Tracking Multiple Objects Using Particle Filters and Digital Elevation Maps

Radu Danescu, Florin Oniga, Sergiu Nedevschi, Marc-Michael Meinecke

***Abstract*—Tracking multiple objects has always been a challenge, and is a crucial problem in the field of driving assistance systems. The particle filter-based trackers have the theoretical possibility of tracking multiple hypotheses, but in practice the particles will cluster around the stronger one. This paper proposes a two-level approach to the multiple object tracking problem. One particle filter-based tracker will search the whole state space for new hypotheses, and when a hypothesis becomes strong enough, it will be passed to an individual object tracker, which will track it until the object is lost. The initialization tracker and the individual object trackers use the same state models and the same measurement technique, based on stereovision-generated elevation maps, and differ only in their use of the estimation results. The proposed solution is a simple and robust one, adaptable to different types of object models and to different types of sensors.**

I. Introduction

Tracking the obstacle objects is an essential function of any sensorial system designed for driving assistance. In the complex, real-traffic scenarios, the sensorial system must be able to successfully track multiple objects simultaneously.

Many trackers rely on the mathematical framework of the Kalman filter, which provides a consistent and computation effective way of handling the stages of prediction, association and update. The use of Kalman filter for tracking is described in many works, the most representative being [1].

A newer tracking technique has gained popularity when its use in visual tracking was presented in [2]. The particle filter-based trackers have, in theory, a natural possibility of tracking multiple objects, as the probability density function approximated by the particle set can be multimodal. However, the particles will tend to cluster in time around the strongest hypothesis, and the end result will be a single object track.

In order to handle the problem of multiple objects with a particle filter, the solution presented in [3] models and tracks the configuration of the scene, including the probability of object creation and object deletion. This paper intends to offer an alternative, simpler solution, possibly more computationally efficient and more easily to reuse in different applications.

Manuscript received January 15, 2009. Radu Danescu, Florin Oniga and Sergiu Nedevschi are with the Technical University of Cluj-Napoca, Computer Science Department (e-mail: radu.danescu@cs.utcluj.ro). Department address: Computer Science Department, Str. C. Daicoviciu nr. 15, Cluj-Napoca, Romania. Phone: +40 264 401457

M.M. Meinecke is with Volkswagen AG, Driving Assistance Electronics Department, e-mail: marc-michael.meinecke@volkswagen.de

The versatility of the particle filtering solutions can lead to multiple ways in which the data and the object model are combined, as opposed to the Kalman filter, which usually requires complex detection algorithms, followed by data association. In [2] the image data consists of simple edges, in [7] multiple cues such as symmetry, edge activity and intensities are fused, and in [6] the particles are used even for stereo reconstruction, by maintaining particle sets for both image spaces of the stereo setup.

This paper presents a multiple object tracking scheme that uses a two-level approach. First, a particle filter-based tracker that starts with an initial probability density function that covers the whole possible object parameter state space runs freely on the 3D data, and its particles will tend to cluster around one object. When the clustering is achieved, the particle state is passed to an individual object tracker, that will track the object until it disappears, and the initialization tracker is reset and will start searching for another object.

The proposed solution is a simple and efficient scheme that can be adapted to multiple types of 3D sensors and object models. The implementation described in this paper relies on the elevation map obstacle area discrimination algorithm described in [5]. The use of other sensors that are able to provide a 3D map of the obstacle areas can be made possible with minimal changes.

II. System Overview

The multiple objects tracking system is organized in a distributed, decentralized fashion (fig. 1). There are *N* individual object trackers built around the particle filter mechanism described in [2], and one similar tracker that is used for initialization. All the trackers use the same object model, the same type of particles and the same measurement data.

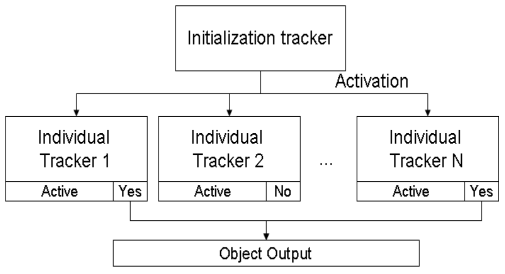


Fig. 1. Top-level architecture of the tracking system

The initialization tracker starts by generating random object hypotheses that cover all the measurement 3D space (this is ensured by providing position and size variation ranges at initialization). The mechanism of the particle filtering will ensure that the hypotheses will eventually group around one object. By monitoring the spread of the particles we can decide when they are grouped enough. At this point, the system takes an inactive individual tracker and passes the particle distribution to it. The initialization tracker is reset to the initial distribution, and the found object is tracked by the recently activated individual object tracker.

Although the individual trackers and the initialization tracker are built around similar particle filters, their behavior is different. Figure 2 shows the flowchart for the initialization tracker. We'll briefly describe each block in the diagram:

Resample – the process that transforms the weighted particle set describing the past probability density into an equivalent set of weightless particles, by random weighted sampling.

Generation of initialization particles – a fraction of the particle set will be sampled from the initial probability distribution, that is, from the whole range of possible object states. These particles help the tracker evade a false hypothesis faster.

Drift – the uniform motion dynamic model is applied to the particles (details in the next section)

Diffusion – a random value is applied to each particle, accounting for the uncertainties of state transition.

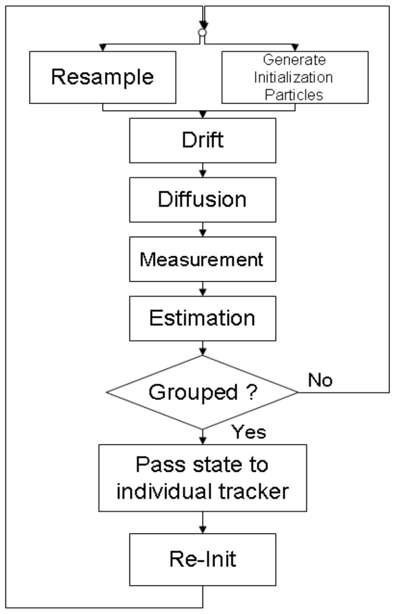
Measurement – each particle is compared to the measurement data, and weighted according to a fitness score. This process will be detailed in section 5.

Estimation – an estimated object state is computed by weighted average of the particle values. This step also computes a standard deviation value for each estimated parameter.

"Grouped" decision - The initialization tracker uses only the standard deviations of the object position to decide whether the particles are grouped. If they are not grouped, the tracker starts another cycle. If they are grouped, the whole particle distribution is passed to an individual tracker, which is made active, and the initialization tracker is reset. In this implementation, the group decision is taken on the whole particle set of the initialization tracker, but significant improvement can be done by using clustering.

The operation of the active individual object tracker is depicted in figure 3. The basic blocks are similar to the initialization tracker, but there are some notable differences. First, there are no initialization particles, as these trackers operate at local level, their target already selected. Second, after estimation the next block is a "valid" condition, which tests the grouping of the particles, the position and size of the estimated object (if grows too big, or too small, or goes out of the field of view, it becomes invalid). If the tracker becomes invalid, it goes into the inactive state, and the tracked object is declared lost.

The active individual tracker also has the job of creating an output object for visualization and transmission to the other driving assistance modules. The output consists of the estimated state parameters and their standard deviation.


Resample
Generate Initialization Particles
Drift
Diffusion
Measurement
Estimation
Grouped ?
No
Yes
Pass state to individual tracker
Re-Init
Fig. 2. Flowchart of operation for the initialization tracker

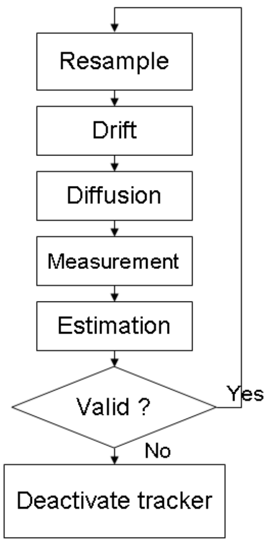


Fig. 3. Flowchart of operation for the individual object tracker

III. Object Model

The initialization tracker, and the individual object trackers, use the same object model and the same method of representing the object state probability. The object state probability density is described at a given time t by a set of N weighted particles $p(\mathbf{x}) \approx \{\mathbf{x}_t^i, \pi_t^i, i = 1...N\}$. The particle value $\mathbf{x}$ is an object state hypothesis, in the form of the following description vector:

$$\mathbf{x}_i = \begin{bmatrix} P_X - \textit{position on axis } X \\ P_Z - \textit{position on axis } Z \\ S_X - \textit{size on axis } X \\ S_Z - \textit{size on axis } Z \\ v_X - \textit{velocity component on } X \\ v_Z - \textit{velocity component on } Z \end{bmatrix}$$

The meaning of the object parameters is depicted clearly in figure 4. The objects are modeled in the bird-eye view space only, without the use of the *Y* (height) coordinate. The origin of the coordinate system is on the road, on the middle axis of the host vehicle's projection on the road plane, in the foremost point of this projection.

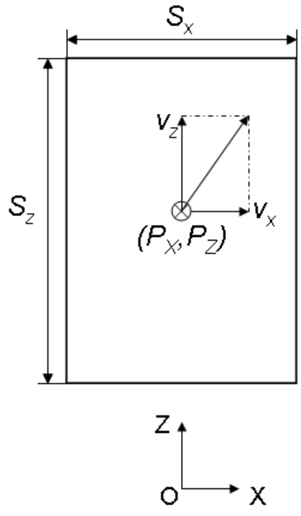


Fig. 4. The parameters of the tracked object

The dynamic object model depicts the way the object state evolves in time, and is expressed by equation 1. The state evolution equation is used in the particle drift and diffusion phase, which both are part of the track state prediction.

$$\bar{\mathbf{x}}_t^i = \mathbf{A}_t \hat{\mathbf{x}}_{t-1}^i + \mathbf{w}_t \tag{1}$$

The deterministic drift is expressed by the matrix multiplication, which, in our case, depicts the uniform motion model for the object center's position, and a static model for the object's size. The matrix $\mathbf{A}_t$ depends on the time elapsed between measurements, Δt.

Besides the deterministic part, each particle's position is altered by a random value $\mathbf{w}_t$, drawn from a Gaussian distribution of zero mean and covariance matrix $\mathbf{Q}_t$. The covariance matrix $\mathbf{Q}_t$ is obtained by scaling a fixed matrix $\mathbf{Q}_0$, calibrated for a time of 100 ms between frames, with the actual elapsed time between measurements (as the frame rate is not fixed).

$$\mathbf{A}_t = \begin{bmatrix} 1 & 0 & 0 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & 0 & 0 & \Delta t \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

IV. Measurement Data

The road and obstacle detection algorithm presented in [5] was employed to compute the binary occupancy grid. This algorithm uses digital elevation maps to represent 3D dense stereo data, in a compact form suitable for real-time processing.

A region of interest (40x13 meters, bird eye-view) of the 3D space is represented as a digital elevation map (DEM). The DEM is a rectangular grid (matrix) of cells and the value of each cell is proportional to the 3D elevation of the highest point (within the cell). The DEM presents poor connectivity between road points at far depths (due to the perspective effect). Connectivity is improved by propagating (to empty cells) the value of valid cells (with 3D points), along the depth (Fig. 5.a). The propagation amount is computed from the stereo geometry of the system, to compensate the perspective effect. Additional features are stored with each DEM cell, such as the density of 3D points per cell.

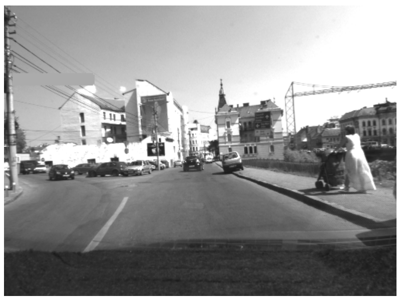
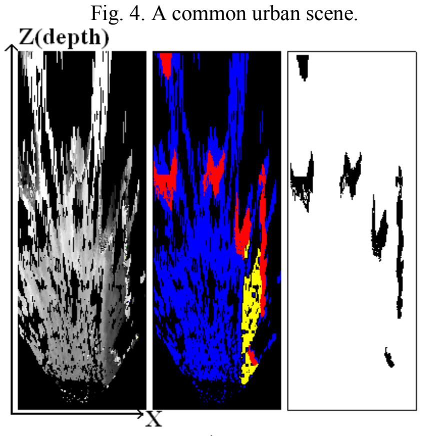


Fig. 4. A common urban scene.



a. b. c.

Fig. 5. The DEM for the scene presented in fig. 4 (a). The result of classification: (b) road cells with blue, traffic isle cells with yellow, and obstacle cells with red. The binary occupancy grid (c) (obstacle cells are black).

A quadratic surface is used to model the road, instead of the less general planar model. A RANSAC (RAndom SAmple Consensus) approach, combined with region growing, is employed to compute the parameters of the surface:

1. The road model is fitted, using RANSAC, to a small rectangular DEM patch in front of the ego vehicle. A primary road surface is extracted optimally for the selected patch.

2. The primary solution is refined through a region growing process, where the initial region is the set of road inliers of the primary surface, from the initial rectangular patch. New cells are added to the region if they are on the region border and they are inliers of the current road surface. The road surface is recomputed (LSQ fitting to the current region) each time its border expands with 1-2 pixels.

The uncertainty of the stereo sensor is modeled and used to discriminate between road inliers and outliers (Fig. 6). Using a height uncertainty increasing with the depth is more reliable (compared to a constant uncertainty strip) and in accordance with the stereo system characteristics [5].

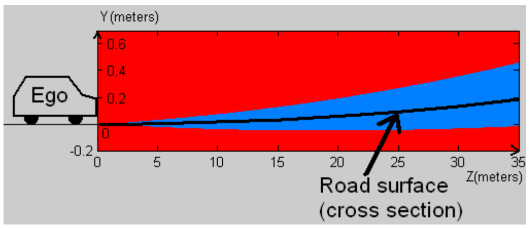


Fig. 6. Lateral view (a slice from 3D with a constant value for the X coordinate) of the inliers (blue) area around a quadratic road.

Next, obstacle / road separation is performed based on the detected road surface. Each DEM cell is labeled as drivable if it is closer to the road surface than its estimated height uncertainty or as non-drivable, otherwise. Small clusters of non-drivable cells are rejected based on criteria related to the density of 3D points. Remaining non-drivable clusters are classified into obstacles or traffic isles, based on the density of 3D points.

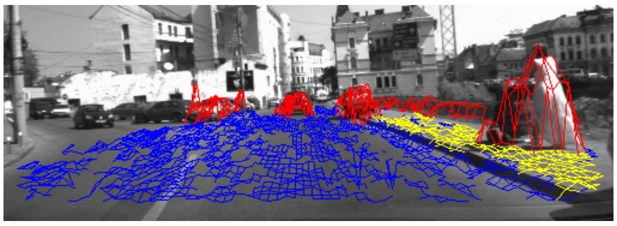


Fig. 7. The classified grid projected back onto the left image. Blue for road, red for obstacles and yellow for traffic isles.

This algorithm outputs the classified grid (fig. 5.b and fig. 7) with three distinct cell types: road, traffic isles and obstacles. The binary occupancy grid needed for tracking is obtained straightforward (fig. 5.c) by taking into account only the obstacle cells of the classified grid.

V. Particle Weighting by Measurement

The elevation map processing provides us with a binary, discrete top-view map of the 3D scene, where each obstacle cell has the value of zero (black), and each free space cell the value of 255 (white). Our aim is to compare each particle (which represents an object hypothesis) to the measurement data, and weight each one according to the quality of the match.

The first step is to map the particle state into the measurement space, a task that is not very difficult, due to the fact that the measurement space is just a discrete, scaled-down representation of the 3D space.

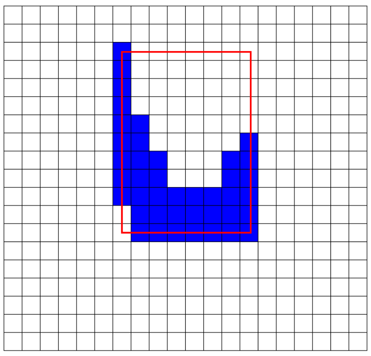


Fig. 8. An object hypothesis versus the measurement data

For each corner of the object hypothesis projection in the measurement data space, we select six points to be compared with the raw data. Three points are selected from the projected contour – they will be called the inner points (the black points in figure 9). Another three points are selected from a perimeter outside the object hypothesis projection – these points will be called the outer points (the gray points in figure 9). The perfect match between an object hypothesis and the measurement data would be that the inner points fall on the obstacle areas in the raw data map, and the outer points will fall on free areas.

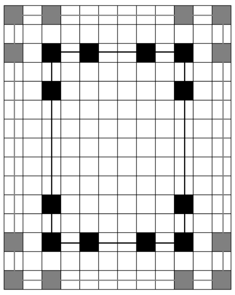


Fig. 9. The inner and outer corner points of a hypothesis representation in the data space

In order to compute the overall matching score, for each point $P_{i,k}^{T}$ (the value of i is from 1 to 4, the number of corners, and the value of k is from 1 to 3, the number of points in a corner, and T is the point type, I for inner, and O for outer) we'll assign the value 0 if the point falls on an obstacle cell in the measurement grid, and 255 if the point falls on a free cell. Then, we can compute the following values for each corner:

$$I_i = \frac{1}{3}\sum_{k=1}^{3} P_{i,k}^{I} \tag{3}$$

$$O_i = \frac{1}{3}\sum_{k=1}^{3} P_{i,k}^{I} \tag{4}$$

The distance measure for each corner is defined by the equation 5:

$$D_i^2 = I_i^2 + (255 - O_i)^2 \tag{5}$$

The final (square) fitness measure is given by the average of all the square corner fitness values:

$$D^2 = \frac{1}{4}\sum_{i=1}^{4} D_i^2 \tag{6}$$

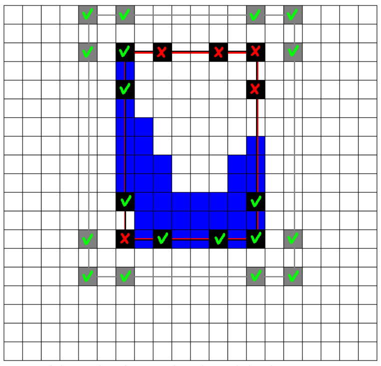


Fig. 10. Matching the hypothesis with the measurement data

An example of corner fitting between the object hypothesis and the measurement is shown in figure 10. The squared fitness values for each corner, starting from the top left and going counter-clockwise, are: 7225, 7225, 0, 65025. The squared fitness measure for the whole object is 19868.75.

Now that the fitness measure (or fitness distance) is computed, the particle may receive its weight using the Gaussian equation:

$$\pi = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{D^2}{2\sigma^2}} \tag{7}$$

Depending on the object's position in the field of view of the sensors (camera, in this case) not all the object's corners are visible. In fact, at most three of the four object corners can be visible at a given instant. The visible corners are decided before the measurement, based on the position of the hypothesis: if the position is central to the observer's field of view, only the nearest two corners are considered, and if the object is found in a lateral position, the observable far corner is added to the list. This reasoning was also implemented in a Kalman filter-based solution, described in [4]. In computing the matching distance, only the visible corners are taken into consideration. For our example, if we consider that the object is to our right, the matching distance gets significantly lower if the far right corner is disregarded.

When managing the measurement data, we have to ensure that the initialization tracker does not cluster around the data already covered by the individual object trackers. This situation is prevented by measurement data map masking. After all the active individual object trackers have completed their tracking cycle, the data map is masked by the areas covered by the individual trackers, effectively removing the data from the set. This way, the initialization tracker will work only on data not covered by the active individual trackers. This step is a form of data association, which only separates initialization data from update data. More complex data association schemes are not required, due to the nature of the particle filter-based solution.

VI. Tests and Results

The proposed tracking technique has been tested in real traffic situations, where the presence of multiple relevant objects is guaranteed. This experimental technique was compared with another tracking system, based on 3D feature grouping into cuboids and their subsequent tracking using the Kalman filter [4], and the estimated position, size (except for the height which is not estimated here) and speed are consistent between the two trackers. The advantage of the technique described in this paper is that it does not require any 3D box reconstruction process as input data, being capable of working directly on raw obstacle data, using a simple measurement metric. The time performance depends on the complexity of the scene to be tracked, taking up to 30 ms per frame, but significant speedup can be achieved by optimizations.

In the next figures we'll show the intermediate steps and the final results for a test sequence, from which we have sampled three significant frames taken approximately three seconds (50 frames) apart (fig. 11). The 3D obstacle data contains errors and uncertainties that are unavoidable in the case of a stereovision sensor (fig. 12).

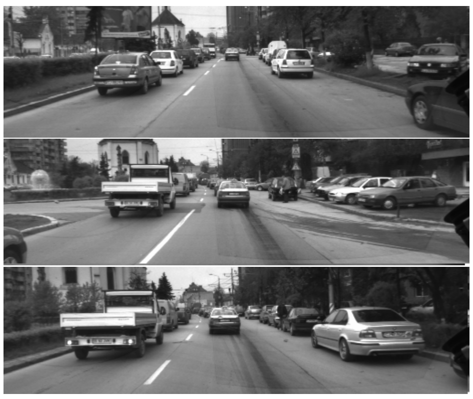


Fig. 11. Samples from a test sequence, approximately three seconds apart

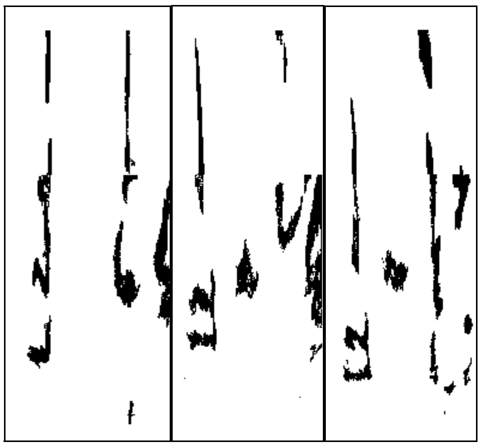


Fig. 12. Measurement data from the sample frames

In figure 13 we see the behavior of the particles of the initialization tracker. This tracker will have a discontinuous behavior, towards clustering around a single hypothesis. When the particles are clustered enough, an individual tracker is activated, and the initialization tracker's particles become spread again.

Figure 14 shows the behavior of the individual object trackers. Initially, there are no active trackers, but after a while they tend to cover the entire scene with their particle clusters. Each particle cluster is then used to estimate an object, the final results being shown in figure 15, in perspective projection for comparison with the original image.

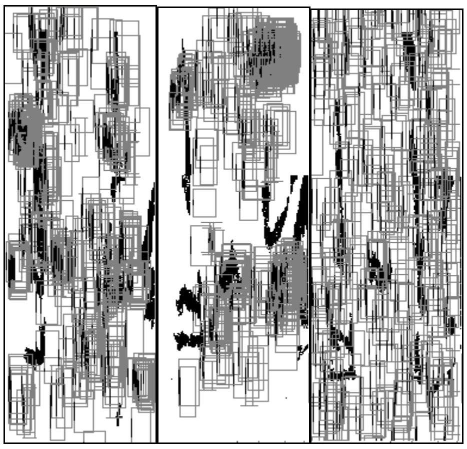


Fig. 13. Initialization tracker particles

VII. Conclusion and Future Work

We have described a tracking technique for multiple objects, based on raw obstacle 3D data that can be adapted to multiple types of objects and sensors. The main issue that will be covered by the future work is speed of initialization, which is not as fast as possible due to the free mode of operation of the initialization tracker (a simple focusing system based on occupied areas may speed up the convergence significantly), and due to the fact that only one object can be initialized at one time, a problem that can be easily solved using a clustering algorithm on the initialization tracker's particles.

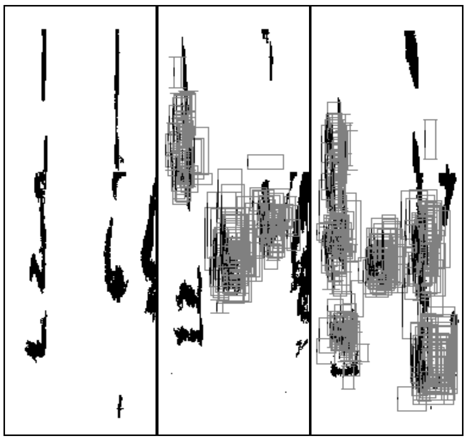


Fig. 14. Individual object tracker particles

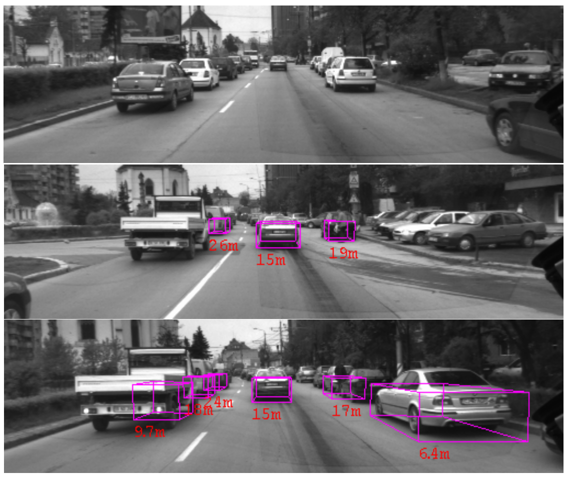


Fig. 15. Estimated objects, in perspective projection

References

[1] Y. Bar-Shalom, T.E. Fortmann, "Tracking and Data Association", *Academic Press Inc., 1988.*

[2] M. Isard, A. Blake, "CONDENSATION -- conditional density propagation for visual tracking", *Int. J. Computer Vision, 29, 1, 5--28, (1998)*

[3] H. Tao, S. S. Sawhney, R. Kumar, "A Sampling Algorithm for Tracking Multiple Objects", *Vision Algorithms: Theory and Practice, International Workshop on Vision Algorithms, held during ICCV '99, Corfu, Greece, September 21-22, 1999*

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

[5] F. Oniga, S. Nedevschi, M-M. Meinecke, T-B. To, "Road Surface and Obstacle Detection Based on Elevation Maps from Dense Stereo", *The 10th International IEEE Conference on Intelligent Transportation Systems, Sept. 30 - Oct. 3, 2007, Seattle, Washington, USA.*

[6] A S. Sabbi, M. Huber,"Particle Filter Based Object Tracking in a Stereo Vision System", in proc. of *the 2006 IEEE International Conference on Robotics and Automation* (ICRA), Orlando, Florida - May 2006

[7] Y. M. Chan, S. S. Huang, L. C. Fu, P. Y. Hsiao, "Vehicle Detection under Various Lighting Conditions by Incorporating Particle Filter", *in proc. of the IEEE Intelligent Transportation Systems Conference (ITSC 2007).*