Grid-based Localization and Online Mapping with Moving Object Detection and Tracking

Julien Burlet, Trung Dung Vu, Olivier Aycard

Grid-based Localization and Online Mapping with Moving Object Detection and Tracking

Julien Burlet, Trung-Dung Vu, Olivier Aycard

LIG & INRIA Rhône Alpes, Grenoble, France
firstname.lastname@inrialpes.fr

Abstract

In this paper, we present a real-time algorithm for local simultaneous localization and mapping (SLAM) with detection and tracking of moving objects (DATMO) in dynamic outdoor environments from a moving vehicle equipped with laser sensor, radar and odometry. To correct vehicle location from odometry we introduce a new fast implementation of incremental scan matching method that can work reliably in dynamic outdoor environments. After a good vehicle location is estimated, the surrounding map of the vehicle is updated incrementally and moving objects are detected without a priori knowledge of the targets. Detected moving objects are finally tracked by a Multiple Hypothesis Tracker (MHT) coupled with an adaptive Interacting Multiple Models filter. The experimental results on datasets collected from different scenarios such as: urban streets, country roads and highways demonstrate the efficiency of the proposed algorithm.

Key words: occupancy grid, simultaneous localization and mapping, moving object detection, multiple object tracking, interacting multiple model, laser radar data fusion

1 INTRODUCTION

Perceiving or understanding the environment surrounding of a vehicle is a very important step in driving assistant systems or autonomous vehicles. The task involves both simultaneous localization and mapping (SLAM) and detection and tracking of moving objects (DATMO). While SLAM provides the vehicle with a map of static parts of the environment as well as its location in the map, DATMO allows the vehicle being aware of dynamic entities around, tracking them and predicting their future behaviors. It is believed that if we are able to accomplish both SLAM and DATMO reliably in real time, we can detect every critical situations to warn the driver in advance and this will certainly improve driving safety and can prevent traffic accidents.

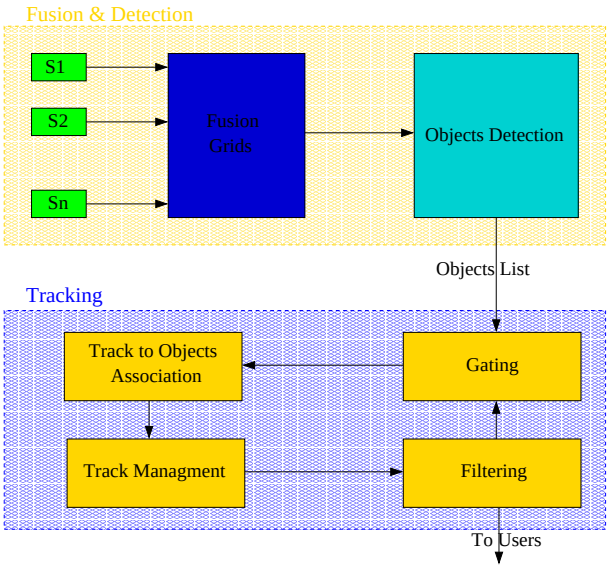

Fig. 1. Architecture of the perception system

Recently, there have been considerable research efforts focusing on SLAM and DATMO [1][2][3][4]. However, for highly dynamic outdoor environments like crowded urban streets, there still remains many open questions. These include, how to represent the vehicle environment, how to obtain a precise location of the vehicle in presence of dynamic entities, and how to differentiate moving objects and stationary objects as well as how to track moving objects over time.

In this context, we design and develop a generic architecture to solve SLAM and DATMO in dynamic outdoor environments. The architecture (Fig. 1) is divided into two main parts: the first part where the vehicle environment is mapped and moving objects are detected; and the second part where previously detected moving objects are verified and tracked. This architecture has been used in past projects [5], and in this paper, we detail the implementation on a vehicle moving at high speed equipped with laser scanner, radar and odometry. This work has been carried out in the framework of European project PReVENT-ProFusion [1] .

In the first part of the architecture, to model the environment surrounding the vehicle, we use the Occupancy Grid framework developed by Elfes [6]. Compared with feature-based approaches [7], grid maps can represent any environment and are specially suitable for noisy sensors in outdoor environments where features are hard to define and extract. Grid-based approaches also provide an interesting mechanism to integrate different kinds of sensors in the same framework taking the inherent uncertainty of each sensor reading into account.

In general, in order to perform mapping or modeling the environment from a moving vehicle, a precise vehicle localization is essential. To correct vehicle loca-

[1] www.prevent-ip.org/profusion

tions from odometry, we introduce a new fast laser-based incremental localization method that can work reliably in dynamic environments. When good vehicle locations are estimated, by integrating laser measurements we are able to build a consistent grid map surrounding of the vehicle. Finally by comparing new measurements with the previously constructed local vehicle map, dynamic objects then can be detected.

In the second part, detected moving objects in the vehicle environment are tracked. Since some objects may be occluded or some are false alarms, multi object tracking helps to identify occluded objects, recognize false alarms and reduce misdetections. In general, the multi object tracking problem is complex: it includes the definition of filtering methods, but also association methods and maintenance of the list of objects currently present in the environment [8][9]. Regarding tracking techniques, Kalman filters [10] or particle filters [11] are generally used. These filters require the definition of a specific dynamic model of tracked objects. However, defining a suitable motion model is a real difficulty. To deal with this problem, Interacting Multiple Models [12][13] have been successfully applied in several applications. In previous works [14][15], we have developed a fast method to adapt on-line IMM according to trajectories of detected objects and so we obtain a suitable and robust tracker. To deal with the association and maintenance problem, we extend our approach to multiple objects tracking using the Multiple Hypothesis Tracker [16][17].

Demonstrator vehicle

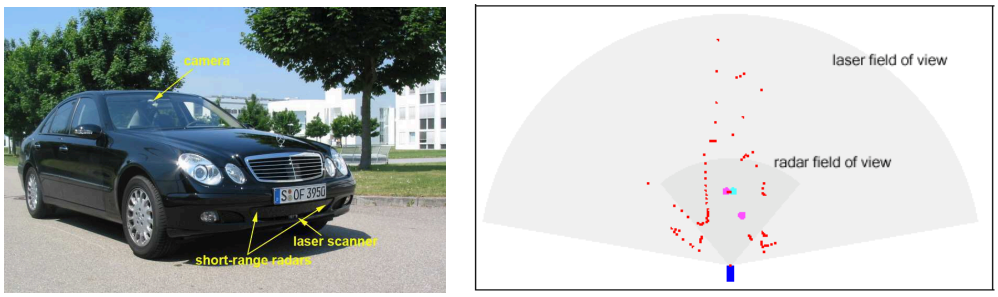

Fig. 2. Left: the DaimlerChrysler demonstrator car. Right: an example of sensor data, laser measurements are displayed in small red dots and radar measurements displayed as bigger dots.

The proposed algorithm for solving SLAM and DATMO is tested on data collected from DaimlerChrysler demonstrator car equipped with a camera, two short range radar sensors and a laser scanner (Fig. 2 left). The laser scanner can detect obstacles at a range of 70 m under a field of view of 160°. The laser scanner provide raw data as a list of impacts with an angular resolution of 1°. The radar sensors detect typical targets up to 30 m within a field of view of 80° and return pre-filtered data as a list of objects comprised of the estimated object positions and Doppler velocities (Fig.

2 right). In addition, vehicle odometry information such as velocity and yaw rate are provided by the vehicle sensors. The measurement cycle of the sensor system is 40 ms.

In our implementation, laser data is used to perform mapping as well as detection and tracking of moving object. Radar data is then fused with the results of object detection in order to provide supplemental information on velocities of objects detected in the radar field of view. Images from camera are only for visualization purpose. Experimental results show that our algorithm can perform both SLAM and DATMO in real time for different types of dynamic outdoor environments.

The rest of the paper is organized as follows. In the next section we describe our approach to mapping and localization in dynamic outdoor environments. Algorithm for detecting moving objects is presented in Section 3. Multi objects tracking approach is detailed in Section 4. Experimental results are reported in Section 5 and finally conclusions and future works are given in Section 6.

2 LOCAL SLAM

Since our radar sensors provide pre-filtered data at object level as lists of target points, so to perform mapping, only laser data is used. To our safety vehicle navigation purpose, a good global map is not necessary, so that the problem of revisiting or loop closing in SLAM is not considered in this work. For this reason, we propose an incremental mapping approach based on a fast laser scan matching algorithm in order to build a consistent local vehicle map. The map is updated incrementally when new data measurements arrive along with good estimates of vehicle locations obtained from the scan matching algorithm. The advantages of our incremental approach are that the computation can be carried out very quickly and the whole process is able to run online.

2.1 Notation

Before describing our approach in detail, we introduce some notations used in the paper. We denote the discrete time index by the variable t, the laser observation from vehicle at time t by the variable $z_t = \{z_t^1, ..., z_t^K\}$ including K individual measurements corresponding to K laser beams, the vector describing an odometry measurement from time $t-1$ to time t by the variable u_t, the state vector describing the true location of the vehicle at time t by the variable x_t.

In this representation, the vehicle environment is divided into a two-dimensional lattice M of rectangular cells and each cell is associated with a measure taking a real value in $[0,1]$ indicating the probability that the cell is occupied by an obstacle. A high value of occupancy grid indicates the cell is occupied and a low value means the cell is free. Assuming that occupancy states of individual grid cells are independent, the objective of a mapping algorithm is to estimate the posterior probability of occupancy $P(m\,|\,x_{1:t}, z_{1:t})$ for each cell m of the grid, given observations $z_{1:t} = \{z_1, ..., z_t\}$ at corresponding known poses $x_{1:t} = \{x_1, ..., x_t\}$.

Using Bayes theorem, this probability is determined by:

$$P(m\,|\,x_{1:t}, z_{1:t}) = \frac{P(z_t\,|\,x_{1:t}, z_{1:t-1}, m)\,.\,P(m\,|\,x_{1:t}, z_{1:t-1})}{P(z_t\,|\,x_{1:t}, z_{1:t-1})} \tag{1}$$

If we assume that current measurement z_t is independent from $x_{1:t-1}$ and $z_{1:t-1}$ given we know m, $P(z_t\,|\,x_{1:t}, z_{1:t-1}, m) = P(z_t\,|\,x_t, m)$. Then after applying Bayes Theorem to $P(z_t\,|\,x_t, m)$, equation (1) becomes:

$$P(m\,|\,x_{1:t}, z_{1:t}) = \frac{P(m\,|\,x_t, z_t)\,.\,P(z_t\,|\,x_t)\,.\,P(m\,|\,x_{1:t}, z_{1:t-1})}{P(m)\,.\,P(z_t\,|\,x_{1:t}, z_{1:t-1})} \tag{2}$$

Equation (2) gives the probability for an occupied cell. By analogy, equation (3) gives the probability for a free cell:

$$P(\overline{m}\,|\,x_{1:t}, z_{1:t}) = \frac{P(\overline{m}\,|\,x_t, z_t)\,.\,P(z_t\,|\,x_t)\,.\,P(\overline{m}\,|\,x_{1:t}, z_{1:t-1})}{P(\overline{m})\,.\,P(z_t\,|\,x_{1:t}, z_{1:t-1})} \tag{3}$$

By dividing equation (2) by (3), we obtain:

$$\frac{P(m\,|\,x_{1:t}, z_{1:t})}{P(\overline{m}\,|\,x_{1:t}, z_{1:t})} = \frac{P(m\,|\,x_t, z_t)}{P(\overline{m}\,|\,x_t, z_t)} \cdot \frac{P(\overline{m})}{P(m)} \cdot \frac{P(m\,|\,x_{1:t-1}, z_{1:t-1})}{P(\overline{m}\,|\,x_{1:t-1}, z_{1:t-1})} \tag{4}$$

If we define $Odds(x) = \frac{P(x)}{P(\overline{x})} = \frac{P(x)}{1-P(x)}$, equation (4) turns into:

$$Odds(m\,|\,x_{1:t}, z_{1:t}) = Odds(m\,|\,x_t, z_t).Odds(m)^{-1}.Odds(m\,|\,x_{1:t-1}, z_{1:t-1}) \tag{5}$$

The corresponding *log Odds* representation of equation (5) is:

$$\begin{aligned} &log\,Odds(m\,|\,x_{1:t}, z_{1:t}) \\ &\qquad = log\,Odds(m\,|\,z_t, x_t) - log\,Odds(m) + log\,Odds(m\,|\,x_{1:t-1}, z_{1:t-1}) \end{aligned} \tag{6}$$

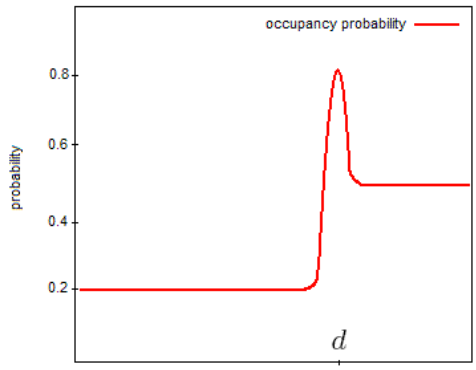

Fig. 3. Profile of an inverse sensor model illustrates the occupancy probability along a laser beam measuring a distance of d.

In (6), what we need to know are two probability densities, $P(m|x_t,z_t)$ and $P(m)$. $P(m)$ is the prior occupancy probability of the map cell which is set to 0.5 representing an unknown state, that makes this component disappear. The remaining probability $P(m|x_t,z_t)$, is called the *inverse sensor model*. It specifies the probability that a grid cell m is occupied based on a single sensor measurement z_t at location x_t. Fig. 3 shows the function we use to compute the occupancy probability of grid cells along a laser beam measuring a distance of d.

From the *log Odds* representation, the desired probability of occupancy $P(m|x_{1:t},z_{1:t})$ can be easily recovered. And since the updating algorithm is recursive, it allows for the map updated incrementally when new sensor data arrives.

The second image in Fig 6b shows an example of an occupancy grid map constructed from laser measurements during the vehicle's movement. The color of grid map cell indicates the probability that corresponding space being occupied: gray=unknown, white=free, black=occupied.

2.3 Localization in Occupancy Grid Map

In order to build a consistent map of the environment, a good vehicle localization is required. Because of the inherent error, using only odometry often results in an unsatisfying map (see Fig. 4 left). When features can not be defined and extracted, direct scan matching techniques like ICP [18] can help to correct the odometry error. The problem is that sparse data in outdoor environments and dynamic entities make correspondence finding difficult. One important disadvantage of the direct scan matching methods is that they do not consider the dynamics of the vehicle. Indeed we have implemented several ICP variants [19] and found out that scan matching results are unsatisfactory and often lead to unexpected trajectories of vehicle. This is because matching only two consecutive scans may be very hard, ambiguous or weakly constrained, especially in outdoor environment and when the vehicle moves at high speeds.

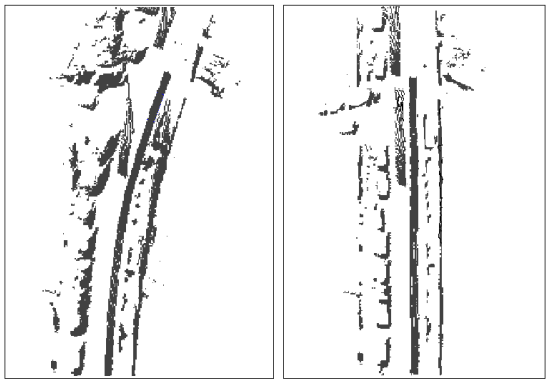

Fig. 4. Hit maps build directly from raw laser data collected from a vehicle moving along a straight street: with vehicle localization using odometry (left); and using results of scan matching (right). Note that the scan matching results are not affected by moving objects in the street.

An alternative approach that can overcome these limitations consists in setting up the matching problem as a maximum likelihood problem (see more detail in our previous work [20]). In this approach, given an underlying vehicle dynamics constraint, the current scan's position is corrected by comparing with the local grid map constructed from all observations in the past instead of only with one previous scan. By this way, we can reduce the ambiguity and weak constraint especially in outdoor environment and when the vehicle moves at high speeds. Mathematically, we calculate a sequence of poses $\hat{x}_1, \hat{x}_2, \ldots$ and sequentially updated maps $M_1, M_2, \ldots$ by maximizing the marginal likelihood of the t-th pose and map relative to the $(t-1)$-th pose and map:

$$\hat{x}_t = \underset{x_t}{\operatorname{argmax}} \left\{ P(z_t \,|\, x_t, M_{t-1}) \,.\, P(x_t \,|\, \hat{x}_{t-1}, u_t) \right\} \tag{7}$$

In the equation (7), the term $P(z_t \,|\, x_t, M_{t-1})$ is the measurement model which is the probability of the most recent measurement z_t given the pose x_t and the map M_{t-1} constructed so far from observations $z_{1:t-1}$ at corresponding poses $\hat{x}_{1:t-1}$ that were already estimated in the past. The term $P(x_t \,|\, \hat{x}_{t-1}, u_t)$ represents the motion model which is the probability that the vehicle is at location x_t given that the vehicle was previously at position $\hat{x}_{t-1}$ and executed an action u_t. The resulting pose $\hat{x}_t$ is then used to generate a new map M_t according to (6):

$$M_t = M_{t-1} \cup \{\hat{x}_t, z_t\} \tag{8}$$

Now the question is how to solve the equation (7), but let us first describe the motion model and the measurement model used.

For the motion model, we adopt the probabilistic velocity motion model similar to that of [9]. The vehicle motion u_t is comprised of two components, the translational velocity v_t and the yaw rate ω_t. Fig. 5 depicts the probability of being at location x_t

given previous location x_{t-1} and control u_t. This distribution is obtained from the kinematic equations, assuming that vehicle motion is noisy along its rotational and translational components.

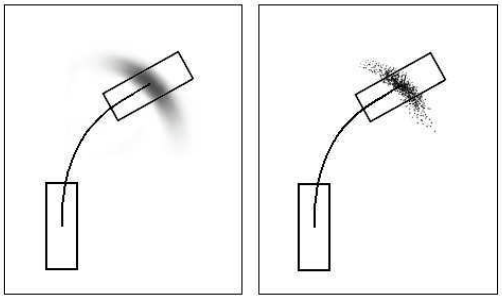

Fig. 5. The probabilistic velocity motion model $P(x_t \mid x_{t-1}, u_t)$ of the vehicle (left) and its sampling version (right).

For the measurement model $P(z_t \mid x_t, M_{t-1})$, mixture beam-based model is widely used in the literature [21][2]. However, the model come at the expense of high computation since it requires ray casting operation for each beam. This can be a limitation for real time application if we want to estimate a large amount of measurements at the same time. To avoid ray casting, we propose an alternative model that only considers end-points of the beams. Because it is likely that a beam hits an obstacle at its end-point, we focus only on occupied cells in the grid map. A voting scheme is used to compute the probability of a scan measurement z_t given the vehicle pose x_t and the map M_{t-1} constructed so far. First, from the vehicle location x_t, individual measurement z_t^k is projected into the coordinate space of the map. Call hit_t^k the grid cell corresponding to the projected end-point of each beam z_t^k. If this cell is occupied, a sum proportional to the occupancy value of the cell will be voted. Then the final voted score represents the likelihood of the measurement. Let $P(M_t^i)$ denote the posterior probability of occupancy of the grid cell M^i estimated at time t (following (6)), we can write the measurement model under the sum following:

$$P(z_t \mid x_t, M_{t-1}) \propto \sum_{k=1}^{K} \{ P(M_{t-1}^{hit_t^k}) \text{ so that } M_{t-1}^{hit_t^k} \text{ is occupied} \} \tag{9}$$

The proposed method is just an approximation to the measurement model because it does not take into account visibility constraints, but experimental evidences show that it works well in practice. Furthermore, with a complexity of $O(K)$, the computation can be done rapidly.

It remains to describe how we maximize (7) to find the correct pose $\hat{x}_t$. Hill climbing strategy in [22][2] can be used but may suffer from a local maximum. Exploiting the fact that the measurement model can be computed very quickly, we perform an extensive search over vehicle pose space. A sampling version of the motion model (Fig. 5 right) is used to generate all possible poses x_t given the previous pose x_{t-1} and the control u_t. The resulting pose will be the pose at which the measurement probability achieves a maximum value. Because of the inherent discretization of

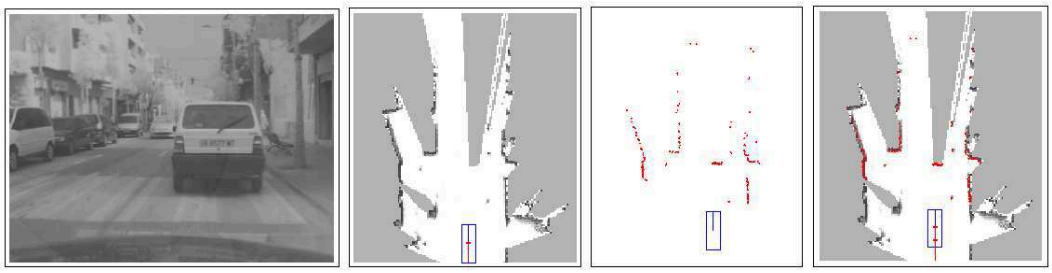

Fig. 6. An example of scan matching. From left to right: reference image; map constructed so far M_{t-1} with previous vehicle location x_{t-1}; new laser measurement z_t; and matching result is obtained by trading off the consistency of the measurement with the map and the previous vehicle pose.

the grid, the sampling approach turns out to work very well. In practice, with a grid map resolution of 20 cm, it is enough to generate about four or five hundreds of pose samples to obtain a good estimate of the vehicle pose with the measurement likelihood that is nearly unimproved even with more samples. The total computational time needed for such a single scan matching is about 10 ms on a low-end PC. An example of scan matching result is shown in Fig. 6. The most likely vehicle pose is obtained when the laser scan is aligned with the occupied parts of the map and at the same time the vehicle dynamics constraint is satisfied.

Besides the computational effectiveness, one attraction of our algorithm is that it is not affected by dynamic entities in the environment (see Fig. 4 right). Since we only consider occupied cells, spurious regions in the occupancy grid map with low occupancy probability that might belong to dynamic objects do not contribute to the sum (9). Since a large part of measurements belong to static objects, the voting scheme ensures that measurement likelihood reach a maximum only when the laser scan is aligned with the static parts of the environment. To some meaning, measurements from dynamic entities can be considered as outliers of the alignment process. This property is very useful for moving object detection process that will be described in the next section.

2.4 Local mapping

Because we do not need to build a global map nor deal with loop closing problem, only one online map is maintained at each point in time representing the local environment surrounding of the vehicle. The size of the local map is chosen so that it should not contain loops and the resolution is maintained at a reasonable level. Every time the vehicle arrives near the map boundary, a new grid map is reinitialized. The pose of the new map is computed according to the vehicle global pose and cells inside the intersection area are copied from the old map.

3 MOVING OBJECTS DETECTION

In the previous section, we represent how to obtain precise vehicle localization and how to build local vehicle map from laser data. In this section we will describe how to identify moving objects by using previously constructed local map. Detected objects can then be confirmed using radar data and are provided with their velocities.

3.1 Using Occupancy Grid to detect Moving Objects

After a consistent local grid map of the vehicle is constructed, moving objects can be detected when new laser measurements arrive by comparing with the previously constructed grid map. The principal idea is based on the inconsistencies between observed free space and occupied space in the local map. If an object is detected on a location previously seen as free space, then it is a moving object. If an object is observed on a location previously occupied then it probably is static. If an object appears in a previously not observed location, then it can be static or dynamic and we set the unknown status for the object in this case.

Another important clue which can help to decide whether an object is dynamic or not is evidence about moving objects detected in the past. For example, if there are many moving objects passing through an area then any object that appears in that area should be recognized as a potential moving object. For this reason, in addition to the local static map M constructed as described in the previous section, a local dynamic grid map D is created to store information about previously detected moving objects. The pose, size and resolution of the dynamic map is the same as those of the static map. Each dynamic grid cell store a value indicating the number of observations that a moving object has been observed at that cell.

From these remarks, our moving object detection process is carried out in two steps as follows. The first step is to detect measurements that might belong to dynamic objects. Here for simplicity, we will temporarily omit the time index. Given a new laser scan z, the corrected vehicle location and the local static map M and the dynamic map D containing information about previously detected moving objects, state of a single measurement z^k is classified into one of three types following:

$$state(z^k) = \begin{cases} static & \text{if } M^{hit^k} = occupied \\ dynamic & \text{if } M^{hit^k} = free \text{ or } D^{hit^k} > \alpha \\ undecided & \text{if } M^{hit^k} = unknown \end{cases}$$

where M^{hit^k} and D^{hit^k} are the corresponding cells of the static and dynamic map respectively at the end-point hit^k of the beam z^k, α is a pre-defined threshold.

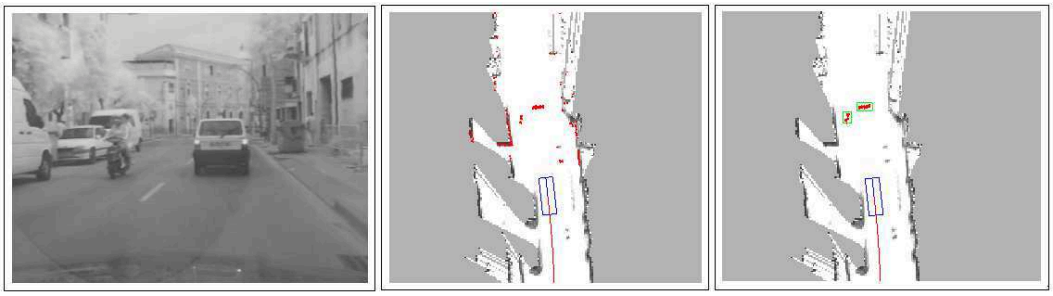

Fig. 7. Moving object detection example. See text for more details.

The second step is after measurements that might belong to dynamic objects are determined, moving objects are then identified by clustering end-points of these beams into separate groups, each group represents a single object. Two points are considered as belonging to the same object if the distance between them is less than 0.2 m.

Fig. 7 illustrates the described steps in detecting moving objects. The leftmost image depicts the situation where the vehicle is moving along a street seeing a car moving ahead and a motorbike moving in the opposite direction. The middle image shows the local static map and the vehicle location with the current laser scan drawn in red. Measurements which fall into free region in the static map are detected as dynamic and are displayed in the rightmost image. After the clustering step, two moving objects are identified (in green boxes) and correctly corresponds to the car and the motorbike.

3.2 Fusion with radar

After moving objects are identified from laser data, we confirm the object detection results by fusioning with radar data and provide the detected objects with their velocities. Within the radar module, a preprocessing of the radar measurements takes place, wherein reflections with a similar distance, relative velocity, and amplitude are grouped together. The radar sensors return pre-filtered data as lists of potential targets. The target lists of the two radar are independent from each other. Each target in the lists is provided with information about the location and the estimated Doppler velocity.

For each moving object detected from laser data as described in the previous section, a rectangular bounding box is calculated and the radar measurements which lie within the box region are then assigned to corresponding object. The velocity of the detected moving object is estimated as the average of these corresponding radar measurements.

Fig. 8 shows an example of how the fusion process takes place. Moving objects detected by laser data are displayed in red with green bounding boxes. The targets

detected by two radar sensors are represented as small circles in different colors along with corresponding velocities. We can see in the radar field of view, two objects detected by laser data are also seen by two radars so that they are confirmed and their velocities are estimated. Radar measurements that do not correspond any dynamic object and fall into other region of the grid are not considered.

Fig. 8. Moving object detected from laser data is confirmed by radar data.

Since the radar field of view is much smaller than laser field of view (Fig. 2 right), the purpose of fusion is only to provide supplemental velocity information of objects detected in the radar field of view.

4 MULTIPLE HYPOTHESIS TRACKING USING ADAPTIVE IMM

In the second part of our architecture, an adaptive MHT method is used to solve the association problem of detected objects with tracks, each track corresponds to a previously known moving object and modeled by a hypotheses tree. Also it permits to detect and reject spurious detected objects (caused by sensors' noise and by detection method) and to identify new moving objects incoming in the sensors' range.

4.1 Introduction

The aim of multi-object tracking is to estimate the number and the states of real objects evolving in the environment by generating and maintaining during time a set of tracked objects according to detected (observed) objects [2] obtained at each step of the process *via* the sensors. For convenience we call *track* a tracked object composed by a list of detected objects. The multi-object tracking problem is complex: it

[2] usually the term *observation* is used in such a case but as in our work raw sensors observations are treated to obtain *detections*, the term *detected object* will be use for more clarity

includes the definition of filtering methods, but also association methods and maintenance of the list of objects currently present in the environment. The most known techniques are the the Global Nearest Neigbour (GNN) combined with filtering, Joint Probabilistic Data Association Filter (JPDAF) and the Multiple Hypothesis Tracking (MHT) [8][9]. In the conventional GNN only the most likely assignment is considered at each step, allowing only to associate at most one detected object to one track. The JPDAF method permits to assign several detected objects to on track by weighted probabilistic sum. Nevertheless, it works with a fixed number of tracks and increase the track state uncertainty since several objects with different positions can update on unique track. In MHT alternative associations hypotheses are build over time. In conflict situations, instead of taking a decision (GNN) or combining hypotheses (JPDAF), hypotheses are propagated into the future in anticipation that it will resolve the association uncertainty.

The basic principle of MHT is to generate and update a set of association hypotheses during process. An hypothesis corresponds to a specific probable assignment of detected objects with tracks. By maintaining and updating several hypotheses, none irreversible association decisions are made and ambiguous cases are solved in further steps.

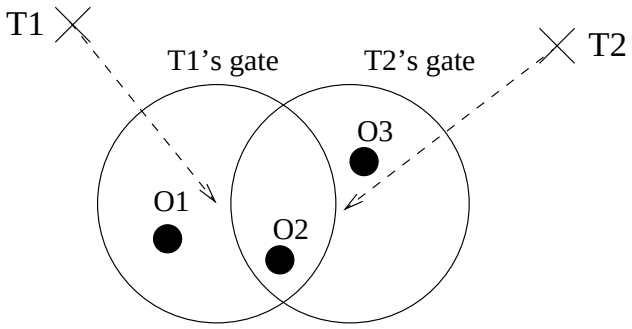

Fig. 9. Example of association problem

Reid introduces first a complete algorithm given a systematic way in which multiple data association hypotheses can be formed and evaluated for the problem of multiple target tracking [23]. Using Fig.9 as an example, the reid's algorithm principle is to consider tracks T_1 and T_2 obtained at the last iteration of the algorithm and their corresponding gates. Next, define a newly formed track $T_3(T_1, O_1)$ to be the track formed from the association of T_1 with O_1 and so one for each possible association. Also define NT_1, NT_2 and NT_3 to be new tracks initialized from each detected objects. Using this formalisation, the set of hypotheses can be formed. For instance H_4 is the hypothesis containing tracks T_1 and T_2 and assuming detectec objects are new tracks and so one for all the set of hypotheses.

$$
\begin{aligned}
H_1 &= T_3(T_1, O_1), T_5(T_2, O_3), NT_2 \\
H_2 &= T_6(T_1, O_2), T_5(T_2, O_3), NT_1 \\
H_3 &= T_3(T_1, O_1), T_7(T_2, O_2), NT_3 \\
H_4 &= T_1, T_2, NT_1, NT_2, NT_3 \\
&\ldots
\end{aligned} \tag{10}
$$

By following this method, we can systematically generate hypotheses. Also a probability $P(H_i)$ is computed for each hypotheses. $P(H_i)$ is obtained using the likelihood of detected objects with prediction, the false alarm probability and the non-detection probability.

As described, a simple example can lead to an important number of hypotheses. Indeed, in practical a straight forward implementation of the MHT will generate an combination explosion of the number of Hypothesis. To cope with, different techniques have been developed to reduce and control the number of hypothesis [17] such as clustering and hypothesis pruning depending on the hypothesis representation.

One possible representation of the MHT hypotheses and tracks structure is illustrated in the Fig. 10 [17]. In this figure, which takes as an example the problem in Fig. 9, tracks are represented by trees, each node of the trees corresponding to a possible data association, *e.g* for the first track, the new leafs are formed considering the mis-detection (O_0) and the two detected objects falling in its gate (O_1 and O_2). Then associations hypotheses at the current step are attached to the set of trees' leafs, on the Fig. 10, the four first hypotheses defined upper by the Reid's algorithm are shown.

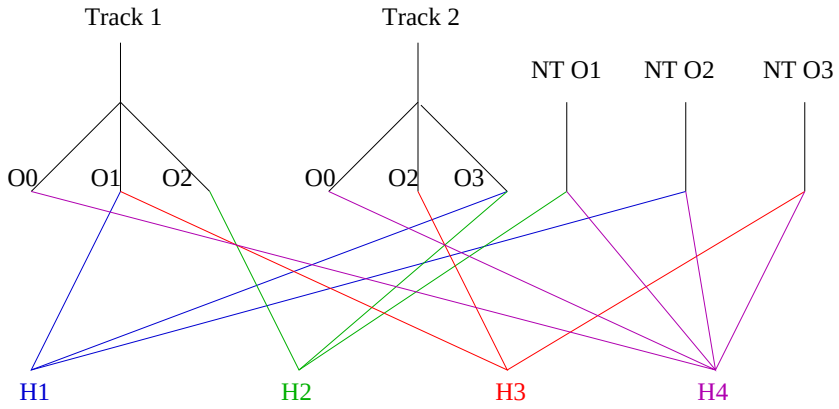

Fig. 10. Formation of hypothesis from track trees

As shown in Fig. 11, our multi-object tracking method is composed of four different parts:

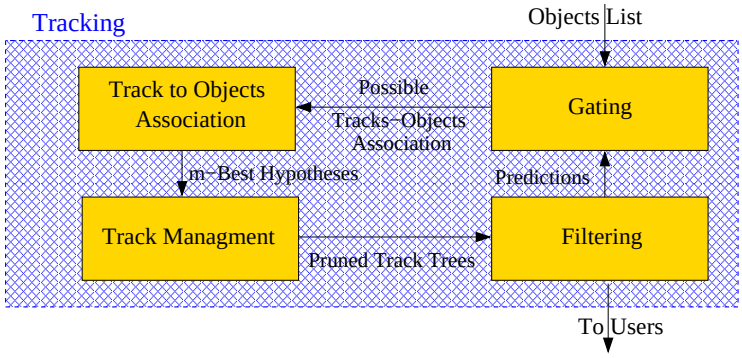

Fig. 11. Architecture of multi-object tracking system

- The first one is the gating. In this part, taking as input predictions from previous computed tracks, we compute the set of new detected objects which can be associated to each track.
- In a second part, using the result of the gating, we perform object to tracks association and generate association hypotheses, each track corresponding to a previously known moving object. Output is compoed of the computed set of association hypotheses.
- In the third part called track management, tracks are confirmed, deleted or created according to the association results and final track trees are output.
- In the last part corresponding to the filtering step, estimates are computed for 'surviving' tracks and predictions are performed to be used the next step of the algorithm. In this part we use an adaptive method based on Interacting Multiple Models (IMM).

More details about this different parts are outlined next.

4.2 Gating

In this part, taking as input predictions from previous computed tracks and newly detected objects, a gating is performed. It consists in, according to an arbitrary distance function, determine the detected objects which can be associated with tracks. Also during this stage, clustering is performed in order to reduce the number of association hypotheses. It consists in making clusters of tracks which share at least one detected object. In the next stage, association can be performed independently for each cluster decomposing a large problem in smaller problems which induce generation of less hypotheses.

If we take as an example the situation depict by the Fig. 9, in this stage one set is computed as T_1 and T_2 share object O_2. Also according to gates, objects O_1 and O_2 can be assigned to T_1 and objects O_2 and O_3 to T_3.

4.3 Association

In this part, taking as input clusters of tracks and detected objects validated by the gating stage, association hypotheses are evaluated. By considering likelihood of objects with tracks, new track apparition probability and non-detection probability, an association matrix is formed.

Let be $L(o_i, t_j)$ the function giving the likelihood of object i with track j, P_{NT} the new track apparition probability and P_{ND} the non detection probability. Alway taking as an example the situation in the Fig. 9, the association matrix is written:

$$\begin{pmatrix} L(o_1,t_1) & -\infty & P_{NT} \\ L(o_2,t_1) & L(o_2,t_2) & P_{NT} \\ -\infty & L(o_3,t_2) & P_{NT} \\ P_{ND} & P_{ND} & -\infty \end{pmatrix}$$

Thus a possible association hypothesis corresponds to a valid assignation in the matrix of detected objects with tracks *i.e* one unique element in each row and each column is chosen to compose the assignation. In order to reduce the number of hypothesis, only the m-best association hypotheses are considered as done in Cox work [24] using this matrix. This m-best implementation of the Reid's algorithm permits to reduce the number of hypotheses and thus to control the trees' growth in width. So for each cluster (each set of tracks sharing at least one detected objects) the m-best assignment in the association matrix are computed using the Murty method [25] which computes the m-Best assignations in the matrix and by this way be obtain the m-Best Hypotheses.

4.4 Track management

In this third stage, using the m-Best Hypotheses resulting of the association stage, the set of track trees, is maintained *i.e* tracks are confirmed, deleted or created.

The track management consists in only kept the branches with leafs attached to the m-best hypothesis and prune all other branches.

New tracks are created if a new track creation hypothesis appears in the m-best hypotheses. A new created track is confirmed if it is updated by detected objects after a fixed number of algorithm steps (three in our implementation). Thus spurious measurement which can be detected as objects in the first step of our method are never confirmed.

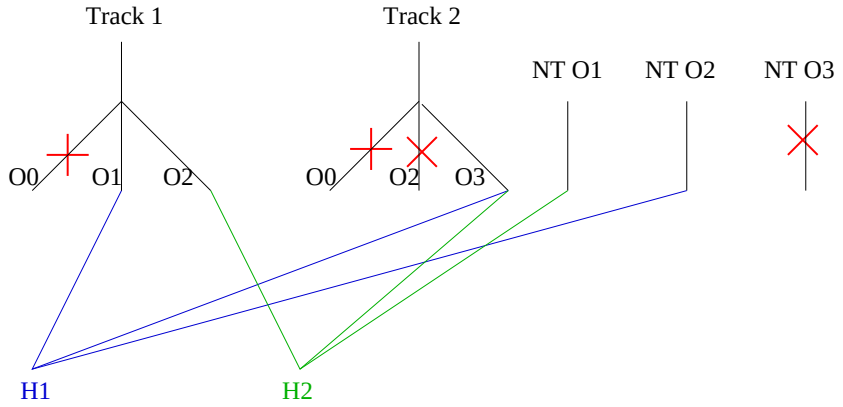

Fig. 12. Track trees pruning principle

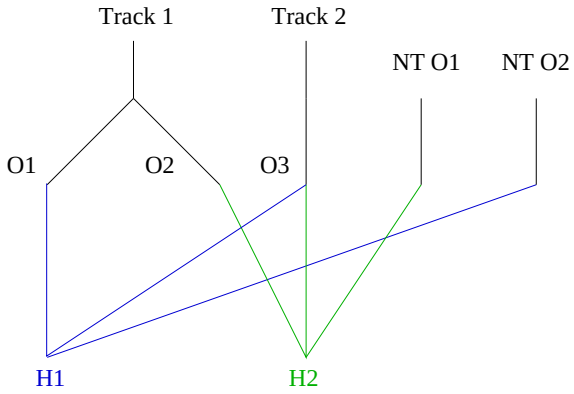

Fig. 13. Final track trees after pruning

If a non-detection hypothesis appears and so to deal with non-detection cases (which can appear for instance when an object is occulted by an other one, tracks without associated detected objects are updated according to their last associated objects and next filtering stage becomes a simple prediction. But if a track is not updated by a detected object for a given number of steps, it is deleted.

To illustrate this principle we consider the situation depicts by Fig. 9 and track trees associated showed on Fig. 10. Supposing that in the last stage we have computed the two-best hypotheses and that these hypotheses are $H_1 = T_3(T_1, O_1), T_5(T_2, O_3), NT_2$ and $H_2 = T_6(T_1, O_2), T_5(T_2, O_3), NT_1$, the set of track trees showed on Fig. 10 is modified as depict in Fig. 12. All branches with no hypothesis attached to their leafs are pruned and new tracks are created if new track creation assignement appears in hypotheses. Finally, we obtain the set of track trees depicted by Fig.13.

Furthermore, to reduce the continuously tracks' growth, an other pruning is performed. Typically trees' growth is controlled in length by the so called N-Scans pruning technique which consists in only kept the N last scans in the trees. By this way, the maximum length of tracks trees is N and it permits to apply the MHT algorithm on realistic problems.

4.5 Adaptive Filtering using Interacting Multiple Models

4.5.1 Introduction

In this filtering stage, according to previously computed predictions, estimations are performed for each leaf of all pruned track trees and new predictions are computed for the gating stage.

Regarding filtering techniques, there exists several kinds of filters, the most classical is the well known Kalman filter [10]. But in all kinds of filters, the motion model is the main part of the prediction step.

However, in the presence of uncertainties on objects' motion, defining a suitable motion model is a real difficulty. Indeed, under real world conditions, the object can have very different displacement models and it is therefore quite impossible to define an unique motion model which can match all different motions a highly maneuverable object such as pedestrian for instance could execute. Thus it is necessary to cope with motion uncertainties in such a case.

To deal with these motion uncertainties, Interacting Multiple Models (IMM) [12][13] have been successfully applied in several applications [26][27][28]. The IMM approach overcomes the difficulty due to motion uncertainty by using more than one motion model. The principle is to assume a set of models as possible candidates of the true displacement model of the object at one time. To do so, a bank of elemental filters is ran at each time, each corresponding to a specific motion model, and the final state estimation is obtained by merging the results of all elemental filters according to the distribution probabilty over the set of motion models (in the next part we note μ this probability). By this way different motion models are taken into account during filtering process.

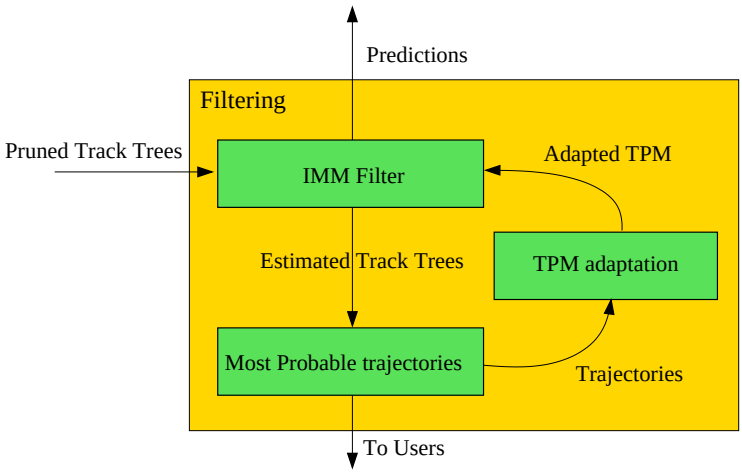

Fig. 14. Principle of our adaptive filtering program

As the quality of gating relies directly on the quality of filtering and especially the

prediction step, we have chosen Interacting Multiple Models (IMM) [12][13] to deal with motion uncertainties in this filtering part.

Besides, we developed an efficient method in which critical parameter of the IMM is on-line adapted [14][15] according to the most probable trajectories formed by tracks. Thus as Fig. 14 shows our filtering stage is composed of three parts : an IMM filtering part, a part in which most probable trajectories are computed and a last part in which we adapt the IMM filter.

4.5.2 IMM Filtering

Principle : As explained, the IMM approach overcomes the difficulty due to motion uncertainty by using a set of M elemental filters at each time, each corresponding to a specific motion model, and the final state estimation is obtained by merging the results of all filters according to the distribution probability $P(\mu)$ over the set of motion models. Also, the probability the object changes of displacement model is encoded in a transition probability matrix(TPM) which gives the distibution $P(\ \mu_t|\ \mu_{t-1})$, i.e the transition between models which is assumed Markovian.

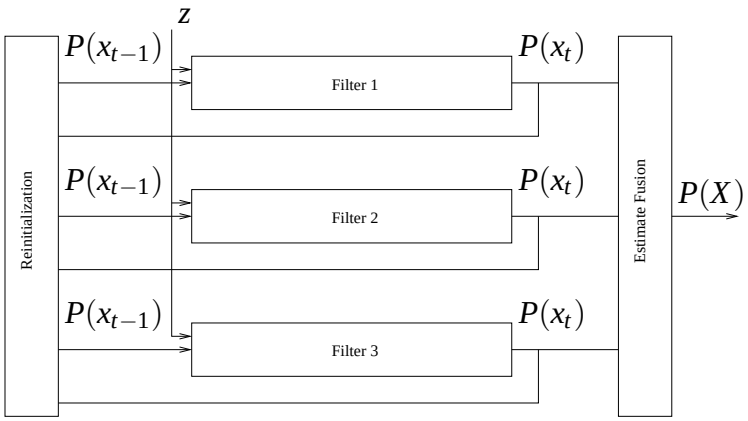

Fig. 15. Principle of IMM

One cycle of an IMM is composed of tree steps (Fig. 15): A step in which filter execution is done and μ_t is updated, a fusion step allowing to compute estimate fusion and a reinitialization step.

An unique filter give us the distribution at time t over object state x_t knowing the current detected object z_t and previous estimation:

$$\begin{aligned} P(x_t|z_t) &= \frac{1}{\alpha} \sum_{x_{t-1}} P(x_t\ x_{t-1}\ z_t) \\ &= \frac{1}{\alpha}\, P(z_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1}) P(x_{t-1}) \end{aligned} \tag{11}$$

In this equation, $P(z_t|x_t)$ is the so called sensor model distribution which give us the distribution over detected objects knowing the current state and $P(x_t|x_{t-1})$ is the motion model we need to define for each filter.

Thus as we use a bank of filters and we want to obtain an estimate fusion $P(X_t)$, knowing the current detected object and distribution over models μ and according to all filters outputs, the infered distribution is:

$$P(X_t|\ z_t\ \mu_{t-1}) = \frac{1}{\alpha} \sum_{\mu_t,\mu_{t-1},x_{t-1}^{1:M}} [\ P(\mu_{t-1})\ P(\ \mu_t|\ \mu_{t-1}) \\ P(z_t|\ \mu_t) P(X_t|x_{t-1}^{1:M}\ z_t\ \mu_t)\ ] \tag{12}$$

The first distributions $P(\mu_{t-1})$ and $P(x_{t-1}^{1:M})$ are first given *a priori* and are then computed during the process. The distribution $P(\mu_t|\mu_{t-1})$ corresponds to the TPM, it gives the transition probability between modes and so is defined as a matrix. The next distribution $P(z_t|\mu_t)$ gives the likelihood of the detected object according to the filter prediction. More precisely, for a given value of $\mu_t = i$ we obtain $P(z_t|[\mu_t = i])$ using a function (defined *a priori*) computing the likelihood between detected object and prediction. Also this distribution provides the weight of the current node, this weight allowing to obtain the probability of a given branch and thus the distribution of different hypotheses modelled by track trees. The last one $P(X_t|x_{t-1}^{1:M} z_t \mu_t)$ is obtained by the same way through filter programs : for a given value of μ_t we have $P(X_t|x_{t-1}^{1:M} z_t[\mu_t = i]) = P(x_t|x_{t-1}^{i} z_t)$. The semantic of this last distribution can be illustrated by the following case : if we know with certainty that the target is in a given mode j at time t, that is we have $P([\mu_t = j]) = 1$ and $P([\mu_t = i]) = 0\ \forall i \neq j$, thus the estimate fusion is only given by the j^{th} filter.

Also during the computation process, the new distribution probability over models $P(\mu_t)$ is computed and store in each node of track trees.

To obtain new predictions, filters are reintialized accordinf filters outputs and in each filter the corresponding dynamic model is applied. By this way, we obtain M predictions per leaf which will be use in the gating stage.

Definition of our IMM : Nevertheless, to apply IMM on real applications a number of critical parameters have to be defined, for instance the set of motion models and the transition probability matrix(TPM). To cope with this design step which can no match the reality, we propose an efficient method in which the TPM is on-line adapted [14][15].

The first step to apply our method is to define an appropriate IMM and, in particular, models which compose it.

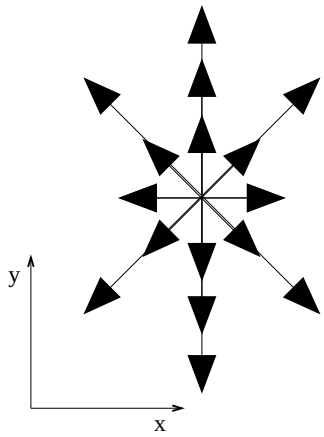

Fig. 16. The sixteen chosen motion models in the vehicle's frame

In this specific application, different objects such as cars or motorcycles can move in any directions and can often change theirs motions. Thus in our aim we choose various IMM's models to cover the set of possible directions and velocities. As each filter corresponds to a specific motion model, we have to define each motion model. So, assuming we have different possible velocities defined according to the vehicule velocity and eight directions in the set of possible directions an object can follow, we obtain sixteen motion models (Fig. 16).

Hence, according to the definition of these sixteen motion models, our IMM is composed of sixteen filters. Kalman filters are chosen for implementation as they allow fast computation.

We must usually also define the TPM. As we develop a method which computes the TPM online, we do not need specific informations concerning the TPM and no modeling are needed. So the TPM is initially chosen to be uniform. As eight modes are defined, the TPM is an uniform square 16×16 matrix. In the next part of the text, we will see how the TPM is on-line adapted.

4.5.3 Computation of the most probable trajectories

Once estimates are performed in all track trees leafs, the most probable trajectory is computed for each track. Basically, it consists in taking the branch having the maximum probability (computed during filtering) to obtain one unique hypothesis for one given track tree. This step permits to give users more readability on what is happening during tracking process and also permits us to adapt on-line the IMM parameter according to these trajectories.

4.5.4 Adaptation of the IMM

To adapt the TPM in our specific situation *i.e* tracking detected objects, most probable trajectories are considered. Taking as input the set of trajectories computed during filtering process, we will adapt one-line the TPM of the IMM filter in order to obtain a better transition between motion models close to the real behavior of tracked objects.

The principle is the following. For a given number N of trajectories we build sequences of associated models probabilities.And then, using this models probabilities, the TPM is adapted and reused in the IMM filters for the next estimations.In more details, algorithm 1, given in pseudo-code, is the algorithm defined to compute one adaptation of the TPM.

Algorithm 1 Adaptive IMM Algorithm

1: **Adaptation_of_TPM(**$T_0, ..., T_N$**)**
2: $n \leftarrow 0$
3: **repeat**
4: $S_n \leftarrow [\,]$
5: /* Store μ_k,...$\mu_{k'}$ from T_n the most probable n^{th} trajectory */
6: **for all** $Object pose\ x_k\ in\ T_n$ **do**
7: $\{\mu_k\} \leftarrow T_n(k)$
8: $S_n \leftarrow S_n \cup [\mu_k]$
9: **end for**
10: /* Compute the most probable model sequence MPS */
11: $MPS \leftarrow Viterby(S_n)$
12: /* Quantification of model transitions */
13: **for all** $Couple\ (\ MPS_k,\ MPS_{k+1})\ in\ MPS$ **do**
14: $i \leftarrow MPS_k$
15: $j \leftarrow MPS_{k+1}$
16: $F_{ij} = F_{ij} + 1$
17: **end for**
18: $n \leftarrow n+1$
19: **until** $n = N$
20: /* Update of TPM in IMM */
21: $TPM \leftarrow Normalization(F)$
22: Return *TPM* in IMM

An adaptation of the TPM is done after a given number N of trajectories obtained from tracks, to update TPM using a window on trajectories (*cf.* loop line 3-19 of algorithm 1). Moreover trajectories are processed one by one in three steps:

1- Models' probabilities are collected by travel through the computed most probable sequence
2- Most probable models' sequence is computed
3- Most probable models' transitions are quantified

Collection of models' probabilities : For each part of a given most probable trajectory computed in last stages of the filtering process, we collect the distribution over models(lines 7). Thus a model probabilities' sequence S_n obtained in such a way and is stored to be processed (line 8).

Computation of the most probable model sequence : In a next step, the most probable models' sequence of S_n is computed (line 11). More precisely, considering the actual TPM and a set $S_n = \mu_0 ... \mu_K$ of model probabilities through time 0 to K, we aim to obtain the most probable models' sequence knowing the estimates computed by the IMM:

$$Max\ P(\mu_0\ \mu_1 ... \mu_k \mid x_0\ x_1 ...\ x_K) \tag{13}$$

We just need to obtain the maximum of the distribution $P(\mu_1\ \mu_2 ... \mu_K \mid x_0\ x_1 ...\ x_K)$, thus the inference is made using the Viterbi Data Algorithm [29]. As complexity of this algorithm is in $O(KM^2)$, we efficiently obtain the most probable models' sequence.

Quantification of most probable model transitions : Using this most probable models' sequence, the number of transitions from one model to an other is quantified (lines 13 to 17). To do so a frequencies matrix is considered. This matrix models the number of transitions which have occurred from one model to an other. We note F this matrix and so F_{ij} gives the number of transitions which has occurred from model i to j. Using the most probable models' sequence corresponding to a specific trajectory and computed by the Viterbi algorithm, the update of F is directly obtained by counting transitions in this sequence. Furthermore, F is kept in memory to be used in next adaptation and before the first update all its elements are set to 1.

Finally, when N trajectories have been treated, the new TPM is obtained by normalization of the frequencies matrix F. Thus the TPM is re-estimated using all model sequences $S_1 ... S_N$ and is reused in the IMM for next executions (lines 21 and 22). In practice, before the first run, the TPM is chosen uniform (according to F initialization) as we do not want to introduce *a priori* data.

By this way an on-line adaptation of the TPM is obtained. Thus, the effectiveness of filtering part of our MHT is improved since the prediction quality is enhanced by our method. And so, the quality of the whole MHT is improved.

Example of adaptation result : Following the numeration of the different motion models defined in Fig. 17, the 16×16 frequencies matrix are shown on Fig. 18,

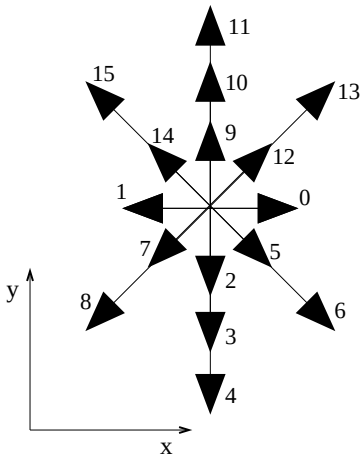

Fig. 17. Numeration of the motion models

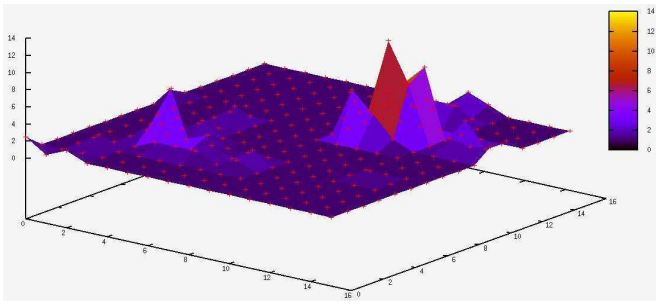

Fig. 18. Frequencies matrix obtained after five trajectories

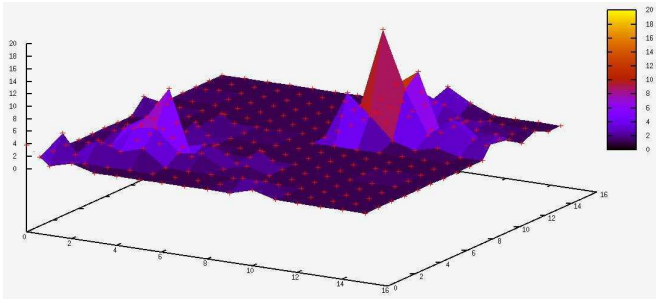

Fig. 19. Frequencies matrix obtained after twenty five trajectories

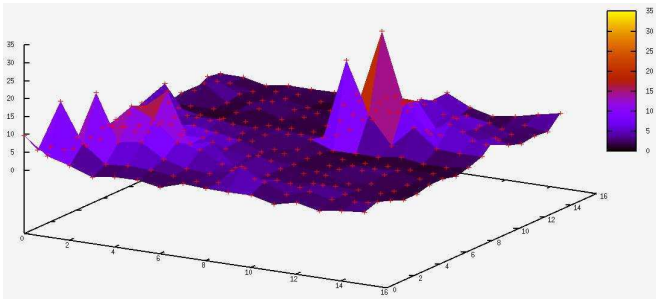

Fig. 20. Frequencies matrix obtained after fifty trajectories

Fig. 19 and Fig. 20 at three different steps of the execution process. We can see that after five trajectories some transitions appear to be more frequent than other (Fig. 18). Also, after twenty five trajectories (Fig. 19) the continuous adaptation makes appear clearly different behaviors, especially transitions between models oriented to the front and the back of the vehicle (models number from two to eight and from nine to fifteen). After a number of trajectories, an efficient model of the real objects' behaviors is obtained. Without our automatic and one-line adaptation it would be difficult to model such behaviors *a priori* and impossible to continuously model the real behavior of objects during one or several processes. Furthermore, obtain a TPM which model the real objects' motion improve the quality of the IMM filtering and thus the quality of the whole filtering process.

5 EXPERIMENTAL RESULTS

Our proposed algorithms for objects detection and tracking is tested on datasets collected with the DaimlerChrysler demonstrator car. The vehicle was driven through different kinds of scenarios such as city streets, country roads and highways with a maximum speed of 120 kph. In our implementation, the width and height of local grid map are set to 160 m and 200 m respectively, and the grid resolution is set to 20 cm. Every time the vehicle arrives at 40 m from the grid border, a new grid map is created. The object detection is run for every new laser scan and tracking process is updated according to detection set.

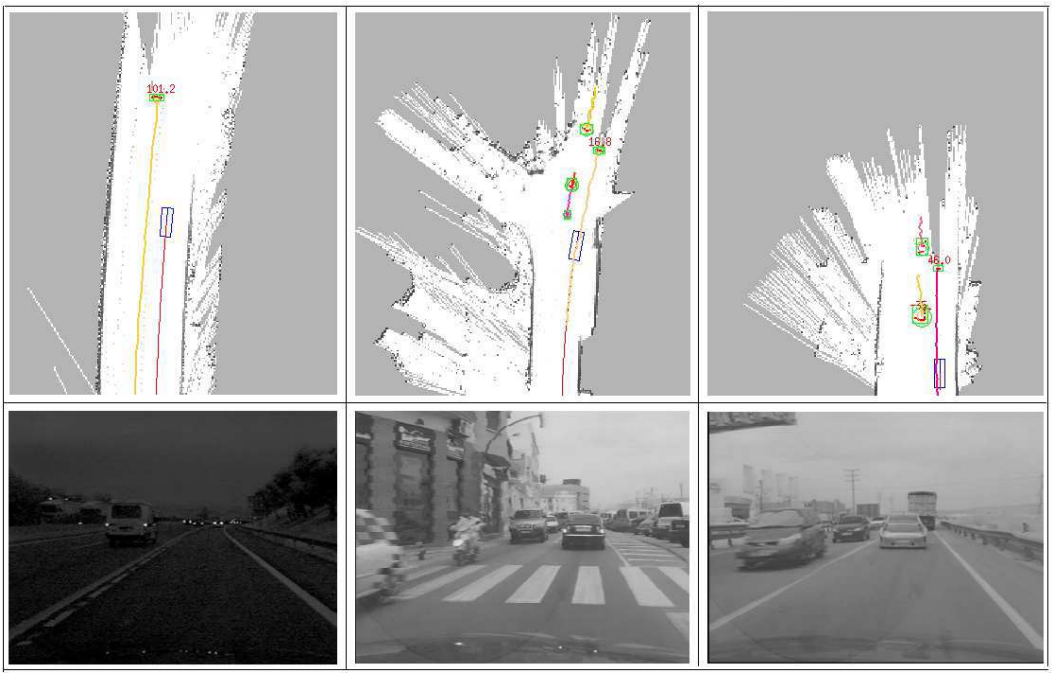

Fig. 21. Experimental results show that our algorithm can successfully perform both SLAM and DATMO in real time for different environments

The detection and tracking results are shown in Fig. 21. The images in the first row represent online maps and objects moving in the vicinity of the vehicle are detected and tracked. The current vehicle location is represented by blue box along with its trajectories after corrected from the odometry. The red points are current laser measurements that are identified as belonging to dynamic objects. Green boxes indicate detected and tracked moving objects with corresponding tracks displayed in different colors. Information on velocities is displayed next to detected objects if available. The second row are images for visual references to corresponding situations.

In Fig. 21, the leftmost column depicts a scenario where the demonstrator car is moving at a very high speed of about 100 kph while a car moving in the same direction in front of it is detected and tracked. On the rightmost is a situation where the demonstrator car is moving at 50 kph on a country road. A car moving ahead and two other cars in the opposite direction are all recognized. Note that the two cars on the left lane are only observed during a very short period of time but both are detected and tracked successfully. The third situation in the middle, the demonstrator is moving quite slowly at about 20 kph in a crowded city street. Our system is able to detect and track both the other vehicles and the motorbike surrounding. In all three cases, precise trajectories of the demonstrator are achieved and local maps around the vehicle are constructed consistently. In our implementation, the computational time required to perform both SLAM and DATMO for each scan is about $20 - 30$ ms on a 1.86GHz, 1Gb RAM laptop running Linux. This confirms that our algorithm is absolutely able to run synchronously data cycle in real time. More results and videos can be found at `http://emotion.inrialpes.fr/~tdvu/videos/`.

Quantitative results

Table 1
Quantitative results

Data Type	Real Objects	Non-detections	False Alarms	Total Tracks
City	57	393	150	88
Road	74	560	147	109
Highway	5	166	34	47

Table 1 shows quantitative results obtained using our method on three sequences of different types of environments. The first column gives the type of environment. The second is the number of real objects which entered the vehicle's sensors range which is manually counted. The third number corresponds to the number of steps in our algorithm in which one object is not detected but always tracked (non-detection cases). The fourth is the number of false alarms *i.e* when our detector (in some cases because of vehicle sensors noise) detected moving objects but our tracking

part recognized these detection has false alarms. The last one is the total number of tracks computed during the given sequence.

This results shows that in the three sequences, the most part of objects are tracked. We can note that the number of tracks remains more important than the number of real objects. It is due to objects which moves across or close to the sensors' range boundary. Indeed, close to the sensors' range boundary, laser sensor loose precision and so the detection stage became less efficient. Then if an object reappears in the sensor range it is so considered as a new one by our tracker. Also, even if an important number of non-detections and false alarms appears, the tracking part permits to cope with such problems especially since the quality of prediction step if greatly improved by our adaptive IMM. Our two stage method permits to cope with sensors noise since an efficient detection is reinforced by a robust tracking of objects.

6 CONCLUSIONS AND FUTURE WORKS

We have presented an approach to accomplish online mapping and moving object tracking simultaneously. Experimental results have shown that our system can successfully perform a real time mapping and moving object tracking from a vehicle at high speeds in different dynamic outdoor scenarios. This is done based on a fast scan matching algorithm that allows estimating precise vehicle locations and building a consistent map surrounding of the vehicle. After a consistent local vehicle map is build, moving objects are detected and are tracked using an adaptive Interating Multiple Models filter coupled with an Multiple Hypothesis tracker.

Future works include incorporating road models and object models that give a more meaningful representation of detected objects with specific shapes and sizes instead of only sets of contour points as in our current work. Also, predictions computed in the tracking part can be introduced as motion models in the detection part to increase robustness of the system.

7 ACKNOWLEDGMENTS

The work is supported by the European project PReVENT-ProFusion, and partially by the Délégation Générale pour L'Armement (DGA).

References

[1] E. Prassler, J. Scholz, P. Fiorini, Navigating a robotic wheelchair in a railway station during rush hour, Int. Journal on Robotics Research 18 (7) (1999) 760–772.

[2] D. Hähnel, D. Schulz, W. Burgard, Mobile robot mapping in populated environments, Advanced Robotics 17 (7) (2003) 579–598.

[3] C.-C. Wang, Simultaneous localization, mapping and moving object tracking, Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA (April 2004).

[4] D. F. Wolf, G. S. Sukhatme, Mobile robot simultaneous localization and mapping in dynamic environments, Autonomous Robots 19.

[5] O. Aycard, A. Spalanzani, M. Yguel, J. Burlet, T. Fraichard, C. Laugier, D. Raulo, Puvame - new french approach for vulnerable road users safety, in: Proc. of the IEEE Intelligent Vehicle Symp., Tokyo (JP), 2006.

[6] A. Elfes, Occupancy grids: a probabilistic framework for robot percpetion and navigation, Ph.D. thesis, Carnegie Mellon University (1989).

[7] J. Leonard, H. Durrant-Whyte, Simultaneous map building and localization for an autonomous mobile robot, Vol. 3, 1991.

[8] Y. Bar-Shalom, T. Fortman, Tracking and Data Association, Academic Press, 1988.

[9] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics (Intelligent Robotics and Autonomous Agents), The MIT Press, 2005.

[10] R. Kalman, A new approach to linear filtering and prediction problems, Journal of basic Engineering 35.

[11] S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filter for online nonlinear/non-gaussian bayesian tracking, IEEE Transactions on Signal Processing 50 (2).

[12] E. Mazor, A. Averbuch, Y. Bar-Shalom, J. Dayan, Interacting multiple model methods in target tracking: a survey, Aerospace and Electronic Systems, IEEE Transactions on 34 (1) (1998) 103–123.

[13] X. Rong Li, V. P.Jilkov, A survey of maneuvering target tracking-part v: Multiple-model methods, IEEE Transactions on Aerospace and Electronic Systems.

[14] J. Burlet, O. Aycard, A. Spalanzani, C. Laugier, Pedestrian tracking in car parks: an adaptive interacting multiple model based filtering method, in: Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems, 2006.

[15] J. Burlet, O. Aycard, A. Spalanzani, C. Laugier, Adaptive interactive multiple models applied on pedestrian tracking in car parks, in: Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems, Beijing (CN), 2006.

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

[17] S. S. Blackman, Multiple hypothesis tracking for multiple target tracking, Aerospace and Electronic Systems Magazine, IEEE 19 (1) (2004) 5–18.

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

[19] S. Rusinkiewicz, M. Levoy, Efficient variants of the icp algorithm.

[20] T. D. Vu, O. Aycard, N. Appenrodt, Online localization and mapping with moving object tracking, in: Proceedings of the IEEE Intelligent Vehicle Symposium, 2007.

[21] D. Fox, W. Burgard, S. Thrun, Markov localization for mobile robots in dynamic environments, Journal of Artificial Intelligence Research 11.

[22] S. Thrun, W. Burgard, D. Fox, A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping, in: ICRA, 2000.

[23] D. B. Reid, An algorithm for tracking multiple targets, IEEE Transactions on Automatic Control 24 (6).

[24] I. J. Cox, S. L. Hingorani, An efficient implementation and evaluation of reid's multiple hypothesis tracking algorithm for visual tracking, Pattern Recognition, 1994. Vol. 1 1.

[25] K. G. Murty, An algorithm for ranking all the assignments in order of increasing costs, Operations Research 16 (1968) 682–687.

[26] S. Blackman, Multiple Target Tracking with Radar Applications, Artech House: Dedham, 1986.

[27] M. Busch, S. Blackman, Evaluation of imm filtering for an air defence system application, in: Proceedings SPIE Signal Data Process. Small targets, Vol. SPIE 2561, 1995, pp. 435–447.

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

[29] G. D. Forney, The viterbi algorithm, Proceedings of The IEEE 61 (3) (1973) 268–278.