Grid-based localization and local mapping with moving object detection and tracking

Trung-Dung Vu, Julien Burlet, Olivier Aycard

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

Trung-Dung Vu, Julien Burlet, Olivier Aycard

Grenoble I University & INRIA Rhônes Alpes, France
{trung-dung.vu,olivier.aycard}@inrialpes.fr,
julien.burlet@gmail.com

Abstract

We present a real-time algorithm for simultaneous localization and local mapping (local SLAM) with detection and tracking of moving objects (DATMO) in dynamic outdoor environments from a moving vehicle equipped with a laser scanner, short range radars and odometry. To correct the vehicle odometry we introduce a new fast implementation of incremental scan matching method that can work reliably in dynamic outdoor environments. After obtaining a good vehicle localization, the map surrounding 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 Model (IMM) 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 critical situations to warn the driver in advance and this will certainly improve driving safety and prevent traffic accidents.

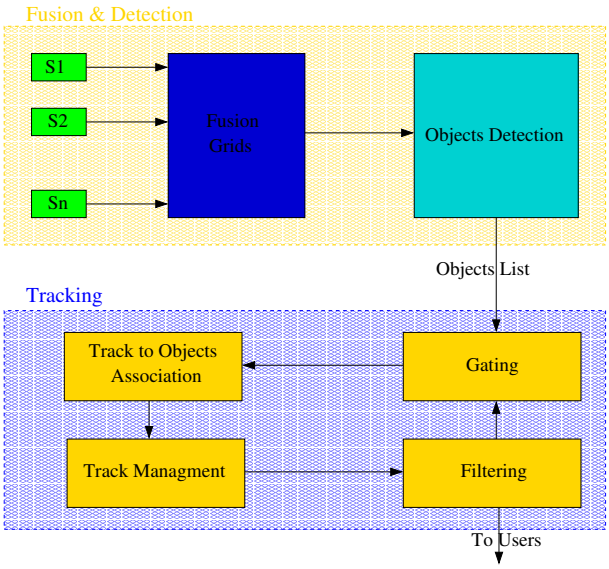

Fig. 1. Architecture of the perception system

In the literature, SLAM and DATMO have been attracted considerable research works [1] [2] [3] and they also are essential parts of the perception modules in driverless cars [4] [5] winning the recent series of DARPA Grand Challenge competitions [1] . However, for highly dynamic outdoor scenarios like in 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 reliably over time.

In this context, we designed and developed a generic perception architecture addressing these problems focusing on outdoor dynamic environments [6]. The architecture (Fig. 1) is comprised of two main parts: the first part where the map of vehicle environment is constructed and dynamic objects are identified; the second part where detected moving objects are verified and tracked.

In the first part of the architecture, to model the environment surrounding the vehicle, we employ the occupancy grid framework proposed by Elfes [7]. In order to perform mapping or modeling the environment from a moving vehicle, generally a precise vehicle localization is necessary. To correct vehicle locations 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. And when new laser measurements are coming, dynamic objects can be then detected based on their discrepancies with the constructed grid map. Related results have been presented in our previous publication [8] and in this paper we employ the radar data combined with object detection

[1] www.darpa.mil/grandchallenge/index.asp

results from laser data in order to obtain a more robust performance.

In the second part, previously detected moving objects in the vehicle environment are passed to the tracking process. Since some objects may be occluded or not detected, some are false alarms, tracking helps to identify occluded objects, recognize false alarms and reduce missed detections. In general, the multiple object tracking problem is complex: it involves the definition of filtering methods as well as the data association methods and maintenance of the list of objects currently present in the environment [9]. Regarding the filtering techniques, Kalman filters [10] and particle filters [11] are mostly used. These filters require the definition of a specific dynamic model of tracked objects. However, defining a suitable motion model is not trivial and Interacting Multiple Models (IMM) [12] have been successfully applied in practice. In our previous works [13], we have developed a fast method to adapt on-line IMM according to trajectories of detected objects and so that we obtain a suitable and robust tracker. To deal with the data association and track maintenance problem, we extend our approach to multiple objects tracking using the Multiple Hypothesis Tracking (MHT) approach [14][15].

1.1 Experimental platform

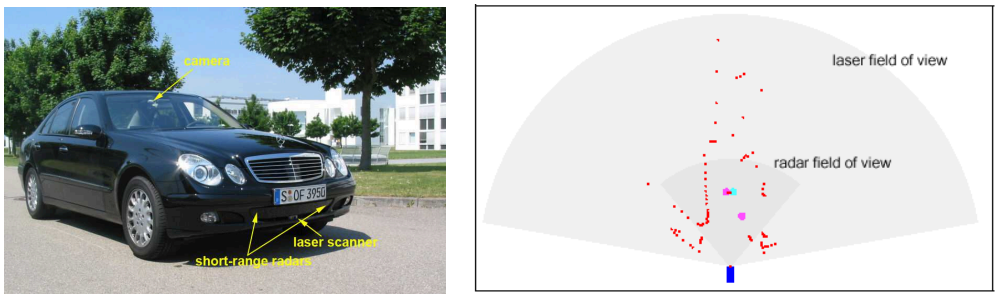

Fig. 2. Left: the Daimler demonstrator car. Right: an example of sensor data, laser measurements are displayed in small red dots and radar measurements displayed as bigger dots.

The proposed algorithm for solving SLAM and DATMO is tested on data collected from the Daimler demonstrator car equipped with a camera, two short range radars and a laser scanner (Fig. 2). The laser scanner can detect obstacles at a range of 70m under a field of view of 160°. It provides raw data as a list of impacts with an angular resolution of 1°. The radars detect targets up to 30m within a field of view of 80° and return pre-filtered data as a list of "dot" objects with their estimated positions and Doppler velocities (Fig. 2 right). In addition, vehicle odometry information such as velocity and yaw rate are provided. The measurement cycle time of the sensor system is 40ms.

In our implementation, laser data is used to perform mapping as well as detection and tracking of moving objects. Radar data is then fused with laser data to confirm the results obtained by laser data in order to give a more reliable results on detection

and tracking objects in the radar field of view. Images from camera are only for visualization purpose.

2 RELATED WORK

Before discussing in detail our approach to problems of SLAM and DATMO, it is interesting to recall some notable works in the domain.

One of the first works on SLAM with DATMO was that of Prassler's group [1]. They described a first system on automated wheelchairs for static and dynamic object detection, moving object tracking and obstacle avoidance. The environment is represented by a time stamp grid map that provide a interesting way to detect and track moving objects. However, this method rely completely on odometry information with suppose that the odometry is ideal and it cannot detect objects moving slowly. Although the proposed solution is not really complete, it identified the need of both SLAM and DATMO for automated mobile systems.

Haehnel *et al.* in [2] used a feature-based approach to identify pedestrians from laser range scans and use Joint Probabilistic Data Association particles filters [16] to track moving pedestrians indoor. The corresponding measurements are then filtered out and classical scan registration and mapping techniques in static environment are used. However, this approach is not able to work in outdoor environment where various dynamic objects can not be described by simple features.

Wang [3] developed the first outdoor real-time system solving both SLAM and DATMO simultaneously for urban environments from a ground vehicle. To correct the vehicle odometry he used an ICP-based matching scan method and moving objects are detected based on a simple geometric analysis. He also presented a mathematical framework integrating both SLAM and DATMO and showed that they can be mutually beneficial from each other. The idea is that the results of SLAM will be more accurate if moving objects can be filtered out and thanks to a more accurate pose estimation and a better map from SLAM, DATMO can detect and track moving objects more reliably.

Recently, in the DARPA Urban Challenge competition, we have been witnessed significant advances in efforts of building autonomous vehicles. It is shown that driverless cars, for instance: Boss [5] and Junior [4], are capable of operating autonomously and safely through urban-alike environments. However, testing scenarios for the competition contains only vehicles as moving objects which limits their approaches to be only able to detect and track vehicles. In addition, to obtain a good performance, participant vehicles are equipped with so many precise and expensive sensors, such as 3D laser scanners, 2D laser scanners, radars, vision, precise inertial sensors...

Inspired by the pioneer work of Wang on SLAM and DATMO, our objective here is trying to put forward the state-of-the-art solutions to these tasks in order to build a general and reliable vehicle perception system with affordable sensors (e.g. 2D laser scanner, short-range radars). To this end, we introduce a new fast grid-based laser scan matching method to correct vehicle odometry that works extremely well even in the presence of dynamic entities. It will be shown later that this is an important step to build an accurate map of the environment and help to detect moving objects reliably. We also present a new approach of multiple object tracking capable of online adapting movements of moving objects which results in a more robust tracker. Parts of this work have been published separately in [8] [13].

In the following section, we describe in detail our approach to vehicle localization and environment mapping. Algorithm for detecting moving objects is presented in Section 4. Multi objects tracking approach is detailed in Section 5. Experimental results are reported in Section 6 and finally conclusions and future works are given in Section 7.

3 LOCAL SLAM

To model the environment surrounding of vehicle, we employ the occupancy grid framework proposed by Elfes [7]. Compared with feature-based approaches [17], grid maps can represent any environment and are especially 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.

To perform mapping, only laser data is used. For our purpose of safety vehicle navigation, 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.

3.1 Notation

Before describing our approach in detail, we introduce some notations used. 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.

3.2 Occupancy Grid Map

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 $Odds(x) = \frac{P(x)}{P(\bar{x})} = \frac{P(x)}{1-P(x)}$, 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:

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

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.

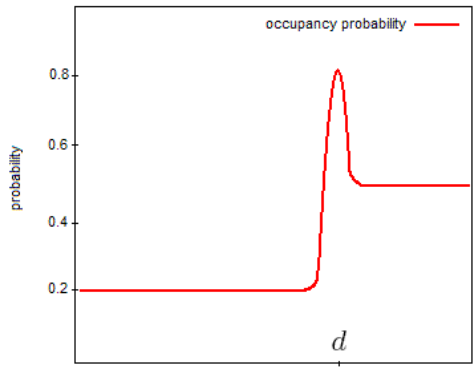

Fig. 3. Profile of an inverse sensor model illustrates the occupancy probability along a laser beam measuring a distance of d.

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 6 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*.

3.3 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.

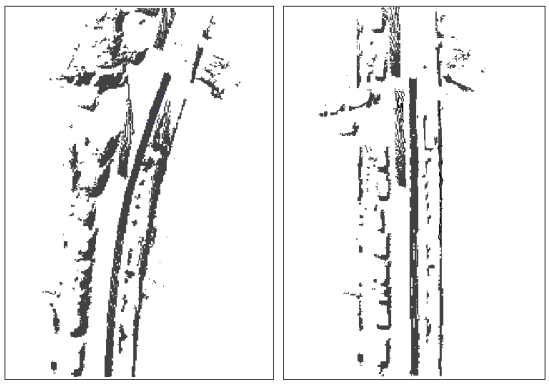

Fig. 4. Hit maps build directly from raw laser data collected from a vehicle moving along a straight street: with vehicle localization using odometry (left); and using results of scan matching (right). Note that the scan matching results are not affected by moving objects in the street.

An alternative approach that can overcome these limitations consists in setting up the matching problem as a maximum likelihood estimation. In this approach, given an underlying vehicle dynamics constraint, the current scan 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 speed. 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):

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

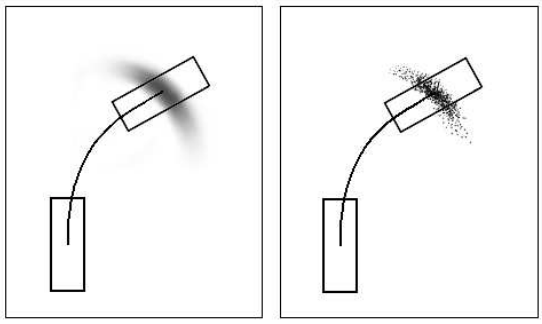

Fig. 5. The probabilistic velocity motion model $P(x_t \,|\, x_{t-1}, u_t)$ of the vehicle (left) and its sampling version (right).

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 [20]. 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 a vehicle being at location x_t given its 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.

For the measurement model $P(z_t \,|\, x_t, M_{t-1})$, mixture beam-based model is widely used in the literature [21][22]. 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:

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

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.

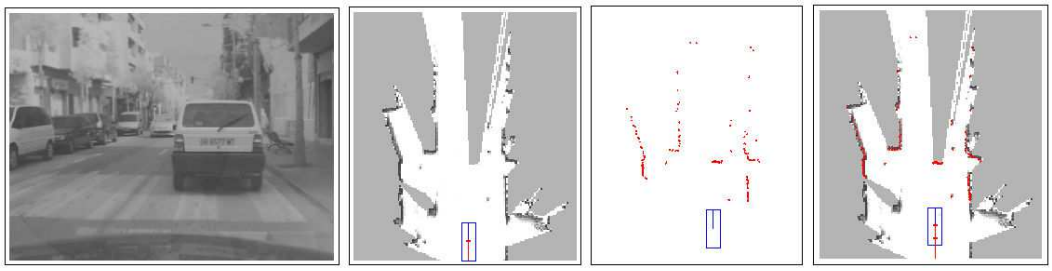

Fig. 6. An example of scan matching. From left to right: reference image; map constructed so far M_{t-1} with previous vehicle location x_{t-1}; new laser measurement z_t; and matching result is obtained by trading off the consistency of the measurement with the map and the previous vehicle pose.

It remains to describe how we maximize (7) to find the correct pose $\hat{x}_t$. Hill climbing strategy in [23][22] 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 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). 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.

3.4 Local mapping

Because we do not need to build a global map nor deal with loop closing problem, only one online map is maintained 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.

4 MOVING OBJECTS DETECTION

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

4.1 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 D^i store a value α^i indicating the number of observations that a moving object has been passed by that cell. The bigger value of α^i, the more probability that object seen at D^i is moving.

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,

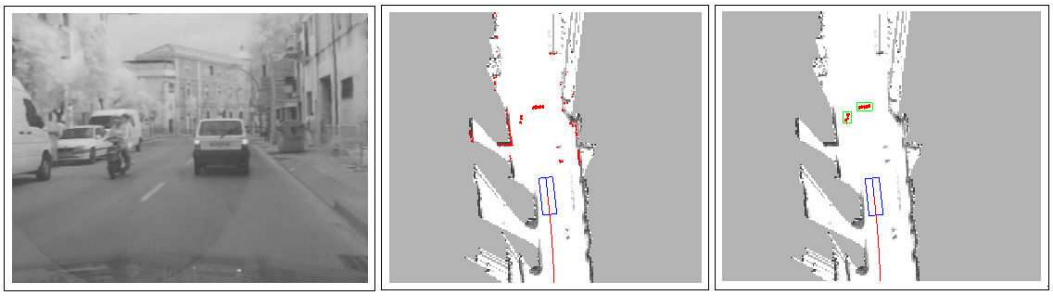

Fig. 7. Moving object detection example. See text for more details.

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.

The second step when 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 a threshold of 0.2 m that is chosen empirically .

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.

4.2 Fusion with radar

The general purpose of the data fusion is to provide a more reliable and more accurate model than a single data would provide. After moving objects are identified from laser data, we confirm the object detection results by fusioning with radar data and estimate velocities of the detected objects.

With the radar sensors being used, a built-in 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 moving objects. The object lists of the two radars are independent from each other. Each object is provided with information about the location and the 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. Since the radar is setup with the field of view much smaller than the laser field of view (Fig. 2), the fusion process indeed did not help much to improve the overall detection results but we can see how detection results could be improved if more sensors available.

Fig. 8. Moving objects detected from laser data are confirmed by radar data.

5 MULTIPLE HYPOTHESIS TRACKING USING ADAPTIVE IMM

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 tracks according to detected (observed) objects [2] obtained at each step. For convenience we call *track* a tracked object that is composed by a list of the same object detected over time. This involves a choice of filtering methods, but also

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

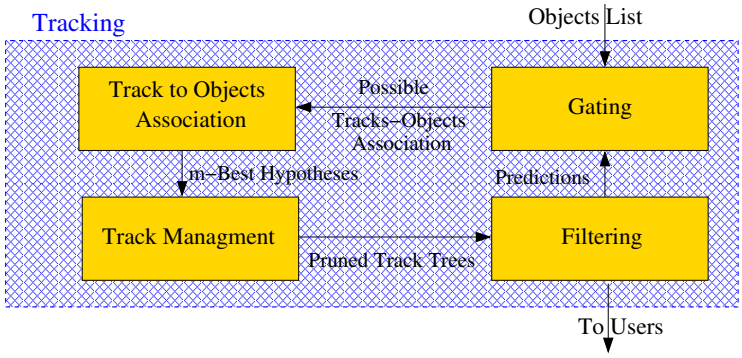

Fig. 9. Architecture of multi-object tracking system

data association methods and a maintenance of the list of objects currently present in the environment. The most known techniques are the the Global Nearest Neighborhood (GNN) combined with filtering, Joint Probabilistic Data Association Filter (JPDAF) and the Multiple Hypothesis Tracking (MHT) [9][20]. 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 one track by a 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 association hypotheses are built over time. In conflict situations, instead of taking an immediate decision (GNN) or combining hypotheses (JPDAF), hypotheses are propagated into the future in anticipation that it will resolve the possible association ambiguities.

The basic principle of MHT is to generate and update a set of association hypotheses during tracking 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. 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 [24]. It permits to systematically generate and evaluate hypotheses by building track trees. For these reasons, we based our MHT on this efficient algorithm.

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 (ie, a motion model). However, defining a suitable motion model is a real difficulty. To deal with this problem, Interacting Multiple Models [12][25] have been successfully applied in several applications.

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 probability over the set of motion models. By this way different motion models are taken into account during filtering process. In previous works [26] [13], we have developed a fast method to adapt on-line IMM according to trajectories of detected pedestrian and so we obtain a suitable and robust tracker. In this work we extent this method in order to track dynamic objects in the vehicle environment.

As shown in Fig. 9, our multi-object tracking method is composed of four different parts:

- 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 the 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).

5.1 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. 10, 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.

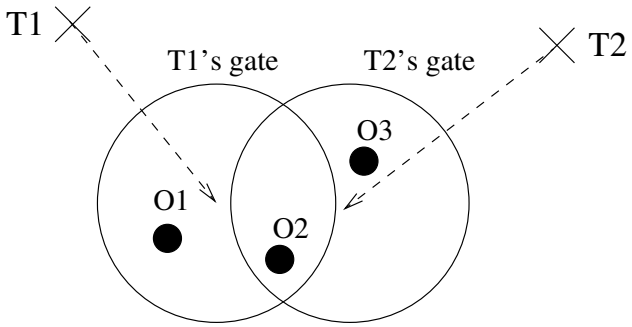

Fig. 10. Example of association problem

5.2 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. 10, 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 [27] 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 [28] which computes the m-best assignations in the matrix and by this way be obtain the m-best Hypotheses.

5.3 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.

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.

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.

5.4 Adaptive Filtering using Interacting Multiple Models

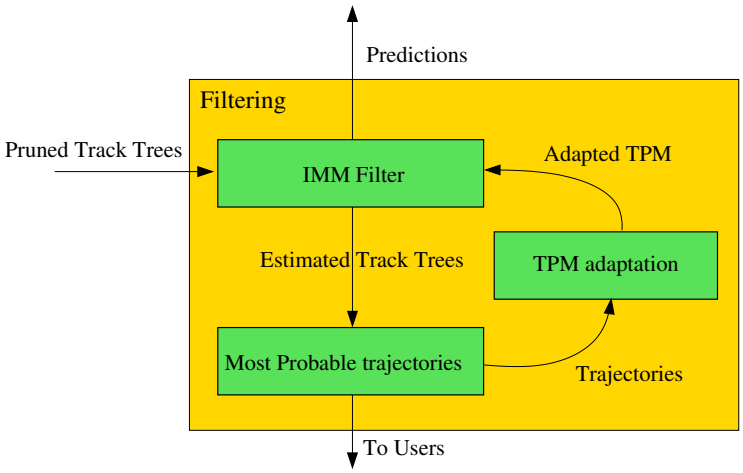

Fig. 11. Principle of our adaptive filtering program

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][25] 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 [26][13] according to the most probable trajectories formed by tracks. Thus as Fig. 11 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.

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 distribution $P(\ \mu_t|\ \mu_{t-1})$, i.e the transition between models which is assumed Markovian.

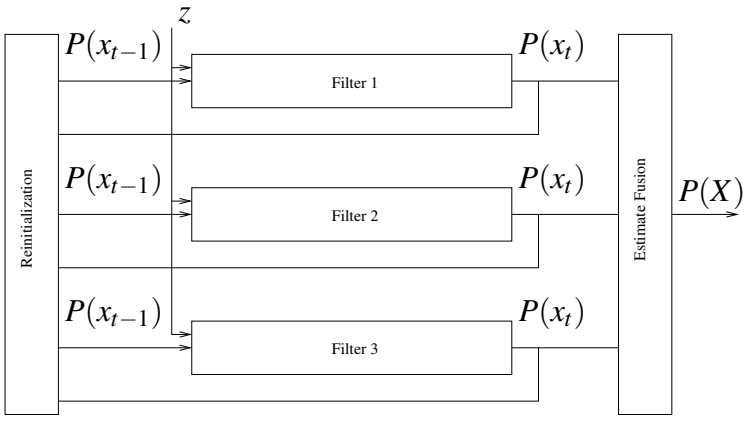

Fig. 12. Principle of IMM

One cycle of an IMM is composed of tree steps (Fig. 12): A step in which filter execution is done and $P(\mu_t)$ 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 $P(x_t|z_t)$. Also, $P(\mu_t)$ is updated according to $P(\mu_t|\mu_{t-1})$ corresponding to the TPM, it gives the transition probability between modes and so is defined as a matrix, $P(\mu_{t-1})$ is the previous distribution over models and the likelihood of the current detected object with each filter.

Thus as we use a bank of filters and we want to obtain an estimate fusion $P(X_t)$, according to all filters outputs. The estimate fusion is obtained by:

$$P(X_t) = \sum_{m=1}^{M} P(\ [\mu_t = m])\ P_m(x_t|z_t) \tag{10}$$

Also during the computation process, the new distribution probability over models

$P(\mu_t)$ is computed and store for each hypothesis.

To obtain new predictions, filters are reinitialized according 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:

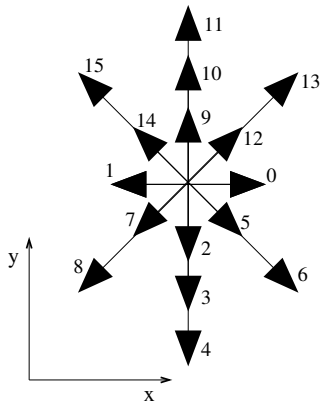

Fig. 13. The sixteen chosen motion models in the vehicle's frame

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 cannot match the reality, we propose an efficient method in which the TPM is adapted online.

The first step to apply our method is to define an appropriate IMM and, in particular, models which compose it. In specific applications, 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. In previous work for one pedestrian tracking we focused only on directions but here we focus on a range of velocities while keeping a set of directions to cope with directions' transition in vehicles' behavior. 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 vehicle velocity and eight directions in the set of possible directions an object can follow, we obtain sixteen motion models (Fig. 13).

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.

5.4.1 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.

5.4.2 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.

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:

$$Max\ P(\mu_0\ \mu_1 ... \mu_k \mid x_0\ x_1 ...\ x_K) \tag{11}$$

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

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:

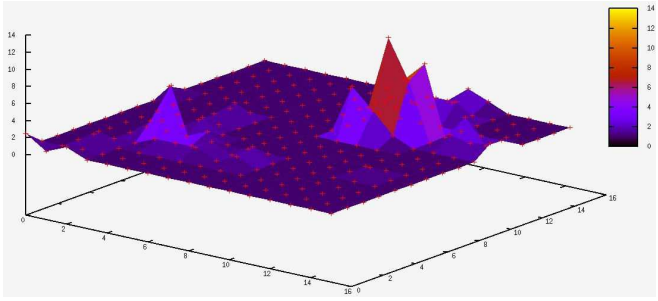

Fig. 14. Frequencies matrix obtained after five trajectories

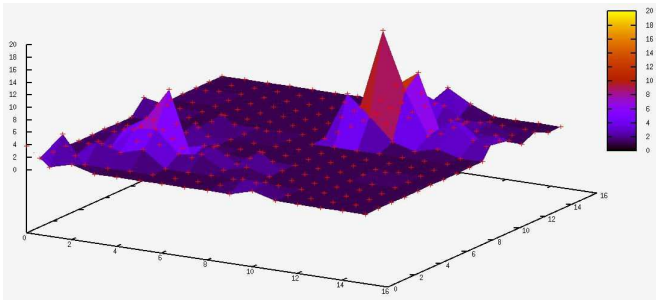

Fig. 15. Frequencies matrix obtained after twenty five trajectories

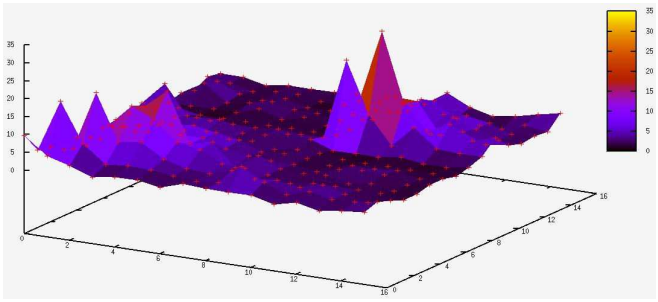

Fig. 16. Frequencies matrix obtained after fifty trajectories

Following the numeration of the different motion models defined in Fig. 13, the 16×16 frequencies matrix are shown on Fig. 14, Fig. 15 and Fig. 16 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. 14). Also, after twenty five trajectories (Fig. 15) 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) [3] . After a

[3] According to nonholonomic constraints we cannot obtain direct transitions from the front model to the back model for instance but as shown in the results transitions between adjacent models occur

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.

6 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 the tracking process is updated according to the detection results.

Fig. 17 shows some snapshots of the moving object detection and tracking process in action. 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 purple 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 estimated velocities is displayed next to detected objects. The second row are images for visual references to corresponding situations.

In the figure, 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 with data cycle in real time. More

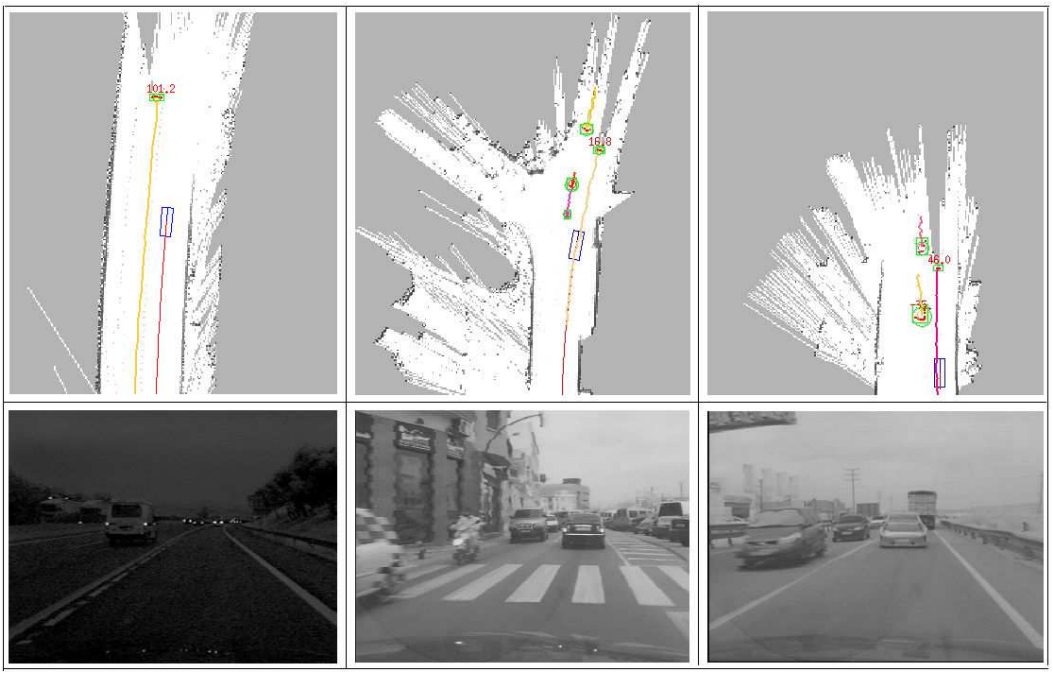

Fig. 17. Experimental results show that our algorithm can successfully perform both SLAM and DATMO in real time for different environments

results and videos can be found at http://emotion.inrialpes.fr/ tdvu/videos/.

Quantitative results

Data Type	Real Objects	Non-detections	False Alarms	Total Tracks
City	57	7%	3%	88
Road	74	11%	3%	109
Highway	5	7%	1.5%	47

The table above shows quantitative results obtained using our method on three sequences of different types of environments. The first column are different types of scenario. The second column shows the numbers of real objects which entered the vehicle's sensors range which is manually counted. The third number corresponds to the numbers of steps in our algorithm in which one object is not detected but always tracked (non-detection cases). The fourth column are the numbers of false alarms *i.e* when our detector (in some cases because of vehicle sensors noise) detected moving objects but our tracker recognized these detection. In the last column are the total numbers of tracks computed during the given sequence.

The results show that during three sequences, most part of object appearances 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 become 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 is greatly improved by our adaptive IMM. Our two stage approach permits to cope with sensors noise where an efficient detection is reinforced by a robust tracking of objects.

7 CONCLUSIONS AND FUTURE WORKS

We have presented an approach to accomplish local mapping with detection and tracking moving objects. 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 reliably using an adaptive Interacting Multiple Models filter coupled with an Multiple Hypothesis tracker.

Future works include incorporating object models of several predefined classes with specific shapes and sizes that give a more meaningful representation of detected objects instead of only sets of contour points as in our current work. In addition, algorithms of road detection and road type classification based on constructed grid map are being considered. The motivation is that road detection can help object detection step to filtering out noisy and irrelevant data off-the-road and focus more on road-likely regions. In all, the fusion of a vehicle map, road detection, moving object classification and tracking modules certainly will enable a better interpretation of driving situations.

8 ACKNOWLEDGMENTS

The work is supported by the European project PReVENT-ProFusion [4] and partially by the French Délégation Générale pour L'Armement (DGA).

[4] http://www.prevent-ip.org/profusion

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, R. Triebel, W. Burgard, S. Thrun, Map building with mobile robots in dynamic environments, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2003.

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

[4] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, S. Thrun, Junior: The Stanford entry in the Urban Challenge, Journal of Field Robotics.

[5] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y. Seo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, D. Ferguson, Autonomous driving in urban environments: Boss and the Urban Challenge, Journal of Field Robotics 25(8) (2008) 425–466.

[6] O. Aycard, A. Spalanzani, M. Yguel, J. Burlet, N. D. Lac, A. D. L. Fortelle, T. Fraichard, H. Ghorayeg, M. Kais, C. Laugier, C. Laurgeau, G. Michel, D. Raulo, B. Steux, PUVAME - new French approach for vulnerable road users safety, in: Proceedings of the IEEE Intelligent Vehicles Symposium 2006, Tokyo, Japan, 2006.

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

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

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

[10] R. E. Kalman, A new approach to linear filtering and prediction problems, Transactions of the ASME–Journal of Basic Engineering 82 (Series D) (1960) 35–45.

[11] M. S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for online non-linear/non-Gaussian Bayesian tracking, IEEE Transactions on Signal Processing 50 (2) (2002) 174–188.

[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] 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.

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

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

[16] D. Schulz, W. Burgard, D. Fox, A. Cremers, Tracking multiple moving targets with a mobile robot using particle filters and statistical data association, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2001.

[17] J. Leonard, H. Durrant-Whyte, Simultaneous map building and localization for an autonomous mobile robot, Vol. 3, Osaka, Japan, 1991, pp. 1442–1447.

[18] P. J. Besl, H. D. Mckay, A method for registration of 3D shapes, Pattern Analysis and Machine Intelligence, IEEE Transactions on 14 (2) (1992) 239–256.

[19] S. Rusinkiewicz, M. Levoy, Efficient variants of the ICP algorithm, in: Proceedings of the 3rd Int. Conf. on 3D Digital Imaging and Modeling, 2001, pp. 145–152.

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

[21] D. Fox, W. Burgard, S. Thrun, Markov localization for mobile robots in dynamic environments, Journal of Artificial Intelligence Research 11 (1999) 391–427.

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

[23] S. Thrun, W. Burgard, D. Fox, A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping, in: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2000.

[24] D. B. Reid, An algorithm for tracking multiple targets, IEEE Transactions on Automatic Control 24 (1979) 843–854.

[25] X. Rong Li, V. P. Jilkov, A survey of maneuvering target tracking - Part V: Multiple-model methods, IEEE Transactions on Aerospace and Electronic Systems 41(4) (2005) 12551321.

[26] 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.

[27] I. J. Cox, S. L. Hingorani, An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (2) (1996) 138–150.

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

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