Particle Grid Tracking System for Stereovision Based Environment Perception

Radu Danescu, Florin Oniga, Sergiu Nedevschi

Abstract— **This paper presents an occupancy grid tracking solution based on particles. The particles will have a dual nature – they will denote hypotheses, as in the particle filtering algorithms, but they will also be the building blocks of our modeled world. The particles have position and speed, and they can migrate in the grid from cell to cell depending on their motion model and motion parameters, but they will also be created and destroyed using a weighting-resampling mechanism specific to particle filter algorithms. An obstacle grid derived from processing a stereovision-generated elevation map is used as measurement information, and the measurement model takes into account the uncertainties of the stereo reconstruction. The resulted system is a flexible, real-time tracking solution for dynamic unstructured driving environments.**

I. Introduction

When the observed environment is highly structured, and its components have a standard geometry, model-based tracking is the natural choice to be employed. The obstacles can be modeled as cuboids having position, size and speed, and the driving surface delimiters can be modeled as parametrical curves. The highway and most of the urban and rural sections of road can be regarded as structured environments.

The conditions change when the environment to be tracked is an intersection, a busy urban center, or an off-road scenario. Even if parts of this environment can be tracked under the structured assumption, many essential parts of the environment will not fulfill the model. For these situations, the natural solution is to use occupancy grids.

An occupancy grid is a probabilistic map of the driving environment, which encodes the present and past knowledge available from processing the sensor data, and which can be dynamically updated when new information becomes available.

A Bayesian occupancy filter [1] is a probability based tracker for the unstructured environments that are best modeled as occupancy grids.

The occupancy grids can be static, encoding only the occupancy probability, or dynamic, adding speed information for each cell. Three types of static occupancy grids are presented in [3], and these types are used for environment tracking based on stereo measurements: Cartesian (rectangular cell), column/disparity, and polar. Dynamic programming is used to compute the free space ahead of the vehicle, and the ego-motion of the camera is compensated by the integration of motion sensor information.

A dynamic occupancy grid is used in [2], where the speed of each cell is modeled as a distribution of possible values, and the tracker computes the probability of each value, along with the occupancy probability of the grid cell. The Bayesian reasoning process uses a set of possible antecedents, which are the cells that can influence the current cell based on the speed hypotheses, and the probabilities of the antecedents are combined with the sensor information. Another solution, presented in [9], handles the problem of moving objects by identifying occupancy variations ("trails") along an object's path, thus removing the need for antecedents.

In [4] the dynamic Bayesian occupancy filter solution introduced in [2] is combined with the use of map information, which guides the hypotheses of the cell speeds to the allowed trajectories on the road. This solution prevents the system to consider unreachable positions, and enables it to better predict the vehicle paths when the road is curved.

The occupancy grid model of the world is well suited for collaborative updating, using the information from multiple sensors or multiple observers. A solution that integrates the observations of multiple mobile observers into a unified description of the environment is presented in [5].

This paper presents an occupancy grid tracking solution based on particles. The particles will have a dual nature – they will denote hypotheses, as in the particle filtering algorithms such as CONDENSATION [6], but they will also be the building blocks of our modeled world. The tracking algorithm described in this paper will be particle-oriented, not cell oriented. The particles have position and speed, and they can migrate from cell to cell depending on their motion model and motion parameters, but they will also be created and destroyed using the same logic as the weighting-resampling mechanism described in [6]. The measurement data is the raw obstacle grid obtained by processing the elevation map, as described in [7], a measurement source which we have previously used for model-based object tracking, a technique described in [8].

The tracking solution presented in this paper has the benefit of simple integration of motion and measurement models, provides an easy mechanism for introducing

Manuscript received February 15, 2010. This work was supported by CNCSIS –UEFISCSU, project number PNII – IDEI 1522/2008.

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.

additional constraints or information, and, by controlling the number of particles, allows the user to reach a tradeoff between accuracy and time performance.

II. The World Model

The world is represented by a 2D grid, mapping the bird-eye view 3D space into discrete 10 cm x 10 cm cells. The size of the grid is 400 rows x 128 columns (this corresponds to a scene size of 40x12.8 meters). Like in other grid tracking solutions, the aim is to estimate the occupancy probability of each grid cell, and the speed components on each axis. However, these values are not key concepts in the workings of the algorithm that will be proposed in this paper, but they will be derived from a particle-based tracking mechanism.

Considering a coordinate system where the z axis points towards the direction of the ego-vehicle, and the x axis points to the right, the obstacles in the world model are represented by a set of particles $S=\{p_i \mid p_i=(x_i, z_i, vx_i, vz_i), i=1...N_S\}$, each particle i having a position in the grid, described by the row z_i (a discrete value of the distance in the 3D world) and the column x_i (discrete value of the lateral position), and a speed, described by the speed components vx_i and vz_i. The total number of particles N_S is not fixed, but depends on the number of obstacles in the scene. Having the population of particles in place, the occupancy probability of a cell C is the ratio between the number of particles whose position coincides with the position of the cell C and the total number of particles allowed for a single cell, N_C.

$$P_O(C)=\frac{|\{p_i \in S \mid x_i=x_c, z_i=z_c\}|}{N_C} \tag{1}$$

The number of allowed particles per cell N_C is a constant of the system. In setting its value, a tradeoff between accuracy and time performance should be considered. A large number means that on a single cell multiple speed hypotheses can be maintained, and therefore the tracker can have a better speed estimation, and can handle fast moving objects better. However, the total number of particles in the scene will be directly proportional with N_C, and therefore the speed of the algorithm will decrease.

The speed estimation of a grid cell is the average speed of its associated particles.

$$(vx_C, vz_C)=\frac{\sum\limits_{p_i\in S, x_i=x_c, z_i=z_c}(vx_i, vz_i)}{|\{p_i \in S \mid x_i=x_c, z_i=z_c\}|} \tag{2}$$

Thus, the population of particles is sufficiently representative for the probability density of occupancy and speed for the whole grid. Multiple speed hypotheses can be maintained simultaneously for a single cell, and the occupancy uncertainty is represented by the varying number of particles associated to the cell. The goal of the tracking algorithm can now be stated: using the measurement information to create, update and destroy particles such that they accurately represent the real world.

III. Prediction

This step will derive the present particle distribution from the past information, preparing the particle set for measurement. The prediction equations will use odometry and motion model information.

The basic odometry information available through the CAN bus of a modern car is the speed v and the yaw rate $\dot{\psi}$. Together with the time interval Δt elapsed between measurements, these parameters can be used to compensate for the ego-motion, and separate it from the independent motion of the objects in the scene. Between measurements, the ego-vehicle rotates with an angle ψ, and travels a distance d.

$$\psi=\dot{\psi}\Delta t \tag{3}$$

$$d=\frac{2v\Delta t \sin\frac{\psi}{2}}{\psi} \tag{4}$$

The origin of the grid representation is displaced along the two coordinate axes by d_x and d_z.

$$d_x = d\cos\psi / DX \tag{5}$$

$$d_z = d\sin\psi / DZ \tag{6}$$

We denote by DX and DZ the cell size of the grid. A point in the grid is displaced by the following equation:

$$\begin{bmatrix} x_n \\ z_n \end{bmatrix}=\begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix}\begin{bmatrix} x \\ z \end{bmatrix}-\begin{bmatrix} d_x \\ d_z \end{bmatrix} \tag{7}$$

The prediction is achieved using equation 8, which combines the deterministic drift caused by the ego-motion compensation and the particle's own speed, with the stochastic diffusion caused by the uncertainties in the motion model. The quantities δx, δz, δvx and δvz are randomly drawn from a Gaussian distribution of zero mean and a covariance matrix **Q** equivalent to the state transition covariance matrix of a Kalman filter.

$$\begin{bmatrix} x \\ z \\ v_x \\ v_z \end{bmatrix}=\begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x_n \\ z_n \\ v_x \\ v_z \end{bmatrix}+\begin{bmatrix} \delta x \\ \delta z \\ \delta vx \\ \delta vz \end{bmatrix} \tag{8}$$

From the grid model point of view, the prediction has the effect of moving particles from one cell to another, as seen in figure 1. The occupancy probability is thus dynamically adjusted using the particle's motion model and the vehicle odometry.

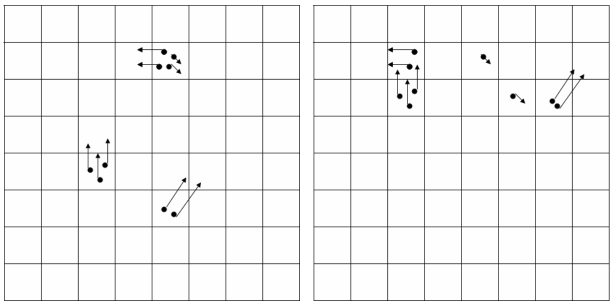

Fig. 1. Particles in the grid, before and after prediction

IV. MEASUREMENT

The classical steps of a particle filter based tracker are resampling, drift, diffusion, and measurement (weighting). This behavior replaces a population of a fixed number of particles with an equal number of particles, which approximates an updated probability density function over a space of parameters. However, this approach works when the particles are hypotheses of the state of a system, not when the particles are the system itself (we can see our tracked world as physically composed of particles).

Our algorithm tries to use the particles in a dual form – as hypotheses, and as building blocks of the world that we track. Their role as building blocks has been already explained. However, if we restrict our reasoning to a single cell in the grid world, we can see *that the particle is also a hypothesis*. A particle in a grid cell is a hypothesis that this cell is occupied, and that the cell has the speed equal to the speed of the particle. More particles in the cell mean that the hypothesis of occupancy is strongly supported. Less particles in the cell means that the hypothesis of the cell being free is supported. We can regard the difference between the number of particles in a cell and the total number of particles allowed in a cell as the number of particles having the occupancy hypothesis zero.

A. Weighting the particles

If we regard the number of particles in the cell to be constant, and some of them having the occupancy value "true" while some having it "false", we can apply the mechanism of weighting and resampling.

If we assume that the measurement data does not contain speed information, the weight of the particle depends only on the "occupied" hypothesis. Also, this means that all the particles having the same occupied hypothesis will have the same weight.

$$w_{occupied} = p(measurement \mid occupied) \quad (9)$$

$$w_{free} = p(measurement \mid free) \quad (10)$$

The computation of *p(measurement|occupied)* and *p(measurement|free)* is detailed in section V.

The number of particles having the "occupied" hypothesis true is the number of "real" particles in the cell.

$$N_{OC} = \left| \{ p_i \in S \mid x_i = x_c, z_i = z_c \} \right| \quad (11)$$

The number of particles (hypotheses) having the "occupied" value false is the complement of N_{OC}.

$$N_{FC} = N_C - N_{OC} \quad (12)$$

The total posterior probability of a cell being occupied and of a cell being free can be computed from the number of free/occupied hypotheses, and their corresponding weights:

$$P_{OC} = \frac{w_{occupied} N_{OC}}{w_{occupied} N_{OC} + w_{free}(N_C - N_{OC})} \quad (13)$$

$$P_{FC} = \frac{w_{free}(N_C - N_{OC})}{w_{occupied} N_{OC} + w_{free}(N_C - N_{OC})} \quad (14)$$

The aggregate particle weights P_{OC} and P_{FC} are used for particle resampling. The resampling of the particle population is done at the end of the measurement step, so that the next cycle can start again with an updated population of particles without concerning about their weight.

B. Resampling

The classical resampling makes N_C random draws from the previous particle population of a cell, and the weight of each particle controls its chances of being selected. Because we don't care for the "cell free" hypothesis particles, our resampling will instead decide for each real particle (particle having the occupied hypothesis true) whether it is destroyed or multiplied (and, if multiplied, how many copies of it are created).

The following algorithm describes the process of resampling, which is materialized as duplication or removal of particles from the particle set.

Algorithm Resample

For each cell C

Compute N_{OC} and P_{OC}

Compute resampled number of particles N_{RC}

$N_{RC}=P_{OC}N_C$

Compute ratio between actual number of particles and the number of resampled particles

$$f_C = \frac{N_{RC}}{N_{OC}}$$

End For

For each particle p_i
  Find corresponding cell C
  If (f_C>1) – *number of particles will increase*
    F_n = Int(f_C)   Integer part
    $F_f = f_C$ -Int(f_C)   Fractional part
    For k=1 **to** F_{n-1}
      S.Add(p_i.MakeCopy)
    End For
    r = random value between 0 and 1
    If ($r<F_f$)
      S.Add(p_i.MakeCopy)
    End if
  End if

  If (f_C <1) – *number of particles will decrease*
    r = random value between 0 and 1
    If ($r> f_C$)
      S.Remove(p_i)
    End if
  End if

End For

The system will compute the number of particles that each cell should have after the process of resampling has been completed. The ratio f_C between this number and the existing number of particles in the cell will tell us if the particles have to be duplicated or removed. If f_C is higher that 1, the number of particles has to be increased. The integer part of the difference between f_C and 1 tells us the number of certain duplications a particle must undergo (for instance, if f_C is 2, each particle will be doubled). The fractional part of the difference is used for chance duplication: each particle will have a probability of being duplicated equal to the fractional part of this difference.

If f is lower than 1, the number of particles has to be decreased, by removing some of the particles. Each particle has 1- f_C chances of being eliminated.

At this point the cycle is complete, and the tracking algorithm can process a new frame. Secondary estimations for occupancy, speed, or clustering the cells into objects can be performed at the end of this step.

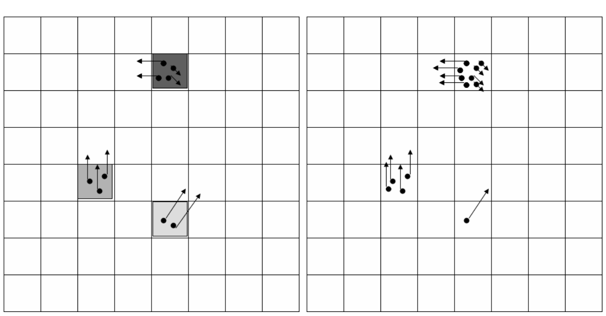

Fig. 2. Weighting and resampling. The weight of the occupied hypothesis is encoded in the darkness of the cell of the left grid.

V. MEASUREMENT MODEL

The measurement model will relate the measurement data, which is a binary occupied/free condition derived from the stereovision-generated elevation map [7], to the conditional probabilities *p(measurement | occupied)* and *p(measurement | free)*, probabilities that will weight the real and the virtual particles presented in the previous section.

In order to compute these probabilities, we start by computing the uncertainty of the stereo reconstruction. First, the uncertainty of the distance reconstruction, in the case of a rectified system, is given by:

$$\sigma_z = \frac{z^2 \sigma_d}{bf} \tag{15}$$

In the above equation, z denotes the distance, b is the baseline of the stereo system, f is the focal distance in pixels, and σ_d is the error in disparity computation (usually about 0.25 pixels, for a good stereo reconstruction engine).

The error in lateral positioning (usually much smaller than the error in z), can be derived from the distance error:

$$\sigma_x = \frac{x\sigma_z}{z} \tag{16}$$

The 3D errors are mapped into grid cell errors, by dividing them with the grid cell size on x and z.

$$\begin{aligned} \sigma_{grid\_z} &= \frac{\sigma_z}{DZ} \\ \sigma_{grid\_x} &= \frac{\sigma_x}{DX} \end{aligned} \tag{17}$$

In order to compute the conditional probability of the measurement cell, under the occupied or free assumption, we have to take into account a reality that is specific to stereovision sensors. The stereo sensor does not perform a scan of the scene, and therefore it does not output a single bird-eye view point for a real-world obstacle cell. We'll take as example a pillar, which has almost no width, and no depth spread. The representation of a pillar in the occupancy grid should be a single cell. If the pillar were observed by a scanner-type sensor, this sensor will output a cell, displaced from the true position by an amount specific to the sensor error. For the stereo sensor, things are different, because the camera observes the whole height of the pillar, and therefore each pillar pixel will get a distance and a lateral position. This means that once we "collapse" the pillar information in the 2D grid representation, each part of the pillar may fall in a different cell, and the pillar will generate a spread of cells. The size of the spread area is controlled by the grid uncertainties on the x and z axes.

This property leads us to find a reasonable approximation for the conditional probabilities of the measurement cells under the occupied/free assumption. We'll count the obstacle cells in the measurement grid around the current cell position, in an area of σ_{grid_z} height and σ_{grid_x} width, and divide the number of found obstacle cells by the total number of cells in the uncertainty area. This ratio will be our approximation for *p(measurement | occupied).*

$$p(m(x,z)\,|\,occupied) = \frac{\sum_{row=z-\sigma_{z\_grid}}^{row=z+\sigma_{z\_grid}} \sum_{col=x-\sigma_{x\_grid}}^{col=x+\sigma_{x\_grid}} O(row,col)}{(2\sigma_{z\_grid}+1)(2\sigma_{x\_grid}+1)} \tag{18}$$

By *O(row, col)* we denote the "occupied" value of the measurement grid, at position *row* and *col*. This value is 1 when an obstacle cell is present and 0 when not.

The conditional probability of the measurement given the "free" assumption is:

$$p(m(x,z)\,|\,free) = 1 - p(m(x,y)\,|\,occupied) \tag{19}$$

These conditional probability values will be used to weight the particles. A graphic comparison between the raw measurement data and the conditional probability of the measurement under the "occupied" assumption is given in the following figure.

Fig. 3. From the occupancy grid to the particle weights. The bottom-right image encodes the weights of the occupied hypothesis.

VI. Initialization

Although the measurement step takes care of particle creation and deletion, this step only works if there are particles to be duplicated or deleted. For the prediction-measurement cycle to work, the particle population has to be initialized.

From a strictly probabilistic point of view, each cell's state is unknown at startup, which means that the cell has equal probability of being occupied or free. In our tracking system, this would mean that each cell should be assigned a number of particles equal to half the total number of particles allowable in a cell. However, this approach would significantly reduce the speed of the system, and would require permanent re-initialization.

Our solution is to use the measurement occupancy grid to create particles. If a measurement cell is of type obstacle, its *p(measurement | occupied)* is high, and there are no particles in the corresponding tracked grid cell, a small number of particles will be created. The initial speed components *vx* and *vz* of the created particles will be sampled randomly from an initial range of possible values, and the initial position is confined to the creation cell. In this way, the initialization is a continuous process.

Particles are automatically removed when they go outside the grid area, in the prediction phase. Another case of "administrative" removal (removal not caused by the probability mechanism described in section IV) is when, due to particle drifting, the number of particles in a cell exceeds the allowed value.

VII. Tests and Results

We have designed two types of tests in order to validate the tracking algorithm: qualitative tests and quantitative (numerical) tests.

The qualitative assessment proves that the system is capable of building an occupancy probability grid from the measurement data, and is capable of identifying the motion associated with the cells in the grid. The qualitative assessment has been performed on real traffic scenes.

Some results, from two traffic scenes, are shown in figure 4. In the top row, the perspective image of the scene is given, and in the bottom row the three panels are, respectively: the measurement data (obstacle cells computed from the elevation map), the occupancy probability, and the speed labels, identifying the motion of the cells. In the left scene, we can identify two incoming vehicles (red), one outgoing vehicle (blue), and a stationary parked vehicle (black). In the right scene, we can identify a stationary parked vehicle, an outgoing vehicle, the stationary traffic sign on the isle, and the pedestrian group with a combination of stationary and lateral motion.

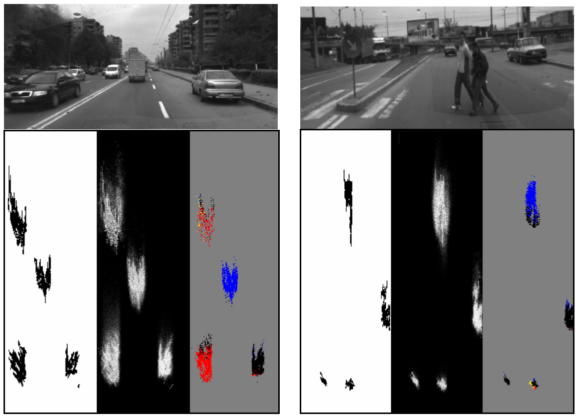

Fig. 4. Qualitative assessment of the algorithm performance. Speed labels: black – stationary, red – incoming, blue – outgoing, yellow – lateral motion.

For numerical evaluation, we have chosen a “follow the leader” scenario, with only one obstacle in the scene, so that a reasonable estimation of the object’s speed can be done in the absence of a cell clustering algorithm. The ego-vehicle is following the target vehicle, matching its speed. Therefore, the speed of the ego-vehicle is a benchmark for the estimated speed of the target. Figure 5 shows that after an initial lag (10 frames, 0.5 seconds), the estimated speed converges to the ego-speed. The absolute mean error after the lag period is 1 km/h.

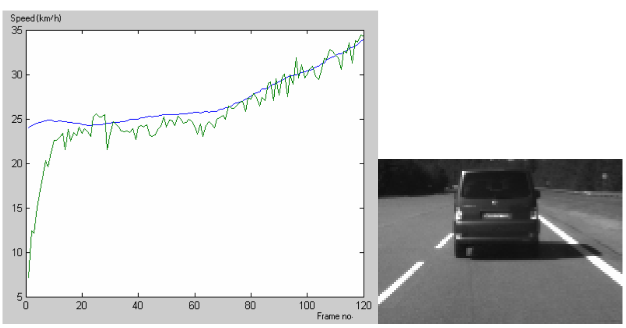

Fig. 5. Left – Ego-speed (blue) versus the estimated target speed (green), in km/h. Right – image of the target vehicle.

A second test implies a static object, observed from a moving vehicle traveling along a circular path. The circular path increases the difficulty of the estimation, due to the fact that the static object is in the field of view for only 3.5 seconds. The results of the speed estimation are shown in Figure 6, against the speed of the observing vehicle. The static nature of the object can be inferred almost immediately.

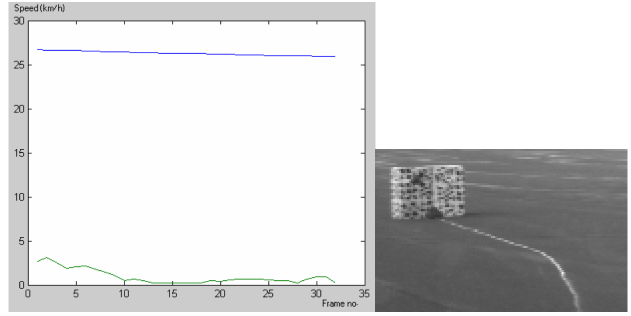

Fig. 6. Ego-speed (blue) versus the static object’s estimated, in km/h. Right – the static object.

The time performance depends on the obstacle load of the scene, which influences the total number of particles. For a typical urban scene, and a total number of particles in a cell of 50, the total running time is about 40 ms per frame.

VIII. Conclusion and Future Work

We have presented a grid tracking technique that models and tracks the driving environment using a set of particles with position and speed. Our solution proves capable of identifying occupancy and motion in unstructured traffic scenes, without the need of feature grouping or obstacle model matching or data association. Future experiments with the grid size, speed and position uncertainties of prediction, and a refinement of the measurement model to include the error of the occupant cell extraction besides the uncertainties of the stereo algorithm, will allow us to optimize this system’s performance and accuracy.

References

[1] C. Coue, C.Pradalier, C.Laugier, T.Fraichard, P.Bessiere, “Bayesian Occupancy Filtering for Multitarget Tracking: An Automotive Application”, *The International Journal of Robotics Research*, 25(1):19, 2006.

[2] C. Chen, C. Tay, K. Mekhnacha, C. Laugier, “Dynamic environment modeling with gridmap: a multiple-object tracking application”, in proc of *International Conference on Automation, Robotics and Computer Vision (ICARCV) 2006*.

[3] H. Badino, U. Franke, R. Mester, “Free Space Computation Using Stochastic Occupancy Grids and Dynamic Programming”, *Workshop on Dynamical Vision, ICCV, 2007*.

[4] T. Gindele, S. Brechtel, J. Scrhoeder, R. Dillmann, “Bayesian Occupancy Grid Filter for Dynamic Environments Using Prior Map Knowledge”, in proc of *IEEE Intelligent Vehicles Symposium 2009*.

[5] J. Y. Chen, J. Hu, “Probabilistic Map Building by Coordinated Mobile Sensors”, in proc of *IEEE International Conference on Networking, Sensing and Control 2004*.

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

[7] F. Oniga, S. Nedevschi, M-M. Meinecke, T-B. To, “Road Surface and Obstacle Detection Based on Elevation Maps from Dense Stereo”, *IEEE Conference on Intelligent Transportation Systems, 2007.*

[8] R. Danescu, F. Oniga, S. Nedevschi, M.-M. Meinecke, “Tracking Multiple Objects Using Particle Filters and Digital Elevation Maps”, in proc of *IEEE Intelligent Vehicles Symposium 2009*.

[9] T. Weiss, B. Schiele, K. Dietmayer, “Robust Driving Path Detection in Urban and Highway Scenarios Using a Laser Scanner and Online Occupancy Grids”, in proc of *IEEE Intelligent Vehicles Symposium 2007.*