ISSN 0005–1144
ATKAAF 46(1–2), 39–48 (2005)

António Almeida, Jorge Almeida, Rui Araújo

Real-Time Tracking of Multiple Moving Objects Using Particle Filters and Probabilistic Data Association

UDK 004.932'1
IFAC 5.9.3;2.5

Original scientific paper

Abstract-Mobile robots and vehicles are increasingly used in dynamic environments populated by humans and other moving objects and vehicles. In this context, tracking of surrounding moving objects is important for obstacle avoidance and motion planning. In this paper we present a method for detection and tracking of multiple moving objects using particle filters to estimate the object states, and sample based joint probabilistic data association filters to perform the assignment between the features detected in the input sensor data and filters. Filters management operations are required for appropriate integration of the currently perceived features. A real-time architecture, developed to implement the tracking system, is briefly described. Experimental results obtained with a laser range scanner will be presented demonstrating the feasibility and effectiveness of the presented methods.[1)]

Key words: mobile robots, particle filters, real-time tracking, probabilistic data association

1) This work was partially supported by FCT, Project POSI/SRI/42043/2001.

1 INTRODUCTION

With the continuous evolution of mobile robotics there will be autonomous vehicles in environments where other moving objects, including humans and, vehicles, evolve. Applications include transportation, home, factory, and office contexts. Learning models of dynamic environments is an important aspect for autonomous robot navigation [1]. Sensor-based methods for modeling and predicting the environment dynamics would increase the degree of anticipatory adaptation to environment evolution. In particular, tracking moving objects is important for motion planning and to anticipate and avoid collisions: giving correct state information of surrounding moving objects to a trajectory planning algorithm would and improve robot motion in dynamic worlds.

One of the problems of tracking is that dynamic objects move in patterns that are highly non-linear. Another important problem is simultaneously tracking multiple moving objects from a common set of sensor data. The object state estimation problem has been tackled using Kalman filters or extended Kalman filters (e.g. [2]). Kalman filters give optimal estimates for linear system and measurement models compounded with unimodal Gaussian noises. However, the Kalman approximation is often not accurate enough to model the non-linear, non-Gaussian, multi-modal characteristics of the system (object(s)) and sensors present in tracking. Extended Kalman filters permit the approximation of non-linear problems by linear models. Recently particle filters were introduced to estimate states for problems with non-linear non-Gaussian process and measurement models [3, 4].

This paper presents a method for tracking in real-time multiple moving objects in dynamic environments using particle filters. Particle filters are based on probabilistic representations of states by a set of samples (particles), with the advantage of making possible the representation of non-linear system and measurement models, and multi-modal non-Gaussian density states. For tracking several moving objects using a common sensor data set, a Sample-based Joint Probabilistic Data Association Filters (SJPDAF) algorithm [5] is used to estimate assignment probabilities between isolated segments on the perceived sensorial data vector (features), and the objects moving on the sensory perceptual range.

Section 2 briefly overviews particle filters, and Section 3 presents the SJPDAF framework. Section 4 presents details about perception and tracking. Section 5 presents the real-time system implementation architecture. Section 6 presents experimental results demonstrating the feasibility and effectiveness of the presented methods. Section 7 makes concluding comments.

2 ESTIMATION BY PARTICLE FILTERS

Particle filters are state estimation methods for systems with non-linear process and measurement models corrupted with noise which may be non--Gaussian and multimodal. These are recursive Monte Carlo (MC) statistical computing methods. Particle filters are an important alternative to Kalman filters which are optimal to linear systems corrupted with Gaussian noise.

Several methods for position determination and navigation use particle filters [6]. In this paper we describe the application of particle filters for tracking moving objects, where estimation of objects positions are based on measurements from a laser range finder.

The key idea of particle filters is to represent and maintain the posteriori density function by a set of random samples with associated weights and to compute the state estimate from those samples and those weights. As the number of samples becomes very large, this MC characterization becomes an equivalent representation to the usual function description of the posteriori probability density function (PDF), and the method approaches the optimal Bayesian estimate.

Let $\{x^i{}_{0:k}, \omega_k^i\}_{i=1}^{N_s}$ denote a random measure that characterizes the posteriori PDF $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$ where $\{\mathbf{x}_{0:k}^i,\ i=1,\ldots,N_s\}$ is a set of points with associated weights $\{\omega_k^i,\ i=1,\ldots,N_s\}$ and $\mathbf{x}_{0:k}=\{\mathbf{x}_j,\ j=0,\ldots,k\}$ and $\mathbf{z}_{1:k}=\{\mathbf{z}_j,\ j=1,\ldots,k\}$ are the set of all states and measurements up to time k, respectively. The weights are normalized such that $\sum_i \omega_k^i = 1$. In this way the posteriori density at k can be approximated as

$$p(\mathbf{x}_{0:k}\,|\,\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N_s} \omega_k^i \delta(\mathbf{x}_{0:k} - \mathbf{x}_{0:k}^i), \tag{1}$$

and we get a discrete approximation of the true posteriori probability $p(\mathbf{x}_{0:k}|\mathbf{z}_{1:k})$, where $\delta(\bullet)$ is the Dirac function, and the particle weights are given by

$$\omega_k^i = \omega_{k-1}^i \frac{p(\mathbf{z}_k\,|\,\mathbf{x}_k^i)\,p(\mathbf{x}_k^i\,|\,\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k^i\,|\,\mathbf{x}_{k-1}^i,\mathbf{z}_k)}. \tag{2}$$

To get equation (2), the principle of *importance sampling* [7, 8] was used. This principle states that the weights are taken from an *importance density* $q(\bullet)$.

After some studies in this area, results conclude that this algorithm has a degeneracy problem, and that an acceptable measure for this value is the effective sample number, N_{eff}, that was introduced by [9] and [10], and is represented by

$$N_{eff} = \frac{N_s}{1+Var\left[\omega_k^{*i}\right]} \tag{3}$$

where $\omega_k^{*i} = p(\mathbf{x}_k^i|\mathbf{z}_{1:k})/q(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i,\mathbf{z}_k)$ is named the »true particle weight«.

The exact value of equation (3) is not determinable. Therefore, an approximation is given by

$$\hat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s}(\omega_k^i)^2} \tag{4}$$

where ω_k^i is the normalized weight. N_{eff} is always less or equal than N_s, thus, when N_{eff} is much smaller than N_s there is a big degeneracy.

This problem could be solved with a good choice of the sampling importance to calculate the correct particle weights, and with a resampling step introduced in the particle filter dynamics.

In our implementation, the importance density is $p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$ from which we need to take samples. A sample $\mathbf{x}_k^i \quad p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$ can be generated by first generating a process noise sampling $\mathbf{v}_{k-1}^i \quad p_v(\mathbf{v}_{k-1})$ and setting $\mathbf{x}_k^i = f_k(\mathbf{x}_{k-1}^i, \mathbf{v}_{k-1}^i)$, where $p_v(\bullet)$ is the PDF of $\mathbf{v}_{k-1}$. For this particular choice of the importance density the weights are given [7] by

$$\omega_k^i \propto \omega_{k-1}^i p(\mathbf{z}_k\,|\,\mathbf{x}_k^i). \tag{5}$$

- Get N ordered numbers:
  $u_k = \frac{(k-1)+\tilde{u}}{N}$ with $\tilde{u} \sim U[0,1[$
- Make n_i copies of particle x_i for the new distribution:
  $n_i =$ number of $u_k \in]\sum_{s=1}^{i-1}\omega_s, \sum_{s=1}^{i}\omega_s]$

Fig. 1 Overview of the systematic resampling algorithm

In our implementation, a resampling step (Systematic Resampling [11], Figure 1) is applied. This step permits the reduction of the effects of degeneracy, observed in the basic particle filter algorithm. The basic idea is to eliminate particles that have small weights and to concentrate on particles with large weights [7]. After the resampling step, applied at every time index, all the particles take the same weight. In this way $\omega_{k-1}^i = 1/N\,\forall i$, and it follows from equation (5) that

$$\omega_k^i \propto p(\mathbf{z}_k\,|\,\mathbf{x}_k^i). \tag{6}$$

The algorithm is presented in Figure 2 and illustrated in Figure 3.

```
• FOR i=1:$N_s$
    - Draw: $\mathbf{x}_k^i \sim p(\mathbf{x}_k|\mathbf{x}_{k-1}^i)$
    - Calculate $\omega_k^i = p(\mathbf{z}_k|\mathbf{x}_k^i)$
• END FOR
• Calculate total weight:
  $t = \sum_{i=1}^{N_s} \omega_k^i$
• FOR i=1:$N_s$
    - Normalize:$\omega_k^i = t^{-1}\omega_k^i$
• END FOR
• Systematic Resampling
```

Fig. 2 Overview of the particle filter algorithm

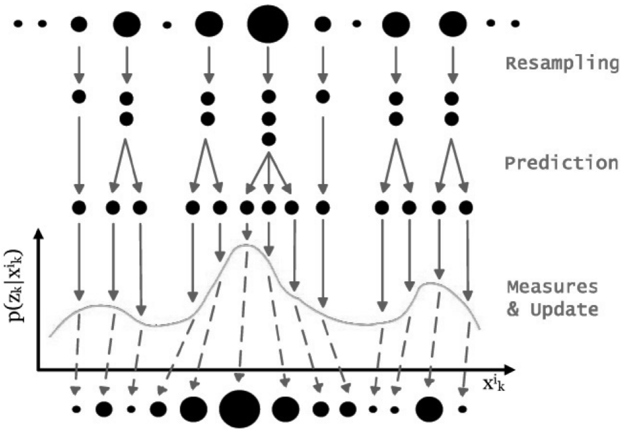


Fig. 3 Particle filter estimation cycle

3 SAMPLED-BASED JOINT PROBABILISTIC DATA ASSOCIATION FILTERS

One way to track various moving objects with particle filters is to estimate the compound state of all objects. However, this method becomes impracticable even for a small number of objects since computation grows exponentially in the number of objects. This problem can be overtaken by tracking the objects in individually. A data association problem problem arises in this context: to determine which measurement is caused by which object. In this paper we apply Joint Probabilistic Data Association Filters (JPDAF) [5] for this purpose.

The JPDAF algorithm is an extension of Probabilistic Data Association algorithm, that is able to track various moving objects at the same time and with the same set of measurements. The JPDAF, calculates the probabilities of association from the last set of measurements $\mathbf{z}_k$ to the various objects. Each object has its prediction and measurement models – state estimation is performed separately for each object.

Let $\mathbf{x}_k = \left\{\mathbf{x}_k^1,\ldots,\mathbf{x}_k^T\right\}$ denote the states of the T moving objects being tracked at instant k. Each $\mathbf{x}_k^i$ is a random variable in state space of a unique object. Let $\mathbf{z}_k = \left\{\mathbf{z}_k^1,\ldots,\mathbf{z}_k^{m_k}\right\}$ be a set of measurements at instant k, where $\mathbf{z}_k^j$ is a feature from that set. $\mathbf{Z}_k =$ $= \left\{\mathbf{z}_1,\ldots,\mathbf{z}_k\right\}$ is the sequence of measurements observed up to instant k. The key idea for tracking is how to associate the observed features to the individual objects.

In the JPDAF model, a joint association event θ is a set of pairs $(j,i) \in \{0,\ldots,m_k\} \times \{1,\ldots,T\}$. Each θ uniquely determines which feature is assigned to which object. Feature $\mathbf{z}_k^0$ is used to model currently undetected objects – no feature found for such objects. Let Θ_{ji} denote the set of all valid joint associations events which assign feature j to the object i. At time k, the JPDAF computes the posteriori probability that feature j is caused by object i according to

$$\beta_{ji} = \sum_{\theta\in\Theta_{ji}} P(\theta|\mathbf{Z}_k). \tag{7}$$

Assuming that the estimation problem is Markovian and using probability theory, the probability $P(\theta|\mathbf{Z}_k)$ of an individual joint association event can be calculated according to

$$P(\theta|\mathbf{Z}_k) = \int P(\theta|\mathbf{z}_k,\mathbf{x}_k)\,p(\mathbf{x}_k|\mathbf{z}_k,\mathbf{Z}_{k-1})\,d\mathbf{x}_k. \tag{8}$$

Here the state of the objects must be known to determine associations θ. Conversely, θ needs to be known in order to determine the objects positions. An incremental approximation is applied to overcome this issue. The key idea is to approximate $p(\mathbf{x}_k|\mathbf{z}_k,\mathbf{Z}_{k-1})$ by the belief $p(\mathbf{x}_k|\mathbf{Z}_{k-1})$ about the predicted state of the objects, i.e. the prediction computed using all measurements perceived before time-step k. According to this, we obtain

$$P(\theta|\mathbf{Z}_k) \approx \int P(\theta|\mathbf{z}_k,\mathbf{x}_k)\,p(\mathbf{x}_k|\mathbf{Z}_{k-1})\,d\mathbf{x}_k \tag{9}$$

$$= \alpha\int P(\mathbf{z}_k|\theta,\mathbf{x}_k)\,P(\theta|\mathbf{x}_k)\,p(\mathbf{x}_k|\mathbf{Z}_{k-1})\,d\mathbf{x}_k \tag{10}$$

where α is a normalizer factor ensuring that $P(\theta|\mathbf{Z}_k)$ sums up to one over all θ. The term $P(\theta|\mathbf{Z}_k)$ corresponds to the probability of the assignment θ given the current objects states, and is approximated as a constant. Assuming that each feature is detected independently from the others, we get

$$P(\mathbf{z}_k|\theta,\mathbf{x}_k) = \gamma^{(m_k-|\theta|)}\prod_{(p,q)\in\theta}\int p(\mathbf{z}_k^p|\mathbf{x}_k^q)\,p(\mathbf{x}_k^q|\mathbf{Z}_{k-1})\,d\mathbf{x}_k^q, \tag{11}$$

where $\gamma^{m_k-|\theta|}$ is the probability of all false alarms (features without object in a perception cycle) in $\mathbf{z}_k$ given θ. Using (11) in (10), and inserting the result in equation (7) we obtain

$$\beta_{ji} = \sum\nolimits_{\theta\in\Theta_{ji}}$$
$$\left[\alpha\gamma^{(m_k-|\theta|)}\prod\nolimits_{(p,q)\in\theta}\int p(\mathbf{z}_k^p\,|\mathbf{x}_k^q)p(\mathbf{x}_k^q\,|\mathbf{Z}_{k-1})d\mathbf{x}_k^q\right]. \quad (12)$$

Once the assignment probabilities are calculated, the updated estimate of the states is obtained as follows

$$p(\mathbf{x}_k^i|\mathbf{Z}_k) = \alpha\sum_{j=0}^{m_k}\beta_{ji}\, p(\mathbf{z}_k^j|\mathbf{x}_k^i)\, p(\mathbf{x}_k^i|\mathbf{Z}_{k-1}), \quad (13)$$

where $p(\mathbf{z}_k^i\,|\mathbf{x}_k^i)$ is the measurement model of the system and $p(\mathbf{x}_k^i\,|\mathbf{Z}_{k-1})$ is the previous estimate projected to instant k using the system model.

Since particles are used to describe the density function, we use the SJPDAF proposed in [12], so the method can be applied to a discrete representation. The idea is to represent the density $p(\mathbf{x}_k^i\,|\mathbf{Z}_{k-1})$ by a set of N random samples, or particles that constitutes a discrete approximation of a PDF. Here, each particle consists on a pair $(\mathbf{x}_k^{i,n},\omega_k^{i,n})$, where $(\mathbf{x}_k^{i,n})$ is the state and $(\omega_k^{i,n})$ is the importance factor. The prediction step of Bayesian filtering is performed by drawing samples from the set computed in the previous iteration and by updating their state according to the prediction model $p(\mathbf{x}_k^i\,|\mathbf{x}_{k-1}^i,t)$. In the correction step, a measurement $\mathbf{z}_k$ is integrated into the samples obtained in the prediction step. With samplebased representation the integration of equation (13) can be done by summing over all samples generated after the prediction step, and we get

$$\beta_{ji} = \sum_{\theta\in\Theta_{ji}}\left[\alpha\gamma^{(m_k-|\theta|)}\prod\nolimits_{(p,q)\in\theta}\frac{1}{N}\sum_{n=1}^{N}p(\mathbf{z}_k^p\,|\mathbf{x}_k^{q,n})\right]. \quad (14)$$

To help understand equation (14) lets look at Figure 4. This figure represents several possible combinations between the detected features and the active particle filters in the present moment. The upper right matrix represents a combination where an object has some probability of being occluded by another. While in all the other combinations all the present objects are detected and attributed to one one of the features produced by the present measurements. Making all the possible combination for each pair (i,j) (represented by equation (14)), the value of each β_{ji} could be represented by a matrix like the one represented in Figure 5.

With assignment probabilities computed, the weight of each particle can be calculated by

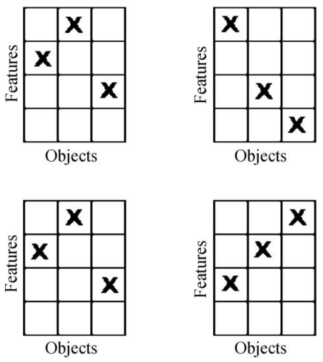
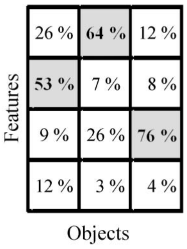


Fig. 4 Possible combinations of feature/object



Features			
	26 %	64 %	12 %
	53 %	7 %	8 %
	9 %	26 %	76 %
	12 %	3 %	4 %
	Objects		

Fig. 5 Possible values of feature/object combinations (β_{ji})

$$\omega_k^{i,n} = \alpha\sum_{j=0}^{m_k}\beta_{ji}\,p(\mathbf{z}_k^j\,|\mathbf{x}_k^{i,n}). \quad (15)$$

α is a normalizing factor, such that all weights sum to one.

4 PERCEPTION AND TRACKING

A SICK LMS200 laser range finder was used as source of sensory data. The laser was programmed to transmit 361 measurements per scan, evenly distributed in a 180 degrees angular range (0.5 degrees resolution).

Constant resolution grid representations are used in several processing steps for abstracting the sensor measurements to the filter. Space is divided into square cells with 10 cm side.

A. Grid Models

The *occupation grids* contains an estimate of the occupation probability of each cell, in the current instant. This probability is estimated as follows for each cell (x, y)

$$p_{xy} = \psi\frac{N_{xy}}{M_{xy}} \quad (16)$$

where N_{xy} represents the number of points in the laser measurement vector that falls inside the (x, y) cell. x and y are the coordinates of the point of the cell that is closest to the origin. ψ is an adjustment factor and M_{xy} is the maximum number of laser points that cell (x, y) can receive. This maximum number depends on the position of the cell with respect to the laser and is calculated as follows:

$$M_{xy} = \frac{180}{|x|+|y|+1}, \tag{17}$$

where the value 180 is due the laser angular resolution of 180 sensor points on the half laser aperture of 90 degrees. M_{xy} is used in the computation of other grids discussed below. Using (16) and (17) a probability distribution $P(occupied_{x,y}|\mathbf{Z}(k))$ is obtained.

The *new occupation grid* contains, the probability of each cell being currently occupied but not occupied in the previous iteration. This grid is constructed by confronting the actual occupation grid with the grid of the last iteration as follows

$$\begin{aligned} P(new^{k}_{x,y} \mid \mathbf{Z}(k),\mathbf{Z}(k-1)) &= P(occupied_{x,y} \mid \mathbf{Z}(k)) \times \\ &\times (1 - P(occupied_{x,y} \mid \mathbf{Z}(k-1))), \end{aligned} \tag{18}$$

where $P(new^{k}_{x,y} \mid \mathbf{Z}(k),\mathbf{Z}(k-1))$ is the probability that an object has moved into cell (x,y) [12].

The *occlusion grid* has in each cell the probability of that cell being occluded by a object (moving or not) relative to the laser. This grid is used for direct treatment of occluded objects during tracking. Each point of the laser measurement vector is analyzed and the cells containing segments of the point's occlusion line are updated on the occlusion grid. For each cell (x,y) a probability $P(occlusion^{k}_{x,y} \mid \mathbf{Z}(k))$ is obtained.

B. Segmentation

To identify the mobile objects in the laser monitoring space we use a segmentation process in Cartesian coordinates. This process consists of grouping the neighboring points, in such a way that we can perform a segment, according to a used criterion. In this way we can work with sets of points instead of the laser measurements. A segment is therefore a set o adjacent laser measurements values close to each other. We stipulate that two consecutive points belong to the same segment if the distance between these points is lower that a threshold value D, given by

$$(x^{j+1}_{k} - x^{j}_{k})^2 + (y^{j+1}_{k} - y^{j}_{k})^2 = D^2_{k}, \tag{19}$$

where x^{j}_{k} and y^{j}_{k} are the coordinates of point j, and x^{j+1}_{k} and y^{j+1}_{k} are the coordinates of the point $j+1$ of the laser measurements vector. The D value is selected in order to adjust the process performance.

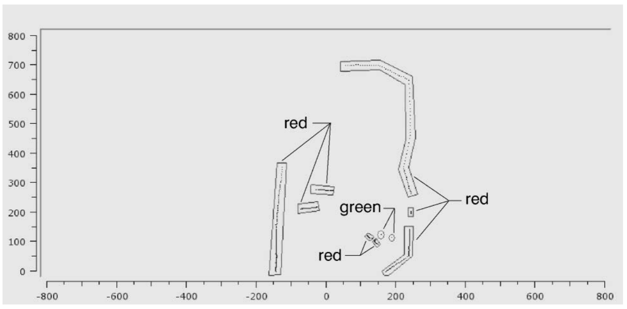


Fig. 6 Laser range scan where exists eight segments

Figure 6 illustrates an example of the segmentation process. There are eight different segments (that are distinguished with red outline). Notice that the isolated points are ignored (that are distinguished with green outline). These points do not actually correspond to any object existing in the environment. They are noisy laser measurements that result from the laser incidence in a transition area, and from the fact that, in this way, the corresponding reflected laser beam becomes diffused and with reduced power.

C. Mobile objects

The final step in the perception involves the fusion of segmentation results and the grid results to identify the moving objects. They are identified by confronting the minimum grid and the new occupation grid with the segmentation results. After this process we have a new grid with probability distribution $P(new\ object_{x,y} \mid \mathbf{Z}(k))$, that indicates the cells occupied by moving objects. This new grid is the input of the SJPDAFs method.

D. Tracking

With SJPDAF theory, it is possible to track various mobile objects using a particle filter for each object. The SJPDAF algorithm returns the assignment probabilities that permit each feature be associated to one filter.

When a new object steps into the laser perceptual range (PR), a new filter must be initialized and starts to track that object. To initialize the new filter around the new object, it is checked which feature does not have a filter tracking it. This verification is performed by analyzing the distances between the mean points of the features, and the mean points of the particles of the filters. A new filter is associated to feature

$$j_{\text{new}} = \underset{j \in \text{Features}}{\arg\max} \; \underset{i \in \text{Filters}}{\min} \; d(mt_j, ml_i), \tag{20}$$

where mt_j and ml_i are the mean points of the features, and the mean points of the particles of the filters. Particles of the new filter are spread around $mt_{j\text{new}}$. Each particle is a vector containing position, orientation and velocity components: (x, y, θ, ϑ). This process is iterated until the number of filters equals the number of features.

When an object steps out of the laser PR, one of the active filters must be closed. For each filter, i, an accumulated discounted average $\hat{\Lambda}_k^i$ of the particles weights sum before the normalization step Λ_k^i is maintained:

$$\hat{\Lambda}_k^i = (1-\eta)\hat{\Lambda}_{k-1}^i + \eta\Lambda_k^i, \tag{21}$$

where η is a constant that regulates the inertia of the process. For each of the $T-m_k$ filters with the least value of $\hat{\Lambda}_k^i$ there is a variable ρ^i initialized to *Max_Cont* when the feature is no longer perceived on the sensor data, and ρ^i is decremented while this situation persists. When ρ^i attains 0, the filter is deactivated. If the feature reappears in the measurements while the filter is in this transition phase, then the filter is not deactivated, and variable ρ^i is re-initialized to *Max_Cont*. These variables permit that a filter will not be initialized just because a temporary individual feature, or a filter will not be closed just because a feature disappears temporarily.

The overall tracking algorithm can be divided into five parts: (i) Prediction step; (ii) Correction step; (iii) Assignment probabilities calculation using (14); (iv) Particles weights update using (15); (v) Resampling step.

The following prediction model was used on the particle filters is described by equations (22)–(25):

$$x_{k+1}^i = x_k^i + x_k^i \vartheta_k^i h \cos(\theta_k^i), \tag{22}$$

$$y_{k+1}^i = y_k^i + y_k^i \vartheta_k^i h \sin(\theta_k^i), \tag{23}$$

where h is the sample period. Direction parameter θ and velocity parameter ϑ, at every time index, are affected by an independent Gaussian noise. θ and ϑ equations are

$$\theta_{k+1}^i = \theta_k^i + v_1 n_1, \tag{24}$$

$$\vartheta_{k+1}^i = \vartheta_k^i + v_2 n_2, \tag{25}$$

where n_1 and n_2 are zero-mean Gaussian random processes with unit variance. Factors v_1 and v_2 adjust the variance for orientation and velocity, respectively.

In the correction step, the calculation of measurements probabilities given the state vector is performed. In this step $p(\mathbf{z}_j(k)\,|\,\mathbf{x}_k^{i,n})$ is calculated. This is needed to compute the assignment probabilities β_{ji}. For $j=0$, we have the probability of the object not being detected:

$$p(\mathbf{z}_0(k)\,|\,\mathbf{x}_k^{i,n}) = p(occluded_k^{\mathbf{x}_{i,n}}) = \\ = P(occlusion_{x,y}^k\,|\,\mathbf{Z}(k)), \tag{26}$$

where $\mathbf{x}_{i,n} \in \text{cell}(x,y)$, i.e. $p(occluded_k^{\mathbf{x}_{i,n}})$ represents the probability of particle n from the filter i be occluded at time k, as obtained from occlusion grid (see also Sec. 4-A). $p(\mathbf{z}_j(k)\,|\,\mathbf{x}_k^{i,n})$ for $j=1,\ldots,k$, is obtained from the grid that represents cells occupied by moving object j (cf. Sec. 4-A):

$$p(\mathbf{z}_j(k)\,|\,\mathbf{x}_k^{i,n}) = p(movobject_k^{\mathbf{x}_{i,n}}) = \\ = P(movobject_{x,y}^k\,|\,\mathbf{Z}(k)), \tag{27}$$

where $\mathbf{x}_{i,n} \in \text{cell}(x,y)$, i.e. $p(movobject_k^{\mathbf{x}_{i,n}})$ is the probability that particle n of filter i is on a moving object position.

5 SYSTEM ARCHITECTURE

This section is divided in two subsections: one to describe the real-time tracking architecure in the PC, and the other to describe the CAN communication between the PC and the Laser range finder.

A. Real-Time Tracking Architecture

Figure 7 depicts the architecture of mobile objects tracking. The first step is to get the measurements vector from laser range finder. With this vector an »*Occupation Grid*« is built and, at the same time, this vector is decomposed in several segments, with the criteria described in Section 4. The segments that correspond to minima in the laser distances vector are used to build the spatial distribution of the probability of features which is called the »*Features map*«. With the aid of the occupation grid from the last iteration process, we can determine which grid elements are newly occupied in the laser monitored space. In this way we can create a »*New Occupation Grid*«. With the conjugation

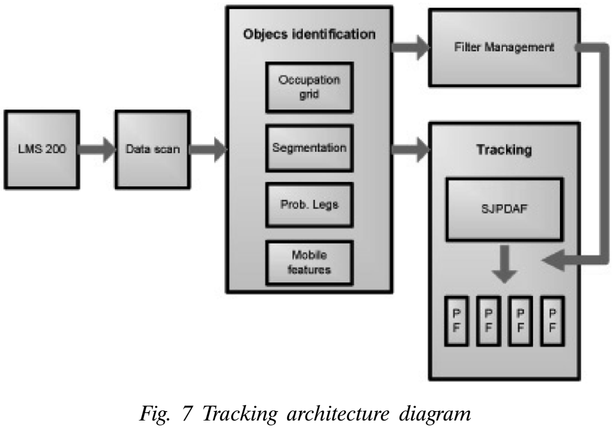


Fig. 7 Tracking architecture diagram

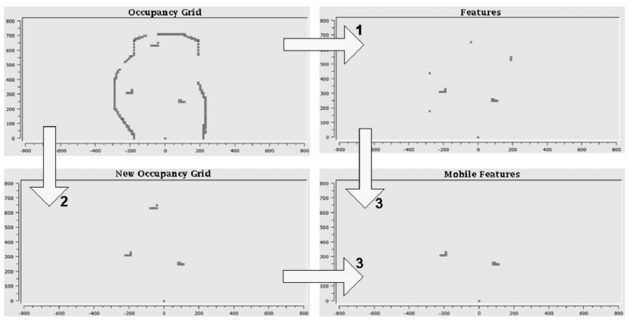


Fig. 8 Mobile features determination

of these Feature Map and the New Occupancy Grid maps we create another grid that only contains the mobile features. Figure 8 illustrates this process of determining the mobile objects. Once we have the separated mobile segments we can update the probability of association between mobile feature and particle filters, using equation (14), to perform tracking of these mobile features.

B. CAN Architecture

The communication between the PC running the algorithms for tracking moving objects, and the SICK LMS200 laser range scanner was established through a CAN (Controller Area Network) bus. This bus permits the accommodation of the 500 Kbaud sensor data transfer rate of the LMS200.

Since the communication with the laser is performed with a RS422 serial channel, a microcontroller-based interface node was implemented for bridging the laser sensor and PC through the CAN bus (Figure 9). The microcontroller (μC) interface implements an algorithm to convert CAN format to serial format and vice versa. A protocol was developed to communicate the laser telegrams (messages) between the μC and the PC through the CAN. For protocol management, a header is annexed to the telegram sent from the PC to the laser. The header and telegram form what we call a CAN packet (Figure 10). Clearly, this header will not be sent to laser. This header consist in four fields (1 byte each): (1) *number of CAN messages* into which the packet was subdivided for sending from the PC to the μC (the CAN protocol defines a maximum message length of 8 data bytes) (Figure 10); (2) *operation code*, which permits the μC to identify the type of the telegram received, so it can proceed with the appropriate actions in the current operation context. (3) *communication velocity* of the interface serial port, that permits μC to know when to change velocity of its serial port; (4) *telegram size* to permit the μC to extract the annexed telegram from the received message, so it can be sent to the laser. To understand the importance of this proto-

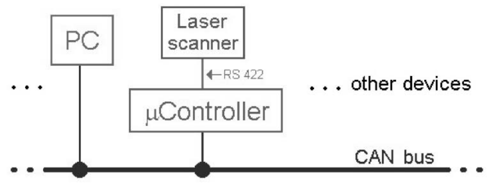


Fig. 9 Communication between PC and laser by the microcontroller

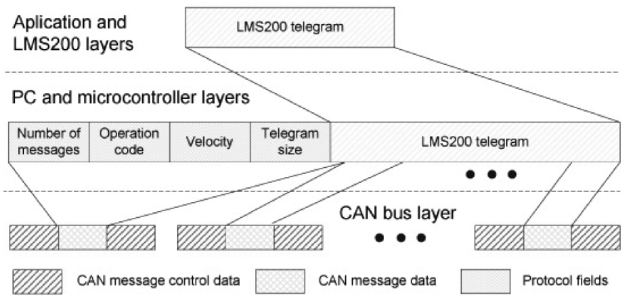


Fig. 10 Packet communication from PC to μC through CAN messages

col notice that the laser can function in continuous mode (measurements sending) or in request mode, making the required μC processing to be completely different. On the other hand, if the sent telegram changes the laser communication velocity, the μC has to reconfigure its own serial port speed, for communication to be continued.

In the laser to PC way, the μC only divides the received telegrams from laser in CAN messages (up to 8 bytes) and puts it on the CAN bus for PC reassembly. The overall effect of this real-time communication interface is to permit a totally transparent communication between the PC and the laser, i.e. the communication is performed as if the laser was directly connected to the PC.

6 EXPERIMENTAL RESULTS

All experiments were performed in an indoor environment where the scenery could be changed from test to test. The laser was placed at a height of 80 cm. The tracking algorithm was implemented with a frequency of 5 iterations per second. The used PC has a Pentium III processor at 1 GHz and 512 Mbytes of RAM. The linux operating (kernel 2.24.25) and the Real-Time Application Interface (RTAI 3.0) were used.

A. Tracking Multiple Objects

In this test several people are tracked in order to demonstrate the tracking algorithm.

The sequence of Figure 11 starts with a single person being tracked. At the third displayed time sample (row) a new object enters the laser PR, a

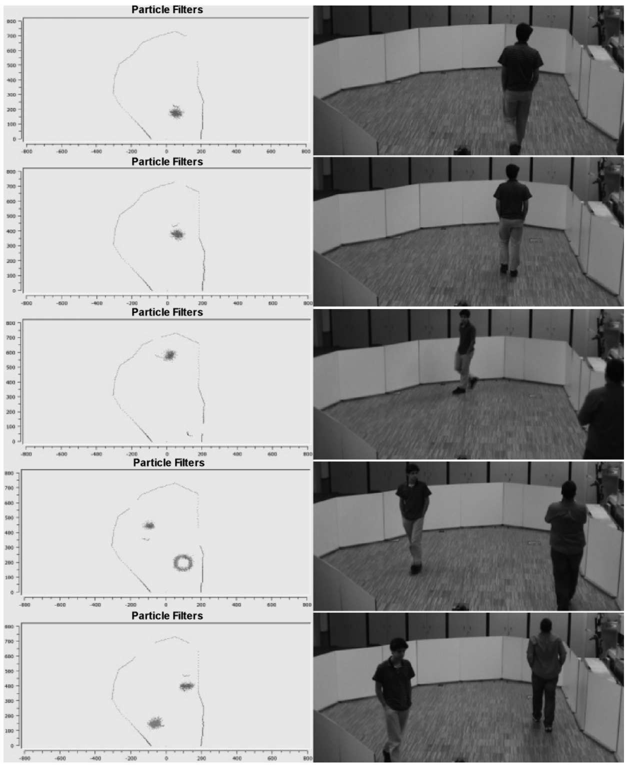


Fig. 11 Tracking two people

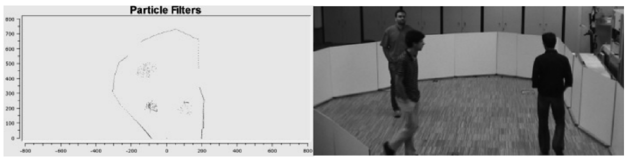


Fig. 12 Tracking three people

corresponding feature was detected, and a new filter has been started and is already tracking that person. However its particles do not appear in the image according to the count variable explained in Sec. 4. In the last row the new filter is already represented. From row four to row five we can see that the filters interchange their features. This is a result of SJPDAF utilization and by random filters evolution. Figure 12 presents a situation, observed later on the same experiment, where there are three people in the PR. It can be observed in this row that the particles of the three active filters have lower diversity. One fact that could justify this occurrence is the used Systematic Resampling that lowers the particles diversity and can provoke the degeneration of the filters. When the tracked mobile object has abrupt motion and a few number of particles are predicted inside the feature, the resampling step makes many equal copies. Although the number of filter particles is always the same, low samples diversity is due to particles being superimposed and have the same direction and velocity (x, y, θ, ϑ). Anyway, in subsequent iterations, the randomness introduced in the filter prediction phase, mitigates this problem, and samples diversity is restored. However, there are rare situations in which the particles of a filter have irrelevant weights and the filter gets a high degenerate level. In such situations the filter has difficulties to track the objects and it is necessary to reinitialize the filter.

A second fact that affects the operation of the filters is that the tracked mobile objects are not always detected, as in the case that a person inverts his/her trajectory, thus staying immobile for some instants. In such situations the new occupation grid does not have a corresponding feature present, and the probability $p(\mathbf{z}_k^j \,|\, \mathbf{x}_k^{i,n})$ is affected, which in turn causes an incorrect evolution of particles. Figure 13 corresponds to this test and specifies the number of features, active filters, and combinations of valid associations made in assignment probabilities computation referenced in Eq. (14). At the iterations corresponding to time instant of Figure 12 (inside the [140, 200] time interval) an instability in the number of mobile features can be observed. However, these inverted peaks are quick and do not influence the number of active filters due the inertia created by the filters closing algorithm (Sec. 4) and in this way each feature always has a filter tracking it. Tracking three people the number of combinations of valid associations is 64. This number grows to 7776 when five people are tracked.

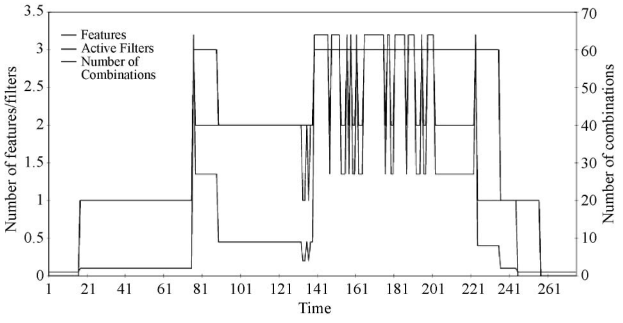


Fig. 13 Number of features detected, number of active filters and number of associations during one experiment of tracking five people

Figure 14 shows another set of data related to the same experiment. It represents the laser scan vector, and the middle point of each particle filter. In the left hand image of each row, all values from the laser vector are represented by vertical lines. The minimum values in this vector, which are caused by the presence of three mobile objects, are perfectly visible. Notice that there are 361 values

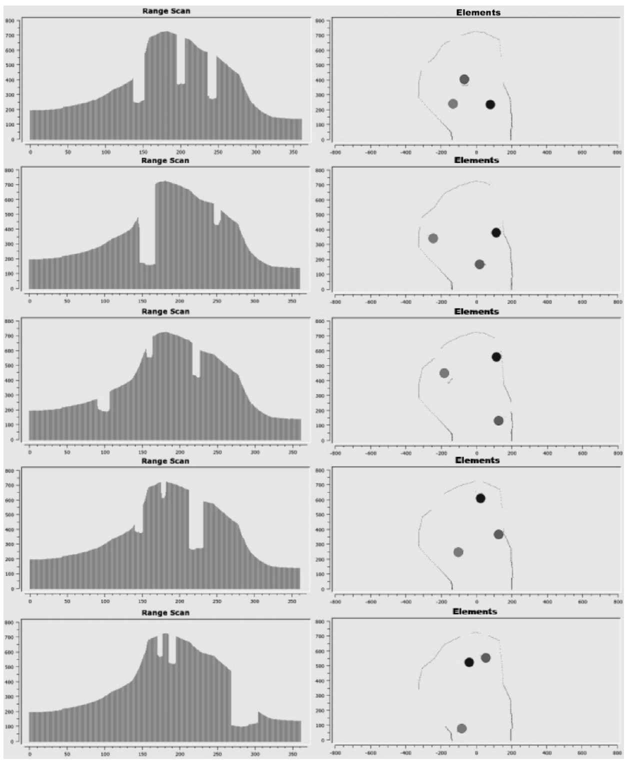


Fig. 14 Scan measurements and particle filter middle point

in each laser vector. Three local minima sets of points (clusters) can be seen in each row, except in the second row where temporarily there are only two local minima but the filter management operations (Sec. 4) retain three filters.

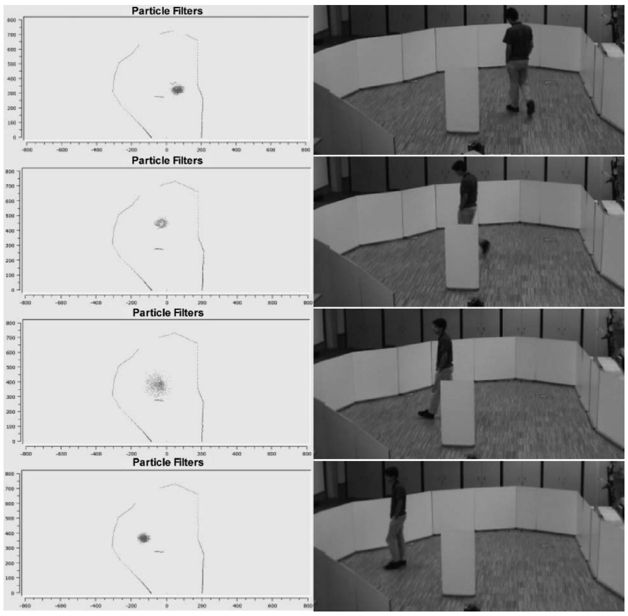


Fig. 15 Sequence of a occlusion situation

B. Occluded Objects

In order to demonstrate the advantage of handling the occlusion of mobile objects, Figure 15 presents a sequence where a person moves behind a static object. In this situation, the particles of the filter disperse in a random way consistently with the temporary lack of sensory data and corresponding increase of filter state uncertainty. In the first row the filter tracks the mobile object as usual. In the second and third samples, since the feature is not detected in the sensory data, the particles of the filter are assigned to the occlusion feature, and predicted over the corresponding area. Once the feature reappears in the range measurements, the filter rearranges its particles in order to track the feature again with lower uncertainty, as show in row (time-stamp) four.

7 CONCLUSIONS

This paper has presented probabilistic methods for tracking multiple moving objects, using sensory data from a laser range scanner. The developed system uses SJPDAF to handle the data association problem, and a particle filter to individually track each object. Particle filters have the advantage that they can be applied to nonlinear and non-Gaussian systems. A method was also presented for perception of moving objects, and separate moving objects from all the static objects existing in the environment, based in probability occupancy grids and obstacle segmentation. The paper also presented a realtime architecture that was developed to link all system components and implement the tracking algorithms. The architecture permits a completely transparent communication between the laser sensor and the host computer. Future work includes improving particle filter behavior regarding the tracking of highly abrupt motions.

REFERENCES

[1] R. Araújo, G. Gouveia, N. Santos, **Learning Self-organizing Maps for Navigation in Dynamic Worlds.** in Proc. IEEE Int. Conf. Robotics and Automation (ICRA), 2003, pp. 1312–1317.

[2] A. Fod, A. Howard, M. J. Mataric, **Laser-based People Traking.** in Proc. IEEE Int. Conf. Robotics and Automation (ICRA), 2002, pp. 3024–3029.

[3] M. Pitt, N. Shephart, **Filtering via Simulation: Auxiliary Particle Filters.** In J. Amer. Statist. Assoc., vol. 94, no. 446, 1999, pp. 590–599.

[4] N. J. Gordon, D. J. Salmond, A. F. M. Smith, **A Novel Approach to Nonlinear/non-gaussian State Estimation.** In IEE Procedings F, vol. 2, 1993, pp. 107–113.

[5] Y. Bar-Shalom, T. E. Fortmann, **Traking and Data Association**, Ser. Mathematics in Science and Engeneering. Academic Press, 1988, vol. 179.

[6] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, P.-J. Nordlund, **Particle Filters for**

Positioning, Navigation and Tracking. IEEE Trans. Signal Processing, vol. 50, no. 2, pp. 425–437, 2002.

[7] M. S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, **A Tutorial on Particle Filters for Online Nonlinear/non-gaussian Bayesian Tracking.** IEEE Transactions on Signal Processing, vol. 50, no. 2, pp. 174–188, Feb. 2002.

[8] A. Doucet, **On Sequencial Monte Carlo Methods for Bayesian Filtering.** Dept. Eng., Univ. Cambridge, UK, Tech. Rep., 1998.

[9] R. C. J. S. Liu, **Sequencial Monte Carlo Methods for Dynamical Systems.** In J. Amer. Statist. Assoc., vol. 93, 1998, pp. 1032–1044.

[10] N. Bergmam, **Recursive Bayesian Estimation: Navigation and Tracking Aplications.** Ph.D. dissertation, Linköping Univ., 1999.

[11] J. D. Hol, **Resampling in particle filters.** Institutionen för Systemteknik, Linköpings Univ., Tech. Rep., May 2004.

[12] D. Schulz, D. Fox, W. Burgard, A. B. Cremers, **People Tracking with a Mobile Robot Using Sample-based Joint Probabilistic Data Association Filters.** In International Journal of Robotics Research (IJRR), vol. 22, no. 2, 2003, pp. 99–116.

Praćenje više gibajućih objekata u stvarnome vremenu primjenom čestičnih filtara i vjerojatnosnog pridruživanja podataka. Mobilni roboti i mobilna vozila sve se više koriste u dinamičkim okruženjima popunjenim ljudima i drugim gibajućim objektima. U tom je smislu važno praćenje gibajućih objekata u neposrednom okruženju kako bi se izbjegavale prepreke i planiralo gibanje. U ovome je radu predložena metoda detekcije i praćenja više gibajućih objekata primjenom čestičnih filtara za estimaciju stanja objekata i filtara za združeno vjerojatnosno pridruživanje uzorkovanih podataka kojima se povezuju značajke detektirane u mjernim podacima s odgovarajućim filtrima. Izvedena je nadzorna ljuska filtara za odgovarajuću integraciju očitanih značajki. Ukratko je opisana arhitektura implementiranog sustava praćenja objekata u stvarnom vremenu. Prikazani eksperimentalni rezultati dobiveni primjenom laserskog senzora udaljenosti potvrđuju izvedivost i učinkovitost predloženog sustava.

Ključne riječi: mobilni roboti, čestični filtri, praćenje u stvarnome vremenu, vjerojatnosno pridruživanje podataka

AUTHORS' ADDRESSES

António Almeida
Jorge Almeida
Rui Araújo

ISR – Institute for Systems and Robotics
Department of Electrical and Computer Engineering
University of Coimbra
P-3030-290 Coimbra, Portugal

Received: 2005-12-01