A cross-prediction, hidden-state-augmented approach for Dynamic Occupancy Grid filtering

Lukas Rummelhard, Jean-Alix David, Andres Gonzalez Moreno, Christian Laugier

A cross-prediction, hidden-state-augmented approach for Dynamic Occupancy Grid filtering

Lukas Rummelhard, Jean-Alix David, Andres Gonzalez Moreno and Christian Laugier
Univ. Grenoble Alpes, Inria, F-38000 Grenoble, France. Email: name.surname@inria.fr

***Abstract*— Accurate modeling of complex dynamic environments is a fundamental requirement in robotics and automotive applications. While grid-mapping approaches used to be limited to static settings, methods for dynamic occupancy grids have recently been developed, tracking spatial occupancy at a sub-object level, in every cell. In this paper, we present a generic dynamic occupancy grid tracker, which filters cell states and infers dynamics of the scene through the interaction of a grid-based and a particle-based model. These are set to represent different parts of the scene, and optimize particle allocation only to relevant areas, their predictions being fused accordingly. New hidden variables in the filtering process permit to address previously mishandled situations, like concurrent state predictions or specific filtering sensitivity. The presented method has been implemented, optimized on a GPU and tested on real-road conditions, embedded on an experimental vehicle.**

I. Introduction

Despite impressive developments over the last decades, and always-increasing embedded intelligence on mobile devices, the ability for an autonomous agent to accurately, robustly and efficiently perceive its surroundings remains a vivid challenge in robotics. The quality of environment modeling relies on the sensors, but also on the interpretation schemes, managing sensor errors, occlusions, contradictions, heavily complex settings. Many probabilistic methods have been designed to formally model uncertainties and prior knowledge.

When confronted to moving objects, additional issues are raised[1]. A classic way to address them is to adopt an object-based representation, leading to multiple-object tracking literature[2], [3]. Another common approach is to use occupancy grids[4], [5], which work on spatial occupancy without higher-level segmentation. This approach presents significant advantages, like a spatially dense model with proper representation of free space, which is important in mobile robotics. Furthermore the critical data-segmentation and recognition steps can be avoided. When applied to dynamic environments, enriching this representation by velocity estimation is usually necessary.

Spatial occupancy can be modeled through continuous representations, such as Gaussian processes[6], [7], or Hilbert spaces[8], [9]. Other approaches represent the environment as a discretized space, some recent approaches working on Neural Networks, training them to predict the states of the cells in the grid using a combination of different sensors inputs[10], [11], [12]. In the context of embedded devices, rule-based Bayesian methods for data fusion and filtering are used for their compactness and explainability. The approach proposed in this paper fits in this category.

II. Related Works

The Bayesian Occupancy Filter (BOF)[13] describes a generic approach for data filtering and motion inference, defining a Bayesian framework which updates a dynamic occupancy grid by filtering occupancy and velocity in each cell in parallel. The observation model is incorporated as an instantaneous occupancy grid generated by mapping the sensor data over a grid using a probabilistic inverse sensor model[14]. A Bayesian filtering methodology, based on a prediction-correction loop, is used to filter the distributions in each cell, using neighbourhood transition histograms. This motion discretization design may seem convenient, but leads to high computational costs and aliasing issues.

Following the BOF approach, many works have expanded the Bayesian occupancy grid methodology. Prior map knowledge have been integrated[15], sampling strategies have been redefined to focus on the occupied areas of the grid, significantly improving performances[16]. To further expand performances and address aliasing problems, a per-cell adaptive particle sampling density was proposed[17]. In contrast to particle-only representations of the occupancy, a model based on a static-dynamic duality[18] has been proposed, where the static part is described as an occupancy grid, while the dynamic field as moving particles. Both parts are jointly evaluated and updated, focusing the particles on the dynamic components, resulting in a more compact model and improved performances[19]. But as the model is not defined to distinguish informative from data-less areas, many particles are still allocated to irrelevant areas.

By applying Dempster-Shafer evidence theory to occupancy grids[20][21], a robust framework to incorporate estimation confidence can be defined, allowing for such characterization. In a Bayesian line of work, data-less non-informative areas can be modeled through the introduction of a specific state[22], permitting proper handling of operations such as particle initialization or state transitions, while enabling particle focus on informative dynamic areas. However, limited by a stiff connection between static and dynamic parts, prediction and filtering sensitivity could lead to limitations and errors, in particular in case of conflicting predictions from the models or miscalibrated sensor models.

In this paper, we propose a new Bayesian filter to address these limitations, introducing new hidden variables to decouple the prediction models of the static and dynamic parts and parameterized fusion strategies, resulting in a more detailed definition of the state filter dynamics.

III. Filter Definition

A. Objectives

Based on the CMCDOT filter [22], the proposed filter aims to address two of its main limitations :

- prediction bias : in previous models, dynamic and static parts of the environment are modeled and predicted separately (particle versus grid), but their contribution to the state prediction were then combined indistinctly, as potential cell antecedent possibility was considered iteratively from both models. When dynamic objects are projected in a previously empty cell, this method leads to an averaged state occupancy prediction, which systematically induces occupancy loss in predictions (Fig. 1), and thus quick loss of tracking when no new observations are available to re-establish the estimation (outside of the normal temporal evolution of the distribution). By separating grid-based ($S_g^p V_g^p$) and particle-based ($S_p^p V_p^p$) predictions, and combining them to handle the different cases for the final prediction ($S^p V^p$), ensures this bias is removable (Fig. 2).
- unspecifiable filter sensibility : the introduction of intermediate state variables ($S^o V^o$) to evaluate the model states, instead of direct evaluation using the data, allow for filter reactivity configuration without modifying filter internal temporal evolution. The "observed states" can have different dimensions than the filtered states, typically includes information about the observation estimate relevance (how based on actual sensor data or prior knowledge a cell estimate is) and affects the final filter state accordingly.

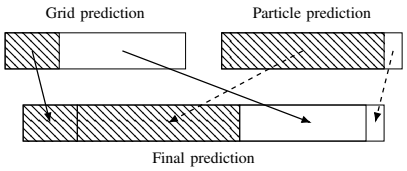

Fig. 1. Previous occupancy prediction pattern : each antecedent contribution being independently integrated, dynamic occupancy from particles is systematically reduced by free space prediction from the grid

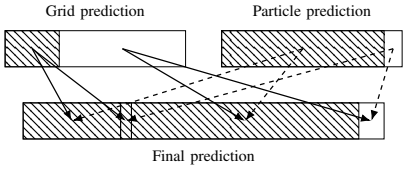

Fig. 2. New occupancy prediction pattern : grid and particle predictions are combined to preserve dynamic occupancy

B. Formalism

The objective of cell-based occupancy filters is to recursively estimate cell variables (SV), such as spatial occupancy, dynamics, semantic information, based on observations by various types of sensors.

Using the Bayesian programming framework [23], we define here a new model, incorporating hidden variables, and a new joint probability decomposition. Specific variables are defined for each model (for grid-based and particle-based predictions), the final states and velocities being the result of the fusion of these fused predictions and observed states.

C. Variable definition

C : Index of the cell.
C_g^{-1} : Index of the cell antecedent, used for the grid-based model.
C_p^{-1} : Index of the cell antecedent, used for the particle-based model.
S : State of the cell.
V : Velocity of the cell.
S_g^{-1} : Previous state of the cell antecedent, according to the grid-based model.
V_g^{-1} : Previous velocity of the cell antecedent, according to the grid-based model.
S_p^{-1} : Previous state of the cell antecedent, according to the particle-based model.
V_p^{-1} : Previous velocity of the cell antecedent, according to the particle-based model.
S_g^p : Predicted state of the cell, according to the grid-based model.
V_g^p : Predicted velocity of the cell, according to the grid-based model.
S_p^p : Predicted state of the cell, according to the particle-based model.
V_p^p : Predicted velocity of the cell, according to the particle-based model.
S^p : Predicted state of the cell.
V^p : Predicted velocity of the cell.
S^o : Observed state of the cell.
V^o : Observed velocity of the cell.
Z : Sensor measurement.

D. Joint distribution

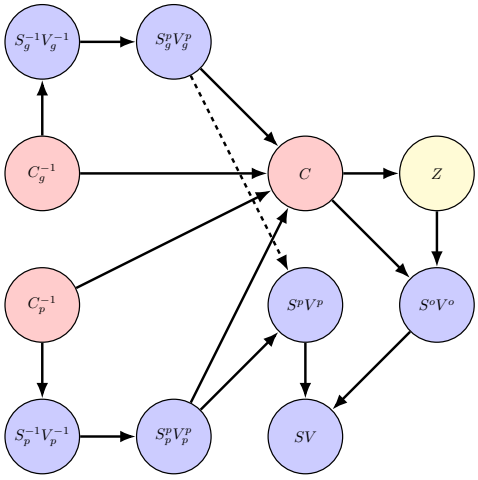

Fig. 3. Bayesian network representing the variable dependencies. The occupancy O can be inferred from the state S.

$$
\begin{aligned}
P(CC_g^{-1}C_p^{-1}SVS_g^{-1}V_g^{-1}S_p^{-1}V_p^{-1}S_g^pV_g^pS_p^pV_p^pS^pV^pS^oV^oZ) \;=\; & P(C_g^{-1}) \\
& P(S_g^{-1}V_g^{-1}|C_g^{-1}) \\
& P(S_g^pV_g^p|S_g^{-1}V_g^{-1}) \\
& P(C_p^{-1}) \\
& P(S_p^{-1}V_p^{-1}|C_p^{-1}) \\
& P(S_p^pV_p^p|S_p^{-1}V_p^{-1}) \\
& P(C|C_g^{-1}V_g^pC_p^{-1}V_p^p) \\
& P(S^pV^p|S_g^pV_g^pS_p^pV_p^p) \\
& P(Z) \\
& P(S^oV^o|ZC) \\
& P(SV|S^pV^pS^oV^o)
\end{aligned}
\tag{1}
$$

The variable dependencies are pictured as a Bayesian network on Fig. 3. Each expression can be interpreted as follows:

- $P(C_g^{-1})$ is the distribution over all possible antecedents of the cell in the grid-based model. It is chosen to be uniform, a cell is considered a priori reachable from all other cells with equal probability.
- $P(S_g^{-1}V_g^{-1}|C_g^{-1})$ is the conditional joint distribution over the state and velocity of the antecedents. This is a result of the previous iteration of the filter.
- $P(S_g^pV_g^p|S_g^{-1}V_g^{-1})$ is the prediction model, according to the grid-based model, based on grids. The states and velocities are jointly predicted, based on correlated dynamic models and state transition models.
- $P(C_p^{-1})$, $P(S_p^{-1}V_p^{-1}|C_p^{-1})$ and $P(S_p^pV_p^p|S_p^{-1}V_p^{-1})$ are the corresponding distributions for the particle-based model.
- $P(C|C_g^{-1}V_g^pC_p^{-1}V_p^p)$ is the antecedent selection model, defining if a cell C is reachable from the antecedent C_g^{-1} with the velocity V_g^p, and from the antecedent C_p^{-1} with the velocity V_p^p. Potential antecedents from grid-based and particle-based models are selected in pairs.
- $P(S^pV^p|S_g^pV_g^pS_p^pV_p^p)$ is the prediction fusion model. The final state-and-velocity prediction is based on both grid-based and particle-based model predictions, enabling the definition of dynamic-and-static model interactions.
- $P(Z)$ is the probability of given observation Z. In our system its actual value will not directly matter, as it will be the same factor for each potential S and V, Z being given.
- $P(S^oV^o|ZC)$ is the inverse sensor model distribution, recaping the current observations, and directly calculated and given by other modules.
- $P(SV|S^pV^pS^oV^o)$ is the final state-and-velocity generation distribution, according to the predicted and observed states, enabling prediction-and-observation interactions.

E. Problem expression

The aim of the Bayesian filtering process is to estimate the state and velocity with respect to the current observation for every cell: $P(SV|ZC)$. In the system, the filtering is applied on the hidden states, the state can then be deducted.

Let $\mathcal{O}$ be the set of all variables:

$$
\mathcal{O} = \{CC_g^{-1}C_p^{-1}SVS_g^{-1}V_g^{-1}S_p^{-1}V_p^{-1}S_g^pV_g^pS_p^pV_p^pS^pV^pS^oV^oZ\} \tag{2}
$$

By using a discretization of the antecedent cells and of the velocities, the filtering equation can be written:

$$
P(SV|ZC) = \frac{\sum\limits_{\mathcal{O}\setminus\{CSVZ\}} P(CC_g^{-1}C_p^{-1}SVS_g^{-1}V_g^{-1}S_p^{-1}V_p^{-1}S_g^pV_g^pS_p^pV_p^pS^pV^pS^oV^oZ)}{\sum\limits_{\mathcal{O}} P(CC_g^{-1}C_p^{-1}SVS_g^{-1}V_g^{-1}S_p^{-1}V_p^{-1}S_g^pV_g^pS_p^pV_p^pS^pV^pS^oV^oZ)} \tag{3}
$$

In the end, the distribution can be expressed as:

$$
\begin{aligned}
P(SV|ZC) \propto \sum_{\mathcal{O}\setminus\{CSVZ\}} \; & P(C_g^{-1}) \\
& P(S_g^{-1}V_g^{-1}|C_g^{-1}) \\
& P(S_g^pV_g^p|S_g^{-1}V_g^{-1}) \\
& P(C_p^{-1}) \\
& P(S_p^{-1}V_p^{-1}|C_p^{-1}) \\
& P(S_p^pV_p^p|S_p^{-1}V_p^{-1}) \\
& P(C|C_g^{-1}V_g^pC_p^{-1}V_p^p) \\
& P(S^pV^p|S_g^pV_g^pS_p^pV_p^p) \\
& P(Z) \\
& P(S^oV^o|ZC) \\
& P(SV|S^pV^pS^oV^o)
\end{aligned}
\tag{4}
$$

If wanted, a simple dynamic occupancy can be deducted:

$$
P(OV|ZC) = \sum_S P(O|S)P(SV|ZC) \tag{5}
$$

IV. Filter Resolution

The filtering process mainly consists of three steps : prediction, fusion-evaluation and particle resampling.

A. Model representations

The system is composed of two interconnected models, a grid-based and a particle-based one. At every iteration of the filter, both models are independently used for prediction, then fused and evaluated together, and finally segmented again. If the type of states taken into account can differ, some correspond to occupied cells (pedestrian, car, building, etc.) or empty cells (road, sidewalk, etc.), each can have specific motion and interaction models.

The grid-based model consists of a classic state grid, each cell encoding its state distribution. The elements estimated to be static by the filter are represented here.

The dynamic model consists of a set of moving particles, each having a real position and velocity, as well as a "weight" corresponding to the probability of the cell it would belong to to be occupied by a dynamic object with that state and dynamics. The elements estimated to be dynamic by the filter are represented here.

B. Predictions

Fig. 4 summarizes the prediction process. The two models are projected separately, the final prediction being the result of the fusion of those predictions. For each model, the prediction steps consist of :

- a dynamic projection : particles are projected according to their velocities and time since previous iteration, and Gaussian noises on position and speed, static grids remain unchanged.

- a state transition : to each model are associated state transition models. They mainly define the forgetting rate of previously estimated data over time, and facilitate certain transitions over others, to represent actual state transition or correction of state estimation errors.
- a reference frame displacement : as the system aims to be embedded on moving vehicles, the models have to be displaced according to the egomotion. The way the models are represented simplify this otherwise complex operation, simply consisting here of the transformation of the particle vectors (translation and rotation for positions, rotation for velocity) and interpolation of grids. The areas in the grid corresponding to newly discovered regions are set to prior states, taken out areas can be forgotten or used in a generated map.

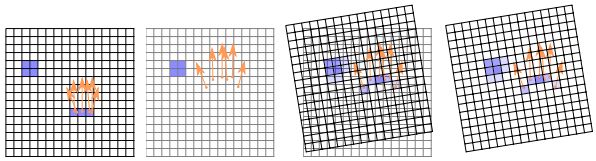

Fig. 4. Prediction step pattern. From left to right :
(i) Models at previous time-step. Both (grids and particles) are depicted in superposition.
(ii) Dynamic projection (for grids and particles, according to motion models) and state transitions (according to transition models).
(iii) Data expression in the new reference frame, by interpolation of grids and particle translation and rotation.
(iv) Predicted models at current time-step.

The potential values of those predictions are precomputed (and presampled), so that every $P(C_g^{-1})P(S_g^{-1}V_g^{-1}|C_g^{-1})P(S_g^pV_g^p|S_g^{-1}V_g^{-1})$ and $P(C_p^{-1})P(S_p^{-1}V_p^{-1}|C_p^{-1})P(S_p^pV_p^p|S_p^{-1}V_p^{-1})$ are calculated and associated to the relevant grid cells and particles.

C. Fusion of predictions and Evaluation

The updated distributions over states are evaluated in parallel in each cell of the current grid. All predicted particles are reorganized according to their position in the grid, so that only the relevant particles and cell data are sent for computation at the cell level (according to $P(C|C_g^{-1}V_g^pC_p^{-1}V_p^p)$).

1) Fusion of predictions: According to the fusion pattern defined by $P(S^pV^p|S_g^pV_g^pS_p^pV_p^p)$, each sample of static and dynamic prediction generate the final predictions. The fusion pattern can be precisely elaborated or learned, to insure dynamic occupancy conservation when displacing in the grid and that informative predictions are favored.

2) Evaluation: In a given cell C, those final predictions are confronted with the observation $P(S^oV^o|ZC)$. The observation model used is based on classic probabilistic sensor model [14], taking into account relevance of sensor data (sensor confidence, location of impacts, height of impacts, vertical distribution, etc.). The observation model can be defined for any kind of sensor, the distributions can be fused using probability fusion rules, allowing for heterogeneous data integration and generic treatment. The state definition used in this model can also be differ from the filtered one.

To generate the final state estimation, the generative distribution $P(SV|S^pV^pS^oV^o)$ is used, defining the likelihood of a final state and velocity according to the predicted and observed states. The chosen model parameters can be learned, and designed for fast appearance of objects in previously empty cells, while ensuring their persistence over time, and the neutral effect of uninformative observations on the final state estimations.

Duration of an update		Number of Particles		
		65536	131072	262144
N cells	$200 * 200$	$2.28ms$	$2.67ms$	$3.61ms$
	$400 * 400$	$2.57ms$	$2.60ms$	$3.50ms$
	$1000 * 1000$	$4.49ms$	$4.82ms$	$5.15ms$

Fig. 5. Duration of one update of the algorithm for different size of grid and number of particles, on a Nvidia Geforce GTX1080.

D. Particle resampling

Once the state distributions over cells are properly estimated, the remaining task is a 2-step particle resampling.

- particle reallocation : according to the dynamic probability of each cell in the grid, and other criteria (position, difference between predicted and observed states, velocity distribution, etc.), the global number of particles is reallocated to the relevant cells.
- particle generation : while evaluating state distributions, the weight of each existing particle is updated and normalized according to its relative contribution in the dynamic part. In each cell to which particles have been associated, particles are then drawn from the existing particle distribution and an initialization distribution, according to the proportion of "dynamic appearance" (gap between predicted and observed states and velocity distribution). Finally the cell dynamic coefficient is uniformly divided between the particles.

V. Results

A. GPU Implementation

A main advantage of the proposed approach is its high level of parallelism. Computations can be parallelized over each cell and particle. The algorithm has been implemented on Nvidia GPU architecture, using the CUDA API. Some implementation performances are indicated in Fig. 5.

B. Experimental platform

A Renault Zoe car (Fig. 6) has been equipped with 4 Ibeo Lux LIDARs, 3 on the front and one on the back. A Velodyne HDL64 LIDAR is placed on the top. Two IDS monocular RGB cameras are mounted on the front and back. Xsens GPS and IMU provide vehicle dynamic state. The computer is equipped with a dual-processor Intel Xeon and a GPU Nvidia Geforce Titan X.

A model of the experimental vehicle has been developped in simulation with the exact same sensor arrangement, and used in a dynamic Gazebo-built simulation environment.

Fig. 6. Experimental Platform: Renault Zoe car equipped with Velodyne HDL64, 4 Ibeo Lux LiDARs, cameras, Xsens GPS and IMU.

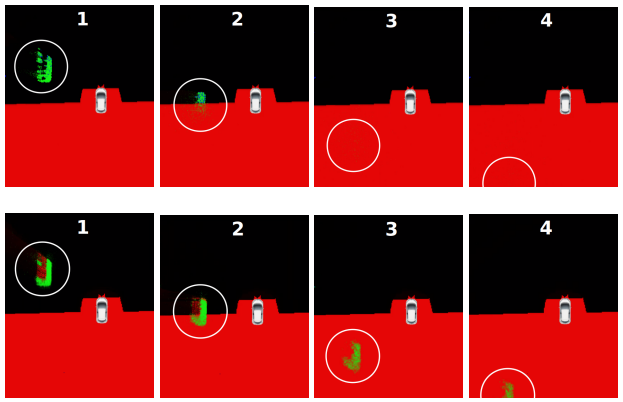

Fig. 7. Dynamic occupancy probability total mass of the particles before (top row) and after (bottom row) the introduction of the double prediction model. The image shows the ego-vehicle in white, the dynamic occupancy vehicle estimation in green, the free region in black and the unobserved region in red.

C. Double prediction model results

To highlight the advantages of the double prediction model, the simulated vehicle is placed in an open environment where a simulated car pass by the ego-vehicle to the left, in the opposite direction. The car sensors are obstructed to only observe in front of the vehicle.

CMCDOT[22] is used for comparison with a single prediction model. As the purpose of this example is not to evaluate the particles dynamic model accuracy, but the conservation of the particle occupancy probability mass in time, the particle filter motion model is parameterized with a low variance, which translates in a particle quasi-rectilinear movement prediction, allowing easy observation of the conservation of the total probability mass in the new model.

The results can be observed in Fig. 7. When the dynamic particles enter the unobserved region, the previous CMCDOT prediction model resulted in a fast loss in dynamic occupancy probability mass, rapidely disappearing from the grid. With new double prediction model, the particle mass is properly preserved, until reaching the border of the grid.

A more detailed analysis is shown in Fig. 8. Here the total dynamic probability mass has been accumulated over the whole grid for both the single prediction (blue dashed) and the double prediction (red solid) versions, highlighting the important difference of behavior. The vertical lines separate the different phases of the test, corresponding to the ones in Fig. 7.

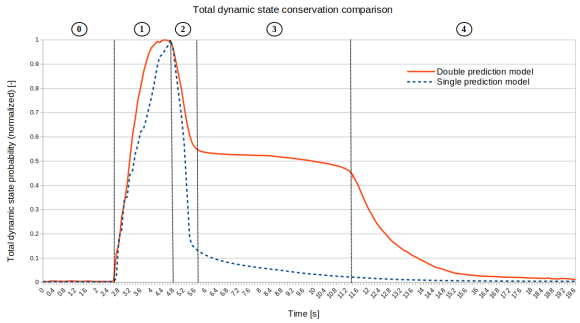

Fig. 8. Comparison of the total dynamic state probability mass over the grid for the single and double prediction model versions.

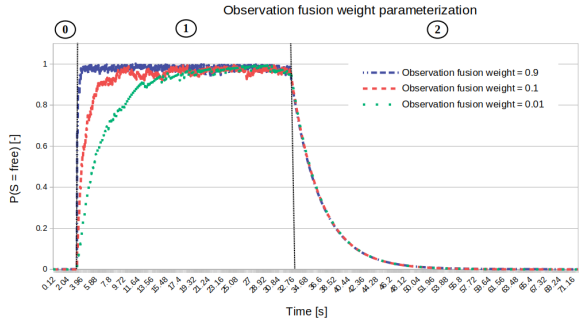

Fig. 9. Comparison of the free-state estimation dynamics on a cell of the state grid for different parameters for the free-state observation fusion weight.

D. Decoupled observation model results

To show the advantages of the introduced observation state and fusion model, the transition of a cell state to a free state (observation) and from it (dissipation) have been plotted in Fig. 9. The observation model transitions from unobserved (0) to free (1) and finally back to unobserved (2). The new degrees of freedom added with the decoupled observation model now allow to parameterize the filter response to inputs while maintaining internal dynamics.

E. Experimental results

The perception framework has been tested on the real experimental platform described before. Fig. 10 displays a filtered grid, in association with the corresponding camera image, with annotations for a car, a motorbike and a van.

Fig. 10. Camera, and filtered state grid showing static occupancy (blue), dynamic occupancy (green) empty space (black) and undefined areas (red).

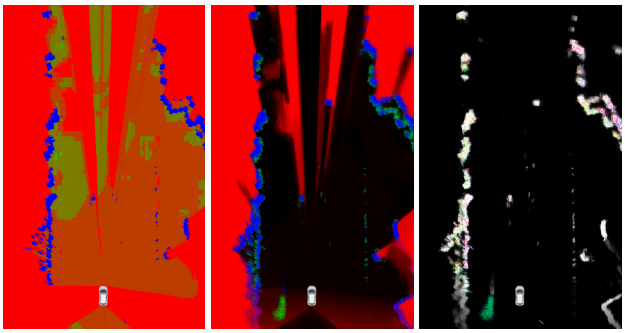

Fig. 11. From left to right, examples of
- Instantaneous occupancy grid, showing occupancy (blue), empty space (green) and undefined areas (red)
- Filtered state grid, showing static occupancy (blue), dynamic occupancy (green) empty space (black) and undefined areas (red)
- Velocity grid, the average estimated velocity of each occupied cell, white for static and using a color wheel for the different directions.

Fig. 11 illustrates the ability of the method to propagate dynamic state over previously-seen-as-free cells when no data is observed in real-case scenarios. A car is passing on the left of the experimentation vehicle, a blind spots of the sensors, thus disappears in the instantaneous occupancy grid. The dynamic cells propagate according to their estimated velocity, the car is still visible in the filtered occupancy grid and in the estimated velocity grid without dilution.

VI. Conclusion

In this paper, we present a new formalization for dynamic occupancy grid tracking, introducing the interaction of a static grid-based and a dynamic particle-based model for prediction, and splitting observation and prediction variables. The introduction of the double prediction model solves important issues, such as the unwanted fast dilution of dynamic occupancy, while the addition of observation states leads to a parameterized reactivity of the filter, a previously mishandled issue which was only hardly addressed by tuning of internal transition models, leading to important side effects. The specific improvements are illustrated, and the overall performances of the system have indeed greatly improved, in theory and in practice. But to make those improvements quantitatively explicit, our and other dynamic occupancy grid filtering methods still lack clear and shared evaluation criteria, metrics and reference database. Hard to define and agree on, such definitions would greatly simplify method comparison and parameter learning.

Acknowledgment

This work has been developed within the scope of an "Institut de Recherche Technologique NanoElec" project, founded by the French program "Investissement d'Avenirs" ANR-10-AIRT-05.

References

[1] A. Petrovskaya, M. Perrollaz, L. Oliveira, L. Spinello, R. Triebel, A. Makris, J.-D. Yoder, U. Nunes, C. Laugier, and P. Bessière, "Awareness of road scene participants for autonomous driving," in *Handbook of Intelligent Vehicles*, A. Eskandarian, Ed. Springer, 2012, pp. 1383–1432.

[2] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Multi-target tracking using joint probabilistic data association," in *Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on*, vol. 19. IEEE, 1980, pp. 807–812.

[3] Z. Khan, T. Balch, and F. Dellaert, "An mcmc-based particle filter for tracking multiple interacting targets," in *Computer Vision-ECCV 2004*. Springer, 2004, pp. 279–290.

[4] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[5] H. Moravec, "Sensor fusion in certainty grids for mobile robots," *AI magazine*, vol. 9, no. 2, p. 61, 1988.

[6] S. O'Callaghan and F. Ramos, "Gaussian process occupancy maps," *I. J. Robotic Res.*, vol. 31, pp. 42–62, 01 2012.

[7] S. T. O'Callaghan and F. T. Ramos, *Gaussian Process Occupancy Maps for Dynamic Environments*. Springer International Publishing, 2016, pp. 791–805.

[8] R. Senanayake and F. Ramos, "Bayesian hilbert maps for continuous occupancy mapping in dynamic environments," *Conference on Robot Learning*, vol. 78, p. 458–471, 2017.

[9] R. S. V. Guizilini and F. Ramos, "Dynamic hilbert maps: Real-time occupancy predictions in changing environments," *Proceedings - IEEE International Conference on Robotics and Automation*, p. 4091–4097, May 2019.

[10] M. B. S. Hoermann and K. Dietmayer, "Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling," *Proceedings - IEEE International Conference on Robotics and Automation*, p. 2056–2063, 2017.

[11] C. G. M. Schreiber, V. Belagiannis and K. Dietmayer, "Motion estimation in occupancy grid maps in stationary settings using recurrent neural networks," *Proceedings - IEEE International Conference on Robotics and Automation*, p. 8587–8593, 2020.

[12] R. C. T. Roddick, "Predicting semantic map representations from images using pyramid occupancy networks," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11 135–11 144, 2020.

[13] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessière, "Bayesian occupancy filtering for multitarget tracking: an automotive application," *The International Journal of Robotics Research*, vol. 25, no. 1, pp. 19–30, 2006.

[14] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. The MIT Press, 2005.

[15] T. Gindele, S. Brechtel, J. Schroder, and R. Dillmann, "Bayesian occupancy grid filter for dynamic environments using prior map knowledge," in *Intelligent Vehicles Symposium, 2009 IEEE*. IEEE, 2009, pp. 669–676.

[16] S. Brechtel, T. Gindele, and R. Dillmann, "Recursive importance sampling for efficient grid-based occupancy filtering in dynamic environments," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 3932–3938.

[17] R. Danescu, F. Oniga, and S. Nedevschi, "Modeling and tracking the driving environment with a particle-based occupancy grid," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 12, no. 4, pp. 1331–1342, 2011.

[18] A. Nègre, L. Rummelhard, and C. Laugier, "Hybrid Sampling Bayesian Occupancy Filter," in *IEEE Intelligent Vehicles Symposium (IV)*, Dearborn, United States, June 2014. [Online]. Available: https://hal.inria.fr/hal-01011703

[19] M. Saval-Calvo, L. Valdés, J. Castillo-Secilla, S. Cuenca-Asensi, A. Martínez-Álvarez, and J. Villagra, "A review of the bayesian occupancy filter," *Sensors*, vol. 17, 03 2017.

[20] D. Nuss, S. Reuter, M. Thom, T. Yuan, G. Krehl, M. Maile, A. Gern, and K. Dietmayer, "A random finite set approach for dynamic occupancy grid maps with real-time application," *The International Journal of Robotics Research*, vol. 37, no. 8, pp. 841–866, 2018. [Online]. Available: https://doi.org/10.1177/0278364918775523

[21] S. Steyer, G. Tanzmeister, and D. Wollherr, "Grid-based environment estimation using evidential mapping and particle tracking," *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 3, pp. 384–396, 2018.

[22] L. Rummelhard, A. Negre, and C. Laugier, "Conditional Monte Carlo Dense Occupancy Tracker," in *18th IEEE International Conference on Intelligent Transportation Systems*, Las Palmas, Spain, Sept. 2015. [Online]. Available: https://hal.inria.fr/hal-01205298

[23] P. Bessière, E. Mazer, J. Ahuactzin-Larios, and K. Mekhnacha, *Bayesian Programming*. CRC Press, Dec. 2013.