Interacting Multiple Models based Classification of Moving Objects

Julien Burlet
University of Grenoble1 & INRIA Rhône-Alpes
Grenoble, France
Julien.Burlet@inrialpes.fr

Olivier Aycard
University of Grenoble1 & INRIA Rhône-Alpes
Grenoble, France
Olivier.Aycard@inrialpes.fr

Qadeer Baig
University of Grenoble1 & INRIA Rhône-Alpes
Grenoble, France
Qadeer.Baig@inrialpes.fr

***Abstract*—In this paper, we present an approach performing object behavior classification embedded in a complex and efficient perception method. This method, applied in dynamic outdoor environments using a moving vehicle equipped with a laser scanner, is composed of a local simultaneous localization and mapping (SLAM) with detection and tracking of moving objects (DATMO).**

While the SLAM is performed by an implementation of incremental scan matching method, the tracking if performed by a Multiple Hypothesis Tracker (MHT) coupled with an adaptive Interacting Multiple Models Filter (IMM). The classification process takes place in the filtering stage and is based on one of the key parameters of the IMM filter which is the Transition Probability Matrix (TPM) modeling objects motion transitions. It permits to automatically classify object behavior and to reuse the classification output to enhance the prediction step in the filtering process.

The experimental results on datasets collected from a Daimler Mercedes demonstrator in the framework of the European Project PReVENT-ProFusion2 demonstrate the capacity of the proposed algorithm.

***Index Terms*—Object behavior classification, SLAM, DATMO, TPM.**

I. 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 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.

In this context, we design and develop a generic architecture to solve SLAM and DATMO in dynamic outdoor environments. This architecture has been used in the framework of the European project PReVENT-ProFusion[1] in cooperation with Daimler [1] and Volvo [2] and is currently in the european project Intersafe2[2] in cooperation with Volkswagen [3]. This architecture (Fig. I) is divided into two main parts.

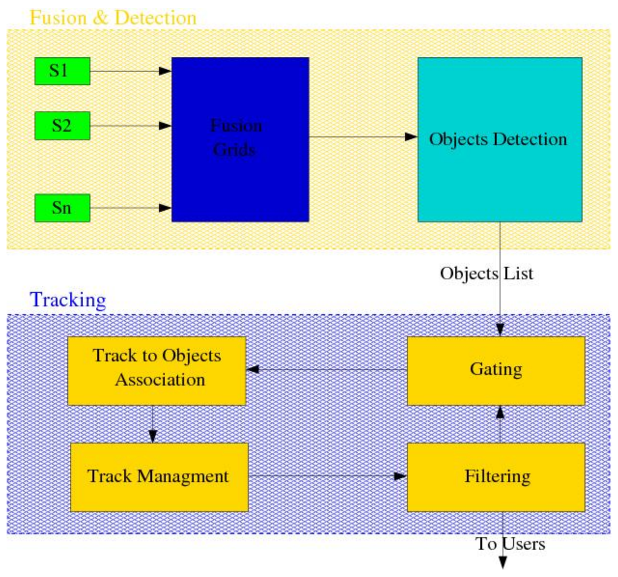

Fig. 1. Architecture of our system

The first level of our architecture is dedicated to Environment Mapping & Localization and Moving Objects Detection [4]. For the SLAM problem, we use a maximum likelihood approach to build a consistent local map using occupancy grid and to localize the ego vehicle inside the map. After a consistent local map has been constructed, moving objects can be detected using inconsistencies between observed free space and occupied space in the local grid map.

In the second part, the previously detected moving objects are verified and tracked. This part is composed of four different

[1]www.prevent-ip.org/profusion
[2]www.intersafe-2.eu

modules to solve the problem of multi-objects tracking:

- 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 objects to observations association and generate association hypothesis, each observation corresponding to a previously known moving object. Output is composed of the computed set of association hypothesis.
- In the third part called objects management, objects are confirmed, deleted or created according to the association results and a pruned set of association hypothesis is 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.

As the quality of gating relies directly on the quality of filtering and especially the prediction step, we have chosen Interacting Multiple Models (IMM) [5] to deal with motion uncertainties in this filtering part. 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 mode of the target 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. Also, the probability the target changes of displacement mode is encoded in a transition probability matrix(TPM), i.e the transition between modes which is assumed Markovian. 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). In practice, the TPM is often assumed known and is chosen a priori. Besides, we developed an efficient method in which TPM is on-line adapted according to the most probable trajectories formed by objects [6]. In this paper, we present an extension of on-line adaptation of TPM to classify behaviors of moving objects. Actually, TPM model changes in displacement of moving objects. So, these set of changes of displacement could be seen as a set of behaviors. For instance, when moving in an environment, some objects will have similar behviors (*e.g* pedestrians crossing a parking or cars moving in a specific direction. These similar behaviors will cause the same changes of displacements. To obtain a classification of behaviors of des objects, we use automatically on-line adapted TPM to characterize typical displacements of objects. Afterwards, this classification is used to specify a precise TPM for a given object and so to obtain more adapted modelization of typical displacements of this object.

The rest of the paper is organized as follows. In the next section, we present the Daimler Mercedes demonstrator. We summarize our previous work on on-line adaptation of TPM in section III. Description of our method to automatically classify tracked objects based on their behaviors is detailled in section IV. Experimental results are given in Section V and finally in Section VI conclusions and future works are discussed.

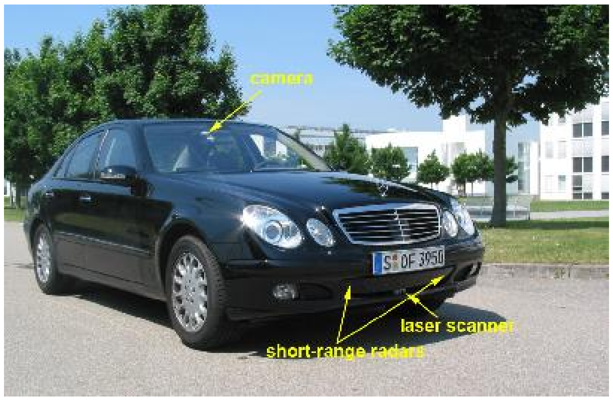

Fig. 2. The Daimler Mercedes demonstrator car.

II. THE DAIMLER MERCEDES DEMONSTRATOR

The DaimlerChrysler demonstrator car is equipped with a camera, two short range radar sensors and a laser scanner (Fig. 2). The radar sensor is with a maximum range of $30m$ and a field of view of $80°$. The maximum range of laser sensor is $80m$ with a field of view of $160°$ and a horizontal resolution of $1°$. 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 $40ms$. Images from camera are for visualization purpose.

III. PREVIOUS WORK

In our specific application, different objects such as cars or motorcycles can move in any directions and can often change their motions. Thus in our aim we choose various IMM's motion 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 vehicle velocity and eight directions in the set of possible directions an object can follow, we obtain sixteen motion models (Fig. 3). Hence, according to the definition of these sixteen motion models, our IMM is composed of sixteen kalman filters.

Besides, we developed an efficient method in which critical parameter of the IMM is on-line adapted according to the most probable trajectories formed by objects. The principle is the following. For a given number N of trajectories we build sequences of associated motion models probabilities. And then, using these motion models probabilities, the TPM is adapted and reused in the IMM filters for the next estimations. The TPM is initially chosen to be uniform. 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

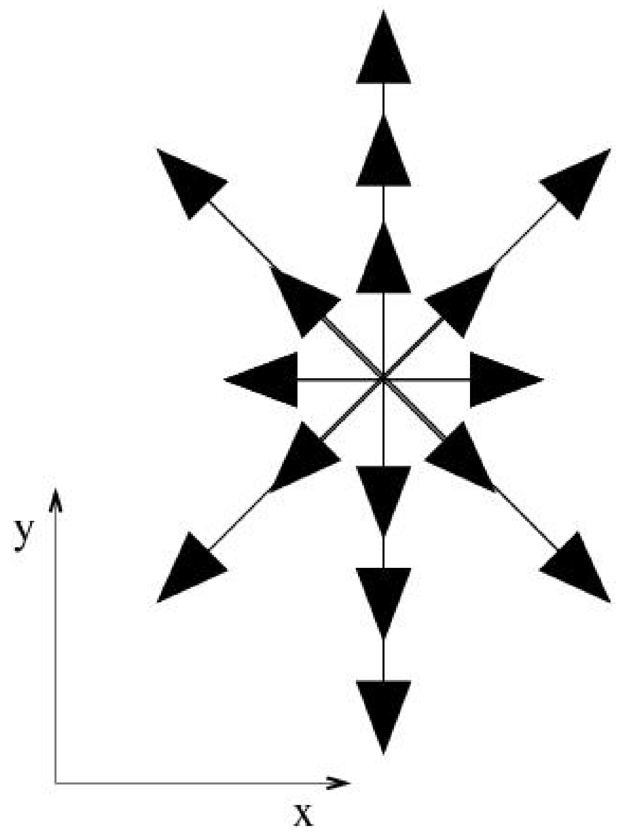

Fig. 3. The sixteen chosen motion models in the vehicle's frame

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- The most probable models' sequence is computed
3- The most probable models' transitions are quantified

A. 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 is obtained in such a way and is stored to be processed (line 8).

B. 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:

$$Max\ P(\mu_0\ \mu_1 ... \mu_k \mid x_0\ x_1 ...\ x_K) \tag{1}$$

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 [7]. As complexity of this algorithm is in $O(KM^2)$, we efficiently obtain the most probable models' sequence.

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 $\mu_k$,...$\mu_{k'}$ from $T_n$ the most probable $n^{th}$
       trajectory */
6:     **for all** $Objectpose\ x_k\ in\ T_n$ **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:    $MPS \leftarrow Viterbi(S_n)$
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
```

C. Quantification of most probable model transitions

Using this most probable models' sequence, the number of transitions from one model to another 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 another. 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.

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

IV. BEHAVIOR BASED CLASSIFICATION

In this section, we detail how we improved our adaptive method to track objects by adding a classification module. It permits to specify a model of TPM for each track. The goal is in a first stage to compute TPM classes modeling typical object behaviors. The second stage aim to identify which behavior and thus at which TPM class it belong. This identification permits afterward to assign a specif TPM to track, modeling in a better way the tracked object behavior.

In the first part we present the general principle. The class computation is exposed in a second part. In the third part we explain how a class is assigned to each track. The way the classification module output is used in the multiple object tracking is shown in the fourth part. Finally a brief conclusion is given.

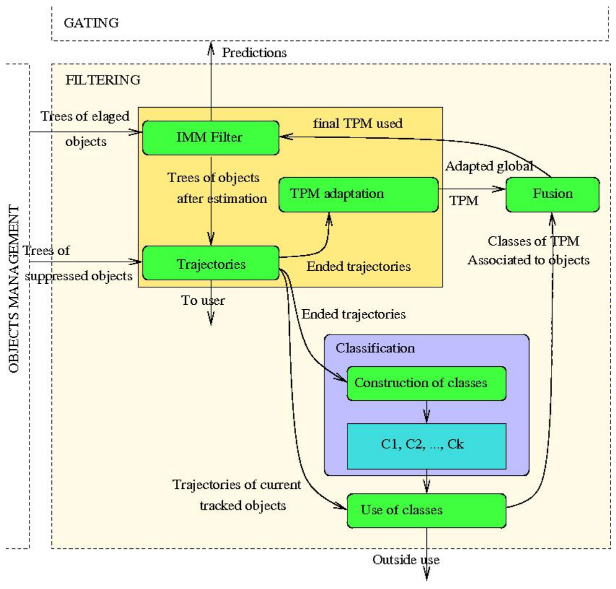

Fig. 4. General schema showing the classification use in the filtering process

1) Principle: The method's general principle is illustrated in the figure 4. The filtering part is again represented with yellow boxes as defined in the previous section. To specify for each track a TPM using a computed set of TPM classes, we add three blocs at our adaptive filtering and the algorithm 2 gives the final modified algorithm:

1) a classification component is added in order to take as an input the same trajectories used in the adaptation method *i.e* the most probable trajectories. This component is shown in blue on the figure 4. It computes a set of TPM classes, $C_1, C_2, \ldots, C_K$, which are stored in memory. This computation figure at lines 17 to 20 in the algorithm 2;
2) In a class usage component, the most probable TPM class is identify using the trajectory given as input. For each trajectory coming from each track, we compute the most probable TPM class (lines 21 to 28);
3) The last fusion component allows to combine the TPM obtain by automatic adaptation and the one computed using the classification. The result will then by used in each track for the next filtering step *via* the IMM (lines 29 to 36).

2) Classes computation: In a first part, TPM classes are computed using the whole set of trajectories given as input (lines 17 to 20 of the algorithm 2). The aim is to generate

Algorithm 2 Filtering using the classification module

```
1: /* Filtering takes as input current and deleted tracks */
2: **Filtering**($T_{elag,1:I}$ , $T_{suppr,1:J}$)
3: /* Compute estimations and predictions using the IMM filter */
4: for all i from 1 to I do
5:     IMM.computeEstimations($T_{elag,i}$)
6:     Return IMM.computePrediction($T_{elag,i}$)
7: end for
8: /* Compute the most probable trajectories */
9: for all i from 1 to I do
10:    $Traj_i \leftarrow$ mostProbableTrajectory($T_{elag,i}$)
11: end for
12: for all j from 1 to J do
13:    $Traj_j \leftarrow$ mostProbableTrajectory($T_{suppr,j}$)
14: end for
15: /* Global TPM adaptation */
16: $TPM_g \leftarrow$ TPM_Adaptation($Traj_{1:J}$)
17: /* Class building */
18: if number $Traj > N$ then
19:    $C_1, .., C_k \leftarrow$ CEM($Traj_{1:J}$)
20: end if
21: /* Class utilisation */
22: if number $Traj > N$ then
23:    for all i from 1 to I do
24:        if $Traj_{1:I} > L$ then
25:            $TPM_i \leftarrow$ MostProbableClass($Traj_{1:I}$)
26:        end if
27:    end for
28: end if
29: /* TPM Fusion */
30: for all i from 1 to I do
31:    if $TPM_i$ exist then
32:        Return $\frac{1}{Z}(TPM_i + TPM_g)$ in IMM
33:    else
34:        Return $TPM_g$ in IMM
35:    end if
36: end for
```

a set of TPM class which models objects behaviors using all achieved (*i.e* deleted) trajectories.

For each completed trajectory, be obtain the distribution over motion models as a sequence. This sequence is then used to compute a frequency matrix which is normalize to obtain a TPM. The process is the same as the one used in the autonomous adaptation but in this case the obtained TPM is local to each deleted track and not global. In this way, the computation for N tracks gives N TPM called *local* TPM but only one TPM called *global* is updated during the autonomous adaptation process.

The purpose is then to compute classes using the set of N local TPM. The class computation is performed using Classification Expectation Maximization (CEM) [] in which classes are represented by multidimensional Gaussians. This classification algorithm have been chosen for its Gaussian modelisation and its iterative mode of action permitting to obtain an output after at any iteration. To apply CEM the local TPM are simply transformed in vector. Thus the $M \times M$ matrix are transformed in M^2 length vectors. By this way, classes are modeled by Gaussians of mean and covariance matrix with respectively a size of M^2 and $M^2 \times M^2$.

In practice the classes computation is made every N achieved tracks as per the autonomous adaptation and it is performed using the whole history of achieved tracks.

Furthermore, it is necessary to define the number of classes to use the CEM algorithm. Therefore this number depend on the user and the application.

3) Classes usage: In this part, a class is assign to each track (lines 21 to 28 of the algorithm 2). The aim is to compute the most probable class for a given track and thus identify the object behavior in order to assign to the track a more specified TPM, enhancing the next predictions. In this usage part, the current tracks are considered not as in the Classes computation. Nevertheless, only tracks with a significant lenght L are considered in order to obtain coherent classification.

The class identification from the set of classes $C_1, C_2, \ldots, C_K$ for each track T_i with a TPM local TPM_i is done by first simply transforming the TPM_i in vector V_i, with d its dimension. Then considering the mean μ_j and covariance Σ_j of each class C_j, the associated class is computed following:

$$C(V_i) = \underset{j}{\operatorname{argmax}} \frac{1}{(2\pi)^{\frac{d}{2}} det(\Sigma_j)^{\frac{1}{2}}} \exp -\frac{1}{2}(V_i - \mu_j)^T \Sigma_j^{-1} (V_i - \mu_j) \quad (2)$$

By this way, the class TPM is associated at each track and become its specified TPM resulting in a better prediction. Also, the associated class can be displayed, giving to the user information on the object behavior.

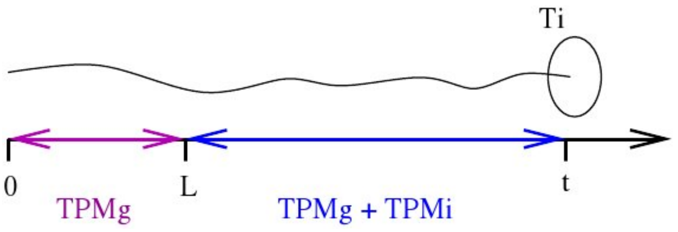

Fig. 5. Utilisation of local TPM for a specific track (TPM_i) and global TPM (TPM_g) according to the track lenght

4) Fusion of TPM: In this final part, the global TPM obtained by autonomous adaptation is merged with the specific TPM computed in the class identification (line 29 to 36 of the algorithm 2). It permits to specify the used TPM while keeping the advantages of the adaptive TPM *i.e* the global behavior of object and the robustness to behavior variation.

Nevertheless, as a specific TPM is only computed for tracks with a length above L, it is necessary to use only the global TPM for other tracks. This principle is illustrated by the figure 5. A track with a length above L will take advantages of both specific and global TPM. As both TPM are important and as complementaries advantages, an *ad hoc* fusion is performed by adding and normalizing the two associated matrix.

5) Conclusion: In this section, a classification module which come as an integrated part of the filtering as been presented. Using the local TPM of terminated tracks a set of class is computed modeling specific objects behaviors. This set permits in a second time to assign a specific TPM to each current track. This specific TPM is merged with the autonomous adapted TPM allowing to model the changed in behavior while using behavior of previously tracked objects.

V. EXPERIMENTAL RESULTS

In this section, an application of our method is presented. This application of our multi object tracking with classification method use real data obtained from the Daimler demonstrator. The aim is in a first time to automatically classify the two main objects behaviors in order to reuse the obtain class in the filtering process. In this normal traffic perception application the two main expected objects behaviors are traveling in the demonstrator direction and traveling in the opposite direction. And in a second time to use these two computed classes.

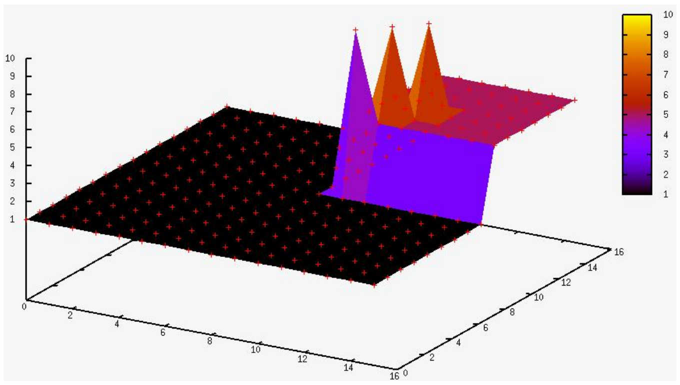

Fig. 6. First class initialisation

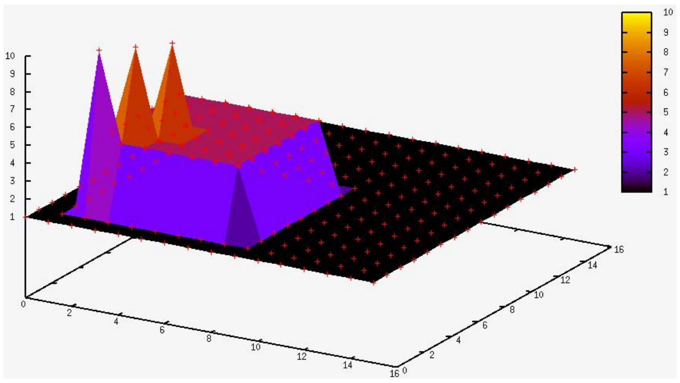

Fig. 7. Second class initialisation

6) Class computation results: Using our classification method, local TPM are computed for each terminated track. After obtaining a significant set of local TPM, two classes are

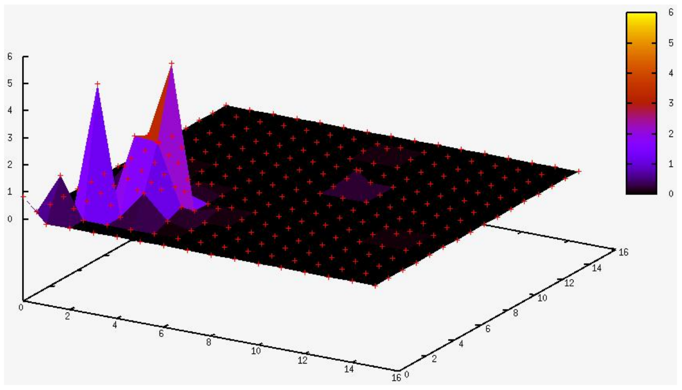
Fig. 8. Matrix vue modeling the first class mean obtained using 30 tracks

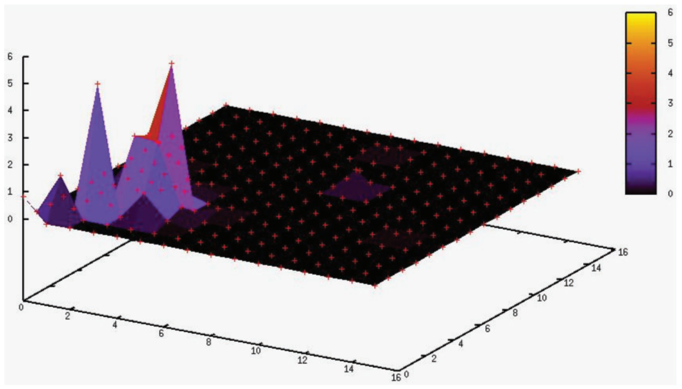
Fig. 9. Matrix vue modeling the second class mean obtained using 30 tracks

automatically computed. As explained, in our tracking application, 16 motions models are defined as shown in figure 3. So local and global TPMs are of size 16×16. And therefor the classes are modeled by Gaussians with 1)a mean vectors of size $16 \cdot 16$ *i.e* 256 constructed using the set of local TPM and 2) covariance matrix of size 256×256.

Given the number of tracks provided by the data set and the classes dimension, a very rough classes initialization is needed in order to ensure the convergence the CEM algorithm. Figures 6 and 7 show these two very approximative initializations.

The final classification output is demonstrated by figures 8 and 9. This classes are obtain after the completion of 30 tracks. We can clearly observe that first the two classes are well distinct and then that the modeled behavior correspond to the two expected main behavior. This classes can in a second stage be used to specify the TPM of the current tracks.

7) Classes utilisation results: The computed classes are in a second stage used during the tracking process and allow to specify a more specific TPM to each current track. this specification permit a better match between prediction model and real object behaviors. Also, it permits to assign a specific known behavior to each object allowing to provide useful information to the system user.

Figures 10, 11 and 12 show the specialization output. To illustrate the principle a object traveling in the opposite direction is considered. This object has been tracked for more

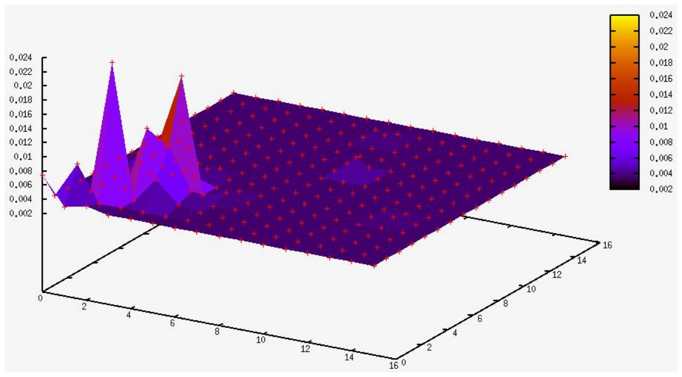
Fig. 10. Most probable TPM for this specific track

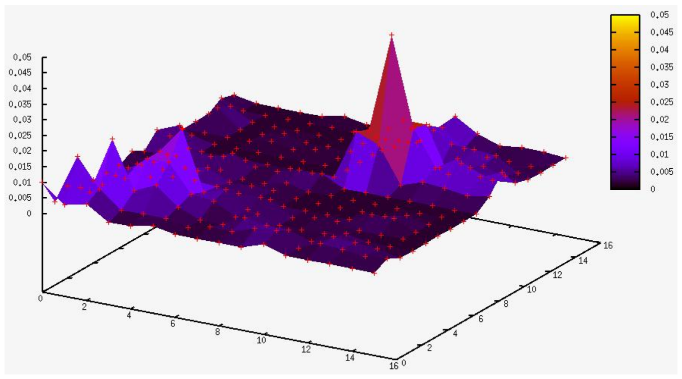
Fig. 11. Global TPM obtain using our adaptive method

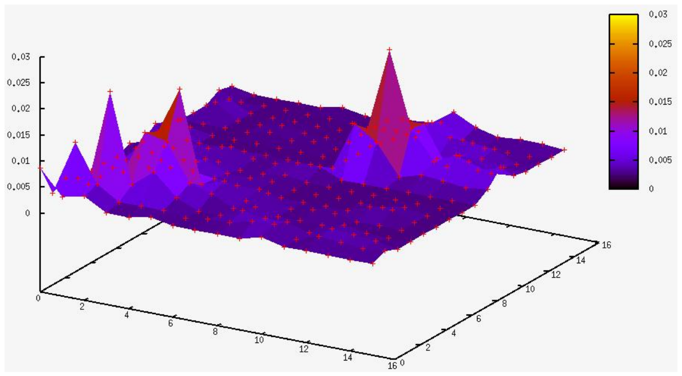
Fig. 12. TPM obtained after the fusion process

than L steps knowing that in our application a time step is 40 ms and that L has been set to 20. Therefore if an object is correctly tracked for less than 0.8 seconds, the global TPM is used alone for the IMM filtering, if not the global TPM and the classes TPM are used together for the filtering. In our illustration the object has been tracked for more than one second and the track is used to computed the most probable models sequence. This sequence is used to find the most probable class and thus assign a class TPM to the track, this TPM is shown in the figure 10. After, this class TPM and the global TPM automatically adapted using the whole set of deleted tracks (figure 11) are merged to obtain the specific TPM. this final TPM is used for all the next filtering stages.

The whole specialization process allow to take into account the whole set of observed tracks and the automatically computed set of classes. By this method adaptive and specific modellisation of behavior is considered improving the match between the real object motion and its prediction.

VI. CONCLUSIONS AND FUTURE WORKS

In this paper a method to perform object behavior classification is presented. This method comes in addition to a global approach to accomplish simultaneously online local mapping and moving object tracking. The classification process rely on the utilization of the TPM adaptation process included in the IMM filter (which is part of the MHT). This process permits in a first step to automatically compute classes modeling objects behavior and more specifically transition between motions via the TPM. In a second step, the classes are then use to specify online the TPM for each track, leading in a better object position prediction.

Results presented on real data using the Daimler demonstrator show that the method is able to automatically compute two classes corresponding to the two main object behaviors. Furthermore, a direct application of the class utilization as been presented showing the fusion between the autonomous global TPM adaptation and the classification result.

In perspectives, the way to obtain automatically the number of classes will be studied. Also, the information on the object belonging to a specific class can be further exploited, for instance for display purposed in the demonstrator by providing external informations to the driver on future object motion.

VII. ACKNOWLEDGMENTS

The work is supported by the European project PReVENT-ProFusion.

REFERENCES

[1] TD. Vu, J. Burlet, and O. Aycard. Grid-based localization and local mapping with moving objects detection and tracking. *International Journal on Information Fusion, Elsevier*, 2009. http://emotion.inrialpes.fr/aycard/Publications/2009/vu09.pdf*[pdf]*. To appear.

[2] Ruben Garcia, Olivier Aycard, Trung-Dung Vu, and Malte Ahrholdt. High level sensor data fusion for automotive applications using occupancy grids. In *IEEE International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2008. http://emotion.inrialpes.fr/aycard/Publications/2008/garcia08.pdf*[pdf]*.

[3] Qadeer Baig, Trung-Dung Vu, and Olivier Aycard. Online localization and mapping with moving objects detection in dynamic outdoor environments. In *IEEE International Conference on Intelligent Computer Communication and Processing (ICCP)*, 2009.

[4] TD. Vu, O. Aycard, and N. Appenrodt. Online localization and mapping with moving objects tracking in dynamic outdoor environments. In *IEEE International Conference on Intelligent Vehicles*, 2007. http://emotion.inrialpes.fr/aycard/Publications/2007/IV/vu07.pdf*[pdf]*.

[5] X. Rong Li and Vesselin P.Jilkov. A survey of maneuvering target tracking-part v: Multiple-model methods. *IEEE Transactions on Aerospace and Electronic Systems*, 2003.

[6] J. Burlet, O. Aycard, A. Spalanzani, and C. Laugier. Adaptive interacting multiple models applied on pedestrian tracking in car parks. In *IEEE/International Conference on Intelligent Robots and Systems*, 2006. http://emotion.inrialpes.fr/aycard/Publications/2006/IROS/burlet06b.pdf*[pdf]*.

[7] G. David Forney. The viterbi algorithm. *Proceedings of The IEEE*, 61(3):268–278, 1973.