing dynamic objects in 3d laser range data". Master's thesis, Albert-Ludwigs-Universitt Freiburg, 2008.

[14] M. Fischler and R. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In Communications of the ACM, vol. 24, no. 6, pp. 381– 395, 1981.

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

[16] S. Rusinkiewicz and M. Levoy, "Efficient Variants of the ICP Algorithm". In proceedings of Third International Conference on 3D Digital Imaging and Modeling (3DIM), June 2001.

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

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

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

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

[21] G. Borgefors, "Distance transformations in arbitrary dimensions". In Comput. Vis. Graph. Image Proc. 27, 321–345, 1984.

[22] J.L. Bentley, "Multidimensional binary search trees used for associative searching". In Communications of the ACM. 18(9):509-517, September, 1975.