Bayesian Occupancy Filtering for Multi-Target Tracking: an Automotive Application

Christophe Coué, Cédric Pradalier, Christian Laugier, Thierry Fraichard and Pierre Bessière
Inria Rhône-Alpes & Gravir -CNRS
http://www.inrialpes.fr/sharp
Contact: firstname.lastname@inrialpes.fr

***Abstract*— Reliable and efficient perception and reasoning in dynamic and densely cluttered environments is still a major challenge for driver assistance systems. Most of today's systems use target tracking algorithms based on object models. They work quite well in simple environments such as freeways, where few potential obstacles have to be considered. However, these approaches usually fail in more complex environments featuring a large variety of potential obstacles, as it is usually the case in urban driving situations. In this paper, we propose a new approach for robust perception and risk assessment in highly dynamic environments. This approach is called *Bayesian Occupancy Filtering*, it basically combines a 4-dimensional occupancy grid representation of the obstacle state-space with Bayesian filtering techniques.**

***Index Terms*— Multi-target tracking, bayesian state estimation, occupancy grid**

I. Introduction

A. The ADAS Context

Unlike regular cruise control systems, Adaptive Cruise Control (ACC) systems use a range sensor to regulate the speed of the car while ensuring collision avoidance with the vehicle in front. ACC systems were introduced on the automotive market in 1999. Since then, surveys and experimental assessments have demonstrated the interest for this kind of systems. They are the first step towards the design of future Advanced Driver Assistance Systems (ADAS) that should help the driver in increasingly complex driving tasks. The use of today's commercially available ACC systems is pretty much limited to motorways or urban expressways without crossings. The traffic situations encountered are rather simple and attention can be focused on a few, well defined detected objects (cars and trucks). Nonetheless, even in these relatively simple situations, these systems display a number of limitations: they are not very good at handling fixed obstacles and because of that may generate false alarms. Moreover, in 'cut-in' situations, *i.e.* when the intrusion of an other vehicle or a pedestrian in the detection beam is too close to the vehicle, they may be unable to react appropriately.

A wider use of such systems requires to extend their range of operation to more complex situations in dense traffic environments, around or inside urban areas. In such areas, traffic is characterised by lower speeds, tight curves, traffic signs, crossings and "fragile" traffic participants such as motorbikes, bicycles or pedestrians.

B. Problem

Prerequisites to a reliable ADAS in such complex traffic situations are:

- *Robust and accurate sensing* of the environment. In particular, dynamic characteristics of the traffic participants, such as position and velocity, have to be correctly estimated;
- *Appropriate sensing representation* that allows both to have a good understanding of the traffic situation, and to select the most appropriate driving decisions.

C. Related Work

1) Multi-Target Tracking: The estimation of the dynamic characteristics of the traffic participants is basically a *multi-target tracking* problem. The objective is to collect *observations*, *i.e.* sensor data, on one or more *potential obstacles* in the environment of the vehicle, and then to estimate at each time step (and as robustly as possible) the obstacles positions and velocities. Classical approach is to track the different objects independently, by maintaining a list of *tracks*, *i.e.* a list of currently known objects. The main difficulty of multi-target tracking is known as the *Data Association* problem. It includes observation-to-track association and track management problems. The goal of observation-to-track association is to decide whether a new sensor observation corresponds to an existing track or not. Then track management includes deciding whether existing tracks should be maintained or deleted, and whether new tracks should be created. Numerous methods exist to perform data association [1], [2], [3]. The reader is referred to [4] for a complete review of the existing tracking methods with one or more sensors.

Urban traffic scenarios are still a challenge in multi-target tracking area: the traditional data association problem is intractable in situations involving numerous appear-

ances, disappearances and occlusions of a large number of rapidly manoeuvring targets.

In [5], a classical Multiple Hypothesis Tracking technique is used to track moving objects while stationary objects are used for SLAM. Unfortunately, the authors did not explicitly address the problem of the interaction between tracked and stationary objects, *e.g.* when a pedestrian is temporary hidden by a parked car. It is one of the purpose of our approach to solve this problem.

2) Grid Representation of the Environment: The *occupancy grids* framework [6], [7] is a classical way to describe the environment of a mobile robot. It has been extensively used for static indoor mapping using a 2-dimensional grid [8]. The goal is to compute from the sensor observations the probability that each cell is full or empty. To avoid a combinatorial explosion of grid configuration, the cell states are estimated as *independent* random variables.

More recently, occupancy grids have been adapted to track multiple moving objects [9]. In this approach, spatio-temporal clustering applied to *temporal maps* is used to perform motion detection and tracking. A major drawback of this work, relatively to the ADAS context, is that a moving object may be lost due to occlusion effects.

D. Contribution

The objective of this paper is to propose a new approach for *a robust perception, representation and analysis of highly dynamic environments*. Four main motivations were taken into account in the design of this approach:

- *Taking explicitly into account the uncertainty* (which is inherently present in any model of a real phenomenon) when estimating the state of the environment;
- *Avoiding the "data association problem"*, which usually fail to solve the complex scenarios we would like to address, *i.e.* scenarios involving numerous appearances, disappearances and occlusions of several rapidly manoeuvring targets;
- *Increasing the robustness of the system relatively to object occlusions, appearances and disappearances*, by exploiting at any time all the relevant information on the vehicle environment; this information includes the description of the occupied areas, of the unoccupied areas, and of the hidden areas (*i.e.* areas of the environment that are temporary hidden to sensors by an obstacle);
- *Designing a method that could be implemented later on a dedicated hardware*, in order to both reach high performances and decrease the costs of the final system.

Our approach is based on a *probabilistic grid representation of the obstacles state space*. This approach allows us to meet the four previous objectives:

- *Uncertainty* is explicitly taken into account thanks to the *probabilistic reasoning paradigm*, which is becoming a key paradigm in robotics: various approaches based on this paradigm have already been successfully used to address several robotic problems, such as CAD modelling [10] or map building and localisation (SLAM) [11], [12], [13].
- *The data association problem* is avoided by reasoning on a probabilistic grid representation of the dynamic environment. In such a model, concepts such as *objects* or *tracks* do not explicitly exist; they are replaced by more useful properties such as occupancy or risk, which are directly estimated for each cell of the grid using both sensor observations and some prior knowledge. Furthermore, when estimating occupancy probability thanks to an adequate sensor model, the hidden parts of the environments can also be explicitly characterised. Since we consider both the positions and the velocities of the potential obstacles with respect to our vehicle, this grid is 4-dimensional and is called the *Obstacle State Space (OSS)* grid .
- *The dynamic nature of the environment and robustness to object occlusion* is addressed using a novel two-steps mechanism allowing to take into account the *sensor observations history and the temporal consistency of the scene*. This mechanism estimates, at each time step, the state of the occupancy grid by combining a prediction step (history) and an estimation step (new measurements). This approach is derived from the *Bayes filters* approach [14]; our filter is called the *Bayesian Occupancy Filter (BOF)*.
- *The Bayesian Occupancy Filter has been designed in order to be highly parallelisable*. So a hardware implementation on a dedicated chip is possible, which will lead to a really efficient way to represent the environment of an automotive vehicle. This problem (*i.e.* SoC for the *BOF*) is currently addressed within the scope of the Safemove France-Korea project.

Fig. 1. The *CyCab* experimental vehicle. It is equipped with a Sick laser range finder

E. Outline of the Paper

The next section presents the estimation of the occupancy grid in a static case, *i.e.* taking into account only the last sensor observation. Section III describes the Bayesian Occupancy Filter itself, *i.e.* how we take into account the sensor observation history. Experimental results are shown in sections II and III. Section IV describes a collision avoidance application based on the *BOF*. This approach has been implemented on the *CyCab*, an electric vehicle[1] equipped with a Sick laser range finder, allowing the system to estimate targets position and velocity (Fig 1).

II. Static estimation of the occupancy probability

In this section, we introduce first the bayesian formalism (called *Bayesian Programming*) which has been used for developing our approach. Then, we present the bayesian program which has been developed for estimating the state of the occupancy grid of the vehicle environment, by using only the latest sensor observations (i.e. static estimation). In a third part, experimental results are presented and discussed.

A. Bayesian formalism and related computational tools

The *Bayesian Programming* framework has initially been developed by our research team, for designing robust robot control programs [15], [16], [17]. Today, it is used for addressing various problems involving uncertain or incomplete knowledge, e.g. in [10], [18]. This framework is based on a well-defined mathematical theory, and it provides both formal and computational tools for designing applications in a systematic way.

From the formal point of view, a *Bayesian Program* is made up of two main parts: a *description* and a *question*.

- *The description.* This part of the bayesian program can be viewed as a knowledge base containing the a priori information available on the problem at hand; it mainly represents a joint probability distribution. This description is made up of three components:
  1) A set of *relevant variables* on which the joint distribution is defined, e.g. motor, sensory, or internal state variables.
  2) A *decomposition* of the joint distribution as a product of simpler terms. It is obtained by applying Bayesian rules and by taking advantages of the conditional independencies that may exists between the variables.
  3) The *parametric forms* assigned to each of the terms appearing in the decomposition (they are required to compute the joint distribution).

[1] `http://www.robosoft.fr`

- *The probabilistic questions.* Given a distribution, it is possible to ask *probabilistic questions*. Basically, a question can be expressed by first partitioning the set of variables into three sub-sets *(S,K,F)* (representing respectively the "searched variables", the "known variables", and the "free variables"), and by writing a probabilistic expression of the type :

$$P(S \mid K)? \tag{1}$$

Given the previous description, it is always possible to answer such a question, *i.e.* to compute the probability distribution $P(S \mid K)$. This can be done using the following general inference :

$$\begin{aligned} P(S \mid K) &= \frac{\sum_F P(S\ F\ K)}{P(K)} \\ &= \frac{1}{\alpha} \times \sum\nolimits_F P(S\ F\ K), \end{aligned} \tag{2}$$

where α is a normalization term.

From the computational point of view, it is well known that a brute force approach cannot be applied in practice for solving the previous inference problem (Bayesian inference has been shown to be NP-Hard [19]), and that a previous symbolic simplification phase can drastically reduce the number of sums necessary to compute a given distribution. This is why we have developed an API[2], called ProBT®, which allows to easily express Bayesian Programs and to perform in an efficient way the related symbolic and numerical operations. This engine operates in two complementary stages:

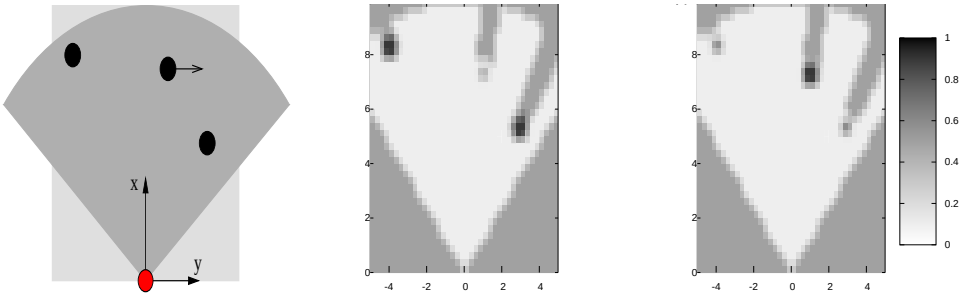
- A symbolic simplification stage which reduces the complexity of the probability distribution to be computed.
- A numerical stage that actually computes the related distributions.

This engine is now commercialized by the Probayes[3] company. The approach described in this paper has been implemented using this framework.

B. Bayesian Program for static estimation

Our goal is to estimate the occupancy probability of each cell of the grid, using the last set of *sensor observations*. These observations represent a pre-processed information given by a sensor. At each time step, the sensor is able to return a list of detected objects, along with their associated positions and velocities in the sensor reference frame. In practice, this set of observations could also contain two types of false measurements : the *false alarms* (i.e. when the sensor detects a non existing object) and the *missed detection* (i.e. when the sensor does not detect an existing object).

[2] Application Programming Interface
[3] `http://www.probayes.com`



a) $z_1 = (8.3, -4, 0, 0)$, $z_2 = (5, 3, 0, 0)$, $z_3 = (7.3, 1.9, 0, 0.8)$

b) $P([E_C = 1] \,|\, \mathcal{Z}\, c)$ $[c = (x, y, 0, 0)]$

c) $P([E_C = 1] \,|\, \mathcal{Z}\, c)$ $[c = (x, y, 0, 0.8)]$

Fig. 2. Example of static grid estimation. a) the situation and the sensor observations, given in $(x, y, \dot{x}, \dot{y})$; b) and c) values of occupancy probability for two "slices" of the 4-dimensional grid, *i.e.* for all possible positions at a given speed.

Then, solving the previous static estimation problem can be done by building the following Bayesian Program :

(1) Chosing the relevant variables and decomposition.

- C : The cell itself; this variable is 4-dimensional and represents a position and a speed relative to the vehicle.
- E_C : The state of the cell C, occupied or not.
- $\mathcal{Z}$: The sensor observation set; one observation is denoted Z_s; the number of observation is denoted S; each variable Z_s is 4-dimensional.
- M : The "matching" variable; its goal is to specify which observation of the sensor is currently used to estimate the state of a cell.

Then, the following decomposition of the joint distribution determined by these variables can be chosen:

$$P(C\, E_C\, \mathcal{Z}\, M) = \left(\begin{array}{c} P(C) P(E_C \,|\, C) P(M) \\ \times \prod_{s=1}^{S} P(Z_s \,|\, C\, E_C\, M) \end{array} \right). \quad (3)$$

(2) Assigning the parametric forms. According to our knowledge of the problem to be solved, we can assign the following parametric forms to each of the terms of the previous decomposition:

- $P(C)$ represents the information on the cell itself. As we always know the cell for which we are currently estimating the state, this distribution does not need to be specified.
- $P(E_C \mid C)$ represents the *a priori* information on the occupancy of the cell. If available, a *prior* distribution could be used to specify it. Otherwise, a uniform distribution has to be selected. Next section will explain how the prior distribution may be obtained from passed estimation.
- $P(M)$ is chosen uniform.
- The shape of $P(Z_s \,|\, C\, E_C\, M)$ depends on the value the matching variable:
  - if $M \neq s$, the observation is not due to the cell C. Consequently, we cannot say anything on this observation. $P(Z_s \,|\, C\, E_C\, M)$ is defined by a uniform distribution.
  - if $M = s$, the form of $P(Z_s \,|\, C\, E_C\, M)$ is given by the *sensor model*. Its goal is to model the sensor response knowing the cell state. Details on this model can be found in [7].

(3) Solution of the problem. It is now possible to ask the *bayesian question* corresponding to the searched solution[4]. Since the problem to solve consists in finding a good estimate of the cell occupancy, the question can be stated as follows :

$$P(E_C \mid \mathcal{Z}\; C)? \quad (4)$$

Following the general inference mechanism given by (2), the result of the inference can be written as follows:

$$P(E_C \mid \mathcal{Z}\; C) \propto \sum_{M=1}^{S} \left(\prod_{s=1}^{S} P(Z_s \mid E_C\; C\; M) \right). \quad (5)$$

The result of this inference is computed by our inference engine. During this inference, the sum on the previous variables allows to *consider every sensor observation* when updating the state of one cell. It should be noticed that the estimation step is performed without any explicit association between cells and observations; this problematic operation is replaced by the integration of all the possible values of M.

C. Experimental result

We have tested the previous approach for the static estimation of an occupancy grid, using both a simulator and a real vehicle equiped with a laser sensor (the Cycab).

[4]The answer to this question is given by our inference engine. It represents the searched probability distribution

Fig 2 shows some resulting grid estimations. The left picture depicts the situation: the sensor mounted on the CyCab is located at $(x = 0, y = 0)$; the part of the environment covered by the grid is represented by the light gray rectangle; the sensor field of view is modeled by the dark gray area. In this situation, three obstacles (black disks) are present in front of the Cycab; two of them are stationary objects, the third one is moving from the left to the righ, at a speed of $0.8\ m/s$ represented by a black arrow. In this first experiment, the CyCab is not moving.

As mentioned earlier, we use a 4-dimensional grid. Thus only 2-dimensional "slices" of the grid are depicted by Fig. 2b and Fig. 2c. On both figures, the occupancy probability value is given by the gray level of the cell. We could see the correspondence between probability value and gray level on the right side of the figure.

Fig. 2b depicts the occupancy probability of each cell corresponding to a null relative velocity (*i.e.* $c = [x, y, 0, 0]$), which is the speed associated to the two previous sensor observations. As expected, two areas with high occupancy probabilities are visible. These probability values depends on several factors attached to the sensor model: the probability of true detection, the probability of false alarm, and the sensor accurancy. Since the measured speed for the third obstacle is not null, any area of high occupancy probability corresponding to this observation is only represented in the related slices of the grid (i.e. the slice corresponding to $c = [x, y, 0, 0.8]$ in this case, see Fig. 2c). It should be noticed that the cells located outside of the sensor field of view, or the cells hidden by one of the three sensor observations (i.e. the cells located behind the three detected obstacles) cannot be observed; consequently, nothing really consistent can be said about these cells, and the system has given an occupancy probability value of 0.5 for these cells (red areas). Finally, all the cells located in the observable area and not related to any sensor obsservation, are associated to a low occupancy probability value (purple areas).

III. The Bayesian Occupancy Filter

A. Problem addressed and approach

We are now interested in taking into account the sensor observation history, in order to be able to make more robust estimations in changing environments (i.e. in order to be able to process temporary objects occlusions and detection problems). Our approach for solving this problem is to make use of an appropriate Bayesian filtering technique called the *Bayesian Occupancy Filter (BOF)*.

Bayes filters [14] address the general problem of estimating the state sequence x^k, $k \in \mathbb{N}$ of a system given by:

$$x^k = f^k(x^{k-1}, u^{k-1}, w^k), \tag{6}$$

where f^k is a possibly nonlinear transition function, u^{k-1} is a "control" variable (*e.g.* speed or acceleration) for the sensor which allows to estimate its ego-movement between time $k-1$ and time k, and w^k is the process noise. This equation describes a Markov process of order one.

Let z^k be the sensor observation of the system at time k. The objective of the filtering is to recursively estimate x^k from the sensor measurements:

$$z^k = h^k(x^k, v^k). \tag{7}$$

where h^k is a possibly nonlinear function and v^k is the measurement noise. This function models the uncertainty of the measurement z^k of the system's state x^k.

In other words, the goal of the filtering is to recursively estimate the probability distribution $P(X^k \mid Z^k)$, known as the *posterior* distribution. In general, this estimation is done in two stages: *prediction* and *estimation*. The goal of the prediction stage is to compute an *a priori* estimate of the target's state known as the *prior* distribution. The goal of the estimation stage is to compute the *posterior* distribution, using this *a priori* estimate and the current measurement of the sensor.

Exact solutions to this recursive propagation of the posterior density do exist in a restrictive set of cases. In particular, the Kalman filter [20][21] is an optimal solution when the functions f^k and h^k are linear and the noises w^k and v^k are Gaussian. But in general, solutions cannot be determined analytically, and an approximate solution has to be computed.

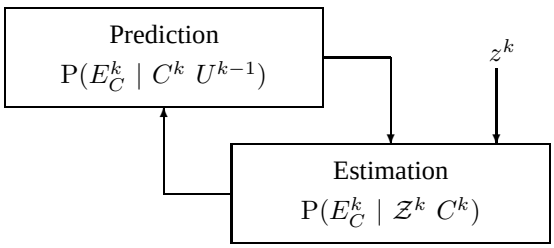


Fig. 3. Bayesian Occupancy Filter as a recursive loop.

In our case, the state of the system is given by the occupancy state of each cell of the grid, and the required conditions for being able to apply an exact solution such as the Kalman filter are not alway verified. Moreover, the particular structure of the model (occupancy grid) and the real-time constraint coming from the ADAS application, has led us to develop the new concept of *Bayesian Occupancy Filter*. This filter consists in estimating the occupancy state in a two-steps, as depicted in fig 3.

B. The BOF estimation step

In this loop, the estimation step is similar to the static estimation of the grid depicted in the previous section, except that the *a priori* on the cell occupancy $P(E_C^k \mid C^k)$ is not given by an uniform distribution, but by the result of the previous prediction step.

C. The BOF prediction step

The goal of this processing step is to estimate an *a priori* model of the occupancy probability at time k of a cell using the latest estimation of the occupancy grid, *i.e.* the estimation at time $k-1$. The variables that are relevant here are the followings :

- C^k : Cell C considered at time k.
- E_C^k : State of cell C at time k.
- C^{k-1} : Cell C at time $k-1$.
- E_C^{k-1} : State of cell C at time $k-1$.
- U^{k-1} : "control" input of the *CyCab* at time $k-1$. For example, it could be a measurement of its instantaneous velocity at time $k-1$.

Using these variables, we can define the following decomposition of the joint distribution :

$$P(C_k\, E_C^k\, C^{k-1}\, E_C^{k-1}\, U^{k-1}) = \begin{pmatrix} P(U^{k-1})P(C^{k-1}) \\ \times P(E_C^{k-1} \mid C^{k-1}) \\ \times P(C^k \mid C^{k-1}\, U^{k-1}) \\ \times P(E_C^k \mid E_C^{k-1}\, C^{k-1}\, C^k) \end{pmatrix} \quad (8)$$

Then, we can assign the following parametric to each of the previous decomposition terms :

- $P(C^{k-1})$ and $P(U^{k-1})$ are chosen as uniform distributions.
- $P(E_C^{k-1} \mid C^{k-1})$ is given by the result of the estimation step at time $k-1$.
- $P(C^k \mid C^{k-1}\ U^{k-1})$ is given by the dynamic model. It represents the probability that an object has moved from the cell C^{k-1} to the cell C^k. This movement is due to the object himself and to the *CyCab* movement between times $k-1$ and k. To define this model, we suppose a constant velocity model subject to zero mean Gaussian errors for the moving objects.
- $P(E_C^k \mid E_C^{k-1}\, C^{k-1}\, C^k)$ represents the probability that an existing object at time $k-1$ (*i.e.* $[E_C^{k-1} = 1]$ still exists at time k (*i.e.* $[E_C^k = 1]$). As we consider that objects can not disappear, Dirac are chosen for these distributions.

The problem to be solved is to find an estimate of the occupancy probability for each cell of the grid. Solving this problem can be done by asking the following question to our inference engine :

$$P(E_C^k \mid C^k\ U^{k-1})? \quad (9)$$

The result of the inference can be expressed as follows :

$$P(E_C^k \mid C^k\ U^{k-1}) \propto \sum_{\substack{C^{k-1} \\ E_C^{k-1}}} \begin{pmatrix} P(C^k \mid C^{k-1}\ U^{k-1}) \\ \times P(E_C^{k-1} \mid C^{k-1}) \end{pmatrix}. \quad (10)$$

Unfortunately, most of the time this expression cannot be expressed analytically, and it cannot be computed in real time. This is why an approximate solution of the integral term has to be computed. Our approach for making this computation is based on the fact that only few points are needed to approximate the integral. Thus, for each cell of the grid at time $k-1$, we can compute the probability distribution $P(C^k \mid C^{k-1})$; then, A cell c^k is drawn according to this probability distribution; finally, the cell C^{k-1} is used to update the predicted state of the cell c^k. It should be noticed that the complexity of the previous algorithm increases linearly with the number of cells in our grid, and ensures that the most informative points are used to compute the sum appearing in (10).

Using the *BOF* approach, the estimation of the occupancy grid at time k is done in two steps : (1) the prediction step makes use of both the result of the estimation step at time $k-1$ and a dynamical model, for computing an *a priori* estimate of the grid; (2) then, the estimation step makes use of both this prediction result and the sensor observations at time k, to compute the grid.

D. Experimental results

Fig 4 shows a short sequence of successive prediction and estimation results, for a dynamic scene involving two moving obstacles. The objective of this example, is to experimentally demonstrate the robustness of our approach to objects occlusions. The first row describes the experimental conditions : the *CyCab* is immobile, and its sensors can observe two moving objects $O1$ and $O2$ ($O1$ is moving from left to right, and $O2$ is moving from right to left). In the situation depicted by the fig 4-c1, $O1$ is temporary hidden by $O2$ (and thus $O1$ is not detected by the Sick laser range finder).

The second and the third row respectively show the results of the prediction step and of the estimation step. We have chosen to only represent the cells of the grid corresponding to a relative speed equal to ($\dot{x} = 0.0m/s,\ \dot{y} = 1.0m/s$), which is close to the speed of $O1$. The occupancy probability of the related cells are represented by several colors.

In this example, an area of "high occupancy probability", which corresponds to the moving objects, is well characterized in fig 4-a2 and in fig 4-a3. One can also

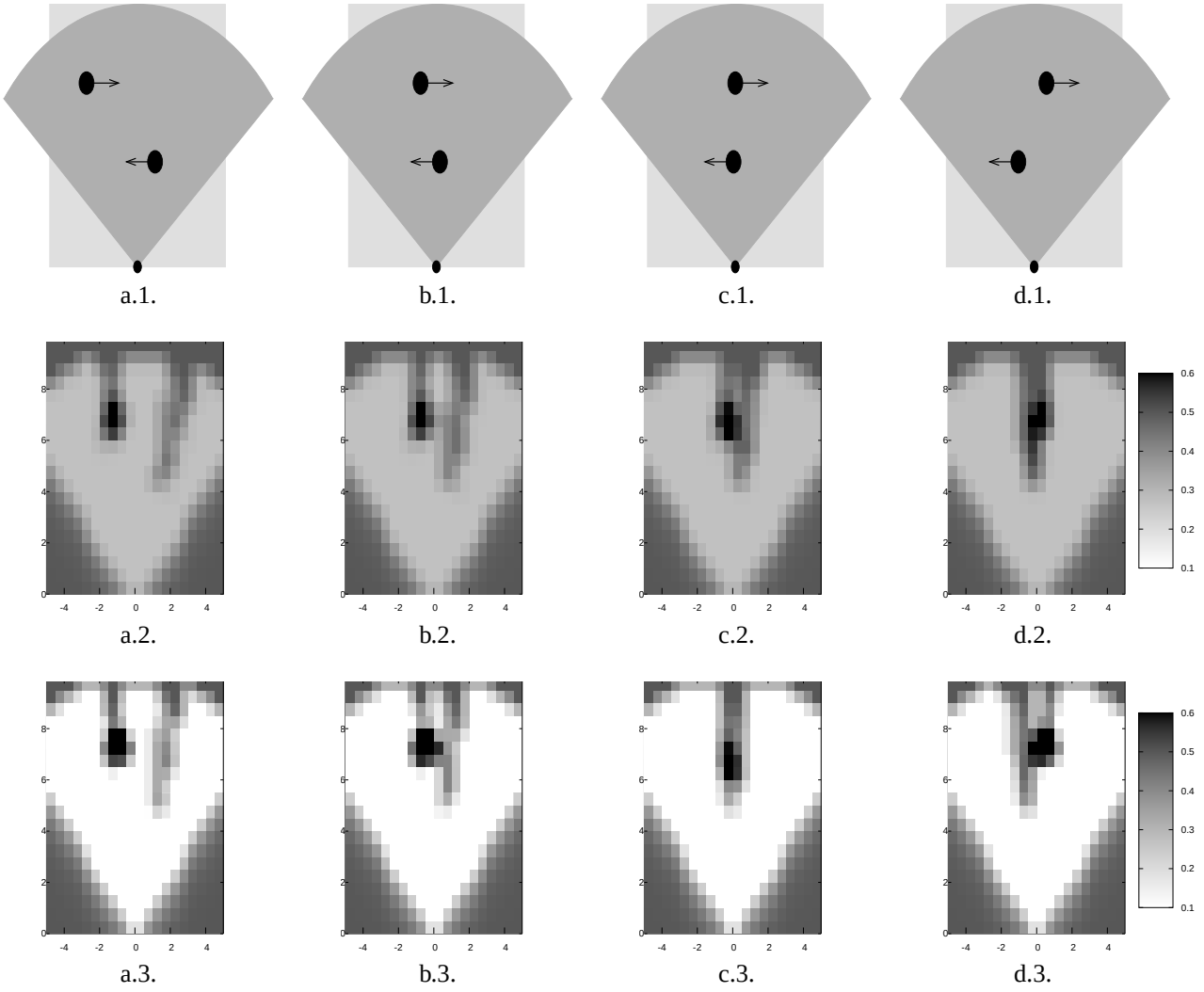


Fig. 4. A short sequence of a dynamic scene. The first row describes the situation: a moving object is temporary hidden by a second object. The second row shows the predicted occupancy grids, and the third row the result of the estimation step. The grids show $P([E_C^k = 1] \mid x\ y\ [\dot{x} = 0.0]\ [\dot{y} = 1.0])$

noticed that the areas hidden by the moving objects have an occupancy probability values equals to 0.5. Similar results can be seen on fig 4-b2 and fig 4-b3. The fig 4-c2 shows the result of the prediction step, based on the grid of fig 4-b3 and on the used dynamic model; this prediction shows that an object is probably located in the area hidden by $O2$ (i.e. an area of high occupancy probability is found in the fig 4-c3). Of course, the confidence in the presence of a hidden object (*i.e.* the values of the occupancy probability in the grid), progressively decreases when this object is not observed by the sensor during the next times steps. In the example depicted by fig 4-d3, the object is no longer hidden by $O2$; it is detected by the laser, and the related occupancy probability values increase.

Fig 5 shows another sequence of successive prediction and estimation results. The first row describes the experimental conditions : the *CyCab* is moving forward at a constant speed equal to $2.0m/s$; a static object is present in the sensor field of view the *CyCab*. Since the *CyCab* is moving forward, this object finally goes out of the sensor field of view, as depicted in fig 5-d1. The second and the third row respectively show the results of the prediction and of the estimation steps. We have chosen to only represent the cells of the grid corresponding to a relative speed equal to $(\dot{x} = -2.0m/s, \dot{y} = 0.0m/s)$, which is close to the relative object/*CyCab*. As in the case of the example of fig 4, the prediction step allows to infer knowledge on the current occupancy state of the *CyCab* environment, even if the object is no longer observed by the sensor; this is the situation depicted by

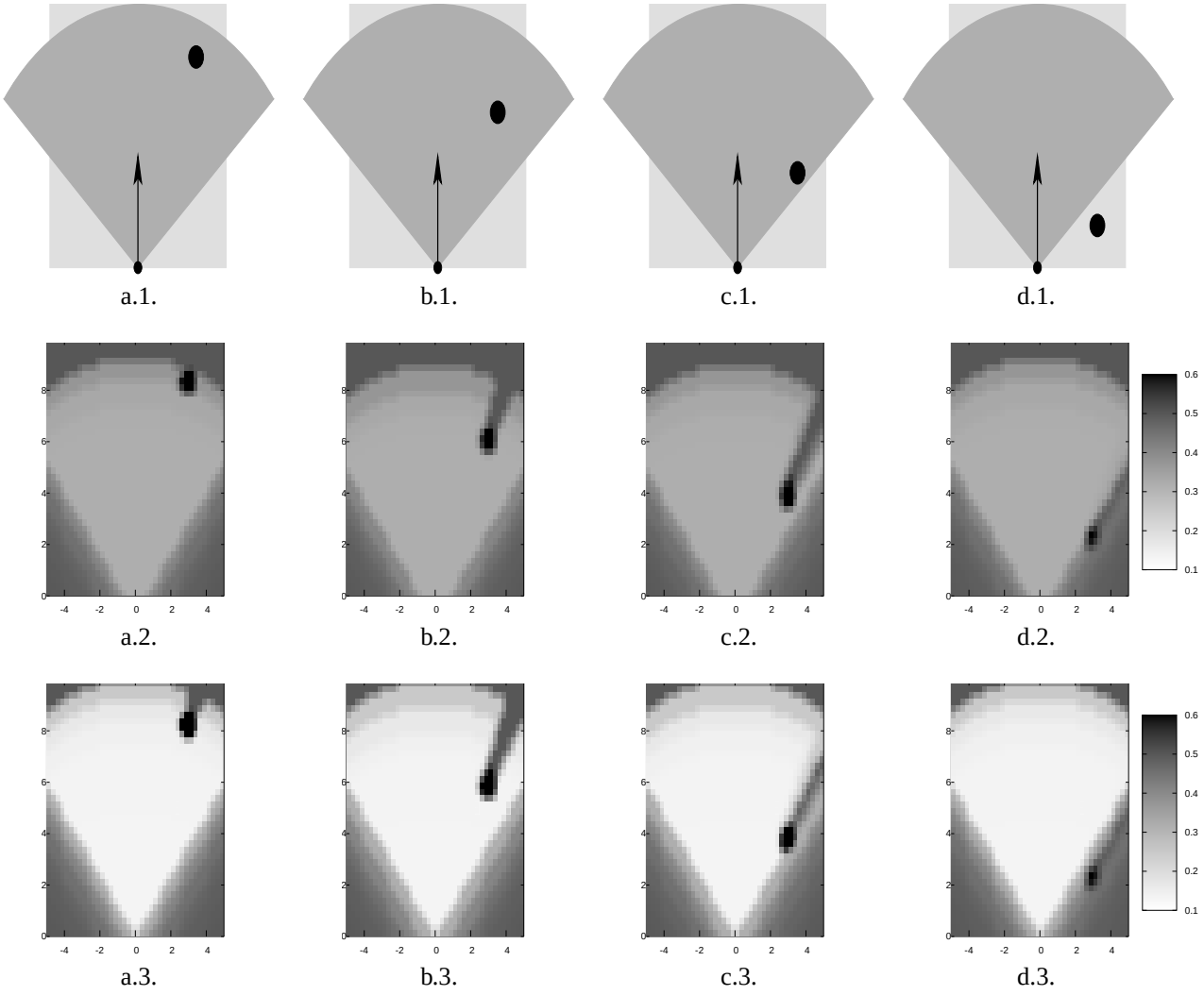


Fig. 5. A short sequence of a dynamic scene. The *CyCab* is moving forward at a constant speed. The grids show $P([E_C^k = 1] \mid x\ y\ [\dot{x} = -2.0]\ [\dot{y} = 0.0])$

fig 5-d3, where an area of high occupancy probability still exists when the object is going out of the sensor field of view. In some sense, our prediction step can be seen as a "short-term memory", which allows to combine in an evolutive way past and current observations.

E. Performance

As mentioned earlier, thanks to our approximation algorithm, both the prediction step and the estimation step complexities *increases linearly with the number of cells* of the grid. This make the approach tractable in real situations involving reasonnable grid sizes. This is the case for the experimental examples described in this section and in the next section; typically, the characteristics of the related grid models are the followings:

- 0 to $10\ m$ in the X direction, with a $0.5\ m$ resolution;
- -5 to $5\ m$ in the Y direction, with a $0.5\ m$ resolution;
- -3 to $1\ m.s^{-1}$ in the $\dot{X}$ direction, with a $0.4\ m.s^{-1}$ resolution;
- -3.2 to $3.2\ m.s^{-1}$ in the $\dot{Y}$ direction, with a $0.4\ m.s^{-1}$ resolution.

Using such a grid of 64.000 cells, the computation time for both prediction and estimation steps is about $100\ ms$ on a $1\ GHz$ computer. This is fast enough to control the *CyCab* at a maximum speed of $2\ m.s^{-1}$.

However, this grid size is not fine enough for an automotive application involving higher speeds. In this case, the number of cells increases quickly, and the required computational time becomes to high for satisfying the

real-time constraints. For instance, doubling the size of the grid in all directions, will result in a global size of $1.024.000$ cells, and in a required computation time of about $2.400\ ms$ (using a single $1\ GHz$ processor).

Hopefully, our *BOF* approach has been designed in order to be *highly parallelisable* : thanks to the hypothesis that each cell is independent, the state of each cell can be computed independently. Current work deals with the development of a dedicated hardware that will exploit this characteristic (and consequently allows a real-time application of the *BOF* in large scale applications).

This section has described the way the system operates for continuously interpreting the state of the environment using the *BOF*. The next section explains how we have exploited this information for implementing *safe basic behaviors* on an autonomous robot (and more precisely on the *CyCab*).

IV. BOF BASED COLLISION AVOIDANCE

The goal of this section is to show how the *BOF* can been used for developing a collision avoidance function on an autonomous vehicle. This function has been implemented and experimented on the *CyCab* experimental vehicle; the goal of the implemented function is to continuously select the forward speed values, in order to move safely (i.e. while avoiding moving obstacles such as pedestrians or other cars) along a given road lane.

As mentioned earlier, the cell state can be used to encode some relevant properties of the robot environment (e.g. occupancy, observability, reachability, etc). In the previous sections, only the occupancy characteristic was stored; in the case of the vehicle application, we will also encode the *danger* property. This will lead us to control the vehicle by combining *occupancy and danger* criteria.

A. Danger estimation

For each cell of the grid, the probability that this cell is hazardous is estimated; this estimation is done independantly of the occupancy probability property. Let $P(D_C^k \mid C^k)$ be the probability distribution associated to the cell C^k of the vehicle environment, where D_X^k is a boolean variable that indicates whether this cell is hazardous or not.

Basically, both "time to collision" and "safe traveling distance" may be seen as two complementary relevant criteria to be used for estimating the danger to associate to a given cell. In our current implementation, we are using the following related criteria which can easily been computed : (1) the *Closest Point of Approach (CPA)*, which defines the relative positions of the pair (vehicle, obstacle) corresponding to the "closest admissible distance" (i.e. safety distance); (2) the *Time to the Closest Point of Approach (TCPA)*, which is the time required for reaching the CPA; and (3) the *Distance at the Closest Point of Approach (DCPA)*, which is the distance separating the vehicle and the obstacle when the CPA has been reached. In some sense, these criteria give an assessment of the future relative trajectories of any pair of environment components of the type (vehicle, potential obstacle).

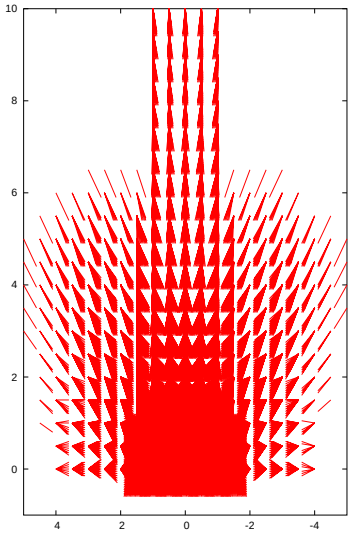


Fig. 6. Cells of high danger probabilities. For each position, arrows model the speed.

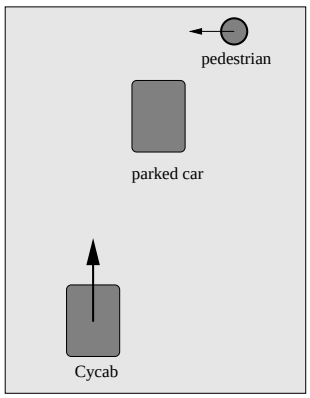


Fig. 7. Scenario description : the pedestrian is temporary hidden by a parked car.

The previous criteria are evaluated for each cell, and at each time step k, by taking into account the dynamics characteristics of both the vehicle and the potential obstacles. In practice, both $TCPA$ and $DCPA$ are estimated under the hypothesis that the related velocities at time k remain constant; this computation can easily be one using some classical geometrical algorithms (see for instance : `http://softsurfer.com/algorithms.htm`).

As previously mentionned, our goal is to estimate the "danger probability" to associate to each cell of the grid (or in other terms, the probability for each cell C^k that

Fig. 8. Snapshots of the experimental pedestrian avoidance scenario (see Extension 1 for the video).

a collision will occur in the near future between the *CyCab* and a potential obstacle in C^k). Since each cell C^k represents a pair (position, velocity) defined relatively to the *CyCab*, it is easy to compute the $TCPA$ and $DCPA$ factors, and in a second step to estimate the associated *danger probability* using given intuitive user knowledge. In the current implementation, this knowledge roughly states that when the $DCPA$ and the $TCPA$ decrease, the related probability of collision increases. In a future version of the system, one could expect that such a knowledge could be acquired using a learning phase.

Fig 6 shows the cells for which the danger probability is greater than 0.7 in our *CyCab* application; in the picture, each cell is represented by an arrow : the tail of the arrow indicates the position, and its length and direction indicates the associated relative speed. This figure exhibits quite intuitive data : any cell located in the vicinity of the front part of the *CyCab* are considered as having a high danger probability for any relative velocity (the arrows are pointing in all directions); the other cells having a high "oriented" danger probability, are those having a relative speed vector oriented towards the *CyCab*. Since we only consider relative speeds for constructing the danger grid, the content of this grid does not depend of the actual *CyCab* velocity.

B. Collision avoidance behaviors

This section describes how we can control the longitudinal speed of the autonomous vehicle (the *CyCab*), for avoiding partially observed moving obstacles having a high probability of collision with the vehicle. The implemented behavior consists in braking or accelerating, in order to adapt the velocity of the vehicle to the level of risk estimated by the system.

As mentionned earlier, this behavior derives from the combination of two criteria defined on the grid : the *danger probability* associated to each cell C^k of the grid (characterized by the distribution $P(D_C^k \,|\, C^k)$, see § IV-A), and the *occupancy probability* of this cell (characterized by the posterior distribution $P(E_C^k \,|\, \mathcal{Z}^k\, C^k)$, see § III). In practice, we search, at each time step, for the most hazardous cell that is considered as probably occupied; this can be done using the following equation :

$$\max_{C^k} \{P(D_C^k \mid C^k),\ with\ P(E_C^k \mid C^k) > 0.5\}.$$

Then the longitudinal acceleration/deceleration to apply to the *CyCab* controller, can be decided according to the obtained level of danger and to the actual velocity of the *CyCab*.

Fig 7 depicts the scenario used for experimentally validating the previous collision avoidance behavior on the *CyCab*; in this scenario, the *CyCab* is moving forward, the pedestrian is moving from right to left, and during a small period of time the pedestrian is temporarily hidden by a parked car.

Fig 8 shows some snapshots of the experiment (see also Extension 1, which shows the entire video): the *CyCab* brakes to avoid the pedestrian which is temporarily hidden by the parked car, then it accelerates as soon as the pedestrian has crossed the road.

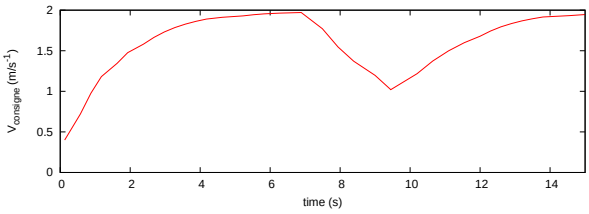


Fig. 9. Velocity of the *CyCab* during the experiment involving a pedestrian occlusion.

Fig 9 shows the velocity of the *CyCab* during this experiment. From $t = 0\ s$ to $t = 7\ s$, the *CyCab* accelerates, up to $2\ m/s$. At $t = 7s$, the pedestrian is detected; as a collision could possibly occur, the *CyCab* decelerates. From $t = 8.2\ s$ to $t = 9.4\ s$, the pedestrian is hidden by the parked car; thanks to the BOF results, the hazardous cells of the grid are still considered as probably occupied; as a consequence, the *CyCab* still brakes. When the pedestrian reappears at $t = 9.4s$, there is no more a risk of collision, and the *CyCab* can accelerate.

V. Conclusions

This paper addressed the problem of designing a new approach for robust perception and danger assesment of

highly dynamic environments. The proposed approach is called *Bayesian Occupancy Filtering*; it basically combines a 4-dimensionnal occupancy grid representation of the obstacle state-space with Bayesian filtering techniques. This approach can be seen as an alternative to complex multi-target tracking algorithms, which usualy fail in situations involving numerous appearances, disappearances and occlusions of a large number of rapidly maneuvering targets. It also bring a significant improvement to traditional approaches, by including a prediction step which allows to make more robust estimation relatively to temporary occlusions.

This approach has experimentally been validated on our experimental vehicle (the Cycab), for avoiding partially observed moving obstacles. A scenario involving the Cycab, a moving pedestrian, and a parked car which temporarily hidde the pedestrian to the sensors of the Cycab, has sussfully been executed. In this experiment, the avoidance behavior has been obtained by combining the *occupancy probability* and the *danger probability* of each cell of the grid.

Current and future work deals with three major points: (a) Improvement of the approximation algorithm used for the prediction step; this improvement should allow to estimate bigger grids (which are required for dealing with complex urban trafic situations). (b) Development of a dedicated hardware that will exploit the parallelisable property of the BOF algorithm, for being able to meet the real-time contraint in large sacle applications. (C) Fusion of the occupancy grid with higher-level information, such as GPS maps, to better estimate the danger of the situation.

Acknowledgements. This work was partially supported by the European project IST-1999-12224 *"Sensing of Car Environment at Low Speed Driving"*(`http://www.carsense.org`).

VI. REFERENCES

[1] Y. Bar-Shalom and X. Li. *Multitarget Multisensor Tracking : Principles and Techniques*. YBS Publishing, 1995.

[2] H. Gauvrit, J.P. Le Cadre, and C Jauffret. A formulation of multitarget tracking as an incomplete data problem. *IEEE Trans. on Aerospace and Electronic Systems*, 33(4), 1997.

[3] R.L. Streit and T.E. Luginbuhl. Probabilistic multi-hypothesis tracking. Technical Report 10,428, Naval Undersea Warfare Center Division Newport, 1995.

[4] S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems*. Artech House, 2000.

[5] C-C Wang, C. Thorpes, and S. Thrun. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. In *Proc of the IEEE Int Conf on Robotics and Automation*, pages 842–849, Taipei (taiwan), September 2003.

[6] H.P. Moravec. Sensor fusion in certainty grids for mobile robots. *AI Magazine*, 9(2), 1988.

[7] A. Elfes. Using occupancy grids for mobile robot perception and navigation. *IEEE Computer, Special Issue on Autonomous Intelligent Machines*, Juin 1989.

[8] S. Thrun. Robotic mapping: A survey. In *Exploring Artificial Intelligence in the New Millenium*. Morgan Kaufmann, 2002.

[9] E. Prassler, J. Scholz, and A. Elfes. Tracking multiple moving objects for real-time robot navigation. *Autonomous Robots*, 8(2), 2000.

[10] K. Mekhnacha, E. Mazer, and P. Bessière. The design and implementation of a bayesian CAD modeler for robotic applications. *Advanced Robotics*, 15(1):45–70, 2001.

[11] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1), 1998.

[12] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101, 1998.

[13] K.O. Arras, N. Tomatis, and R. Siegwart. Multisensor on-the-fly localization : precision and reliability for applications. *Robotics and Autonomous Systems*, 44:131–143, 2001.

[14] A. H. Jazwinsky. *Stochastic Processes and Filtering Theory*. New York : Academic Press, 1970.

[15] O. Lebeltel. *Programmation Bayésienne des Robots*. Thèse de doctorat, Institut National Polytechnique de Grenoble, Grenoble, France, Septembre 1999.

[16] O. Lebeltel, P. Bessière, J. Diard, and E. Mazer. Bayesian robot programming. *Autonomous Robots*, 16:49–79, 2004.

[17] C. Coué and P. Bessière. Chasing an elusive target with a mobile robot. In *Proceedings of the IEEE-RSJ International Conference on Intelligent Robots and Systems*, Hawai (HI), 2001.

[18] R. Le Hy, A. Arrigoni, P. Bessière, and O. Lebeltel. Teaching bayesian behaviours to videogame characters. In *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Las Vegas (NV), October 2003.

[19] G. Cooper. The computational complexity of probabilistic inference using bayesian belief network. *Artificial Intelligence*, 42(2-3), 1990.

[20] R.E. Kalman. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 35, Mars 1960.

[21] G. Welch and G. Bishop. An introduction to the Kalman filter. available at http://www.cs.unc.edu/~welch/kalman/index.html.