Multiple Target Tracking For Mobile Robots Using the JPDAF Algorithm

Aliakbar Gorji, Saeed Shiry and Mohammad Bagher Menhaj

Abstract Mobile robot localization is taken into account as one of the most important topics in robotics. In this paper, the localization problem is extended to the cases in which estimating the position of multi robots is considered. To do so, the Joint Probabilistic Data Association Filter (JPDAF) approach is applied for tracking the position of multiple robots. To characterize the motion of each robot, two models are used. First, a simple near constant velocity model is considered and then a variable velocity model is applied for tracking. This improves the performance when the robots change their velocity and conduct maneuvering movements. This issue gives an advantage to explore the movement of the manoeuvring objects which is common in many robotics problems such as soccer or rescue robots. Simulation results show the efficiency of the JPDAF algorithm in tracking multiple mobile robots with maneuvering movements.

1 Introduction

The problem of estimating the position of a mobile robot in physical environment is an important problem in mobile robotics. Indeed, because of the intrinsic error embedded in the dead-reckoning of a mobile robot, the data gathered form the odometry are not trustable and, therefore, some operations should be conducted on the

Aliakbar Gorji
Amirkabir University of Technology, 424 Hafez Ave., 15914 Tehran, Iran, e-mail: gorji1983@aut.ac.ir

Saeed Shiry
Amirkabir University of Technology, 424 Hafez Ave., 15914 Tehran, Iran, e-mail: shiry@ce.aut.ac.ir

Mohammad Bagher Menhaj
Amirkabir University of Technology, 424 Hafez Ave., 15914 Tehran, Iran, e-mail: tmenhaj@ieee.org

crude data to reach more accurate knowledge about the position of a mobile robot. Nowadays, various algorithms are proposed to modify the accuracy of localization algorithms.

In the field of mobile robot localization, Kalman filter approach [1] plays a significant role. Kalman approach combines the odometry data and the measurements collected by the sensors of a mobile robot and provides an accurate estimation of the robot's position. Now, Kalman filter approach is greatly used in many localization problems such as mobile robot tracking [2] and Simultaneous Localization and Mapping [3]. Although Kalman approach has provided a general strategy to solve localization problems, it suffers from some weaknesses. The most significant problem of the Kalman method is its weakness to the state estimation of many real systems in the presence of severe nonlinearity. Actually, in general nonlinear systems, the posterior probability density function of the states given the measurements cannot be presented by a Gaussian density functions and, therefore, Kalman method results in unsatisfactory responses. Moreover, Kalman filter is very sensitive to the initial conditions of the states. Accordingly, Kalman methods do not provide accurate results in many common global localization problems.

More recently, particle filters have been introduced to estimate non-Gaussian, nonlinear dynamic processes [4]. The key idea of particle filters is to represent the state by sets of samples (or particles). The major advantage of this technique is that it can represent multi-modal state densities, a property which has been shown to increase the robustness of the underlying state estimation process. Moreover, particle filters are less sensitive to the initial values of the states than the Kalman method is. This problem is very considerable in applications in which the initial values of the states are not known such as global localization problems. The mentioned power of particle filters in the state estimation of nonlinear systems has caused them to be applied with great success to different state estimation problems including visual tracking [5], mobile robot localization [6], control and scheduling [7], target tracking [8] and dynamic probabilistic networks [9].

Multiple robot tracking is another attractive issue in the field of mobile robotics. This problem can be introduced as the tracking of mobile robots' position by another mobile robot nominated as reference robot. Although this problem appears similar to the common localization algorithms, the traditional approaches can not be used because the reference robot does not have access to the odometery data of each mobile robot used in localization algorithms to predict the future position of the robot. Moreover, using the particle filter to estimate the position of all robots, known as combined state space, is not tractable because the complexity of this approach grows exponentially with the number of objects being tracked.

To solve the problem of the lack of odometery data, various motion models have been proposed in which no knowledge about external inputs, such as odometery, is necessary. The simplest model proposed is the near constant velocity model. This model has been greatly used in many tracking applications such as aircraft tracking [10], people tracking [11] and some other target tracking applications [12]. Besides this model, some other structures have been proposed usable in some special applications. Near constant acceleration model is a modified version of the previous

model which can be applied to the tracking of maneuvering objects [13]. Also, interactive multiple model(IMM) filters represent another tool to track the movement of highly maneuvering objects where the former proposed approaches are not reliable [14].

Recently, many strategies have been proposed in the literature to address the difficulties associated with multi-target tracking. The key idea of the literature is to estimate each object's position separately. To do so, the measurements should be assigned to each target. Therefore, data association has been introduced to relate each target to its associated measurements. By combining data association concept with filtering and state estimation algorithms, the multiple target tracking issue can be dealt easier than applying state estimation procedure to the combined state space model. Among the various methods, Joint Probabilistic Data Association Filter (JPDAF) algorithm is taken into account as one of the most attractive approaches. The traditional JPDAF [15] uses Extended Kalman Filtering(EKF) joint with the data association to estimate the states of multiple objects. However, the performance of the algorithm degrades as the non-linearities become more severe. To enhance the performance of the JPDAF against the severe nonlinearity of dynamical and sensor models, recently, strategies have been proposed to combine the JPDAF with particle techniques known as Monte Carlo JPDAF, to accommodate general non-linear and non-Gaussian models [16]. The newer versions of the mentioned algorithm use an approach named as soft gating to reduce the computational cost which is one of the most serious problems in online tracking algorithms [16]. Besides the Monte Carlo JPDAF algorithm, some other methods have been proposed in the literature. In [16], a comprehensive survey has been presented about the most well-known algorithms applicable in the field of multiple target tracking.

The Monte Carlo JPDAF algorithm has been used in many applications. In [11], the algorithm presented as sample based JPDAF, has been applied for multiple people tracking problem. In [17], the JPDAF has been used for multi-target detection and tracking using laser scanner. But, the problem of tracking multiple robots by a reference robot has not been discussed in the literature. This issue is very important in many multiple robot problems such as robot soccer.

In this paper, the Monte Carlo JPDAF algorithm is considered for the multiple robot tracking problem. Section II describes the Monte Carlo JPDAF algorithm for multiple target tracking. The JPDAF for multiple robot tracking is the topic of section III. In this section, we will present how the motion of robots in a real environment can be described. To do so, near constant velocity and acceleration model are introduced to describe the movement of mobile robots. Then, it will be shown how a reference robot can track the motion of the other robots. In section IV, various simulation results are presented to support the performance of the algorithm. In this section, the maneuvering movement of the mobile robot is also considered. The results show the suitable performance of the JPDAF algorithm in tracking the motion of mobile robots in the simulated environment. Finally, section V concludes the paper.

2 The JPDAF Algorithm for Multiple Target Tracking

In this section, the JPDAF algorithm considered as the most successful and widly applied strategy for multi-target tracking under data association uncertainty is presented. The Monte Carlo version of the JPDAF algorithm uses common particle filter approach to estimate the posterior density function of the states given measurements $\left(P(x_t|y_{1:t})\right)$. Now, consider the problem of tracking of N objects. x_t^k denotes the state of these objects at time t where k=1,2,..,N is the the target number. Furthermore, the movement of each target can be considered by a general nonlinear state space model as follows:

$$\begin{aligned} x_{t+1} &= f(x_t) + v_t \\ y_t &= g(x_t) + w_t \end{aligned} \tag{1}$$

where v_t and w_t are the white noise with the covariance matrixes Q and R, respectively. Also, in the above equation, f and g are the general nonlinear functions representing the dynamic behavior of the target and the sensor model. The JPDAF algorithm recursively updates the marginal filtering distribution for each target $P(x_t^k|y_{1:t})$, k=1,2,..,N instead of computing the joint filtering distribution $P(X_t|y_{1:t})$, $X_t = x_t^1,...,x_t^N$. To compute the above distribution function, some points should be considered as follows:

- It is very important how to assign each target state to a measurement. Indeed, in each iteration the sensor provides a set of measurements. The source of these measurements can be the targets or the disturbances also known as clutters. Therefore, a special procedure is needed to assign each target to its associated measurement. This procedure is designated as Data Association considered as the key stage of the JPDAF algorithm which is described in the next sections.
- Because the JPDAF algorithm recursively updates the estimated states, a recursive solution should be applied to update the states in each sample time. Traditionally, Kalman filtering has been a strong tool for recursively estimating the states of the targets in multi-target tracking scenario. Recently, particle filters joint with the data association strategy have provided better estimations specially when the sensor model is nonlinear.

With regard to the above points, the following sections describe how particle filters paralleled with the data association concept can deal with the multi-target tracking problem.

2.1 The Particle Filter for Online State Estimation

Consider the problem of online state estimation as computing the posterior probability density function $P(x_t^k|y_{1:t})$. To provide a recursive formulation for computing the above density function the following stages are presented:

- Prediction stage: The prediction step is proceeded independently for each target as:

$$P(x_t^k|y_{1:t-1}) = \int_{x_{t-1}^k} P(x_t^k|x_{t-1}^k)P(x_{t-1}^k|y_{1:t-1})dx_{t-1}^k \tag{2}$$

- Update stage: The update step can be described as follows:

$$P(x_t^k|y_{1:t}) \propto P(y_t|x_t^k)P(x_t^k|y_{1:t-1}) \tag{3}$$

The particle filter estimates the probability distribution density function $P(x_t^k|y_{1:t-1})$ by sampling from a specific distribution function as follows:

$$P(x_t^k|y_{1:t-1}) = \sum_{i=1}^{N} \tilde{w}_t^i \delta(x_t - x_t^i) \tag{4}$$

Where i=1,2,...,N is the sample number, $\tilde{w}_t$ is the normalized importance weight and δ is the delta dirac function. In the above equation, the state x_t^i is sampled from the proposal density function $q(x_t^k|x_{t-1}^k, y_{1:t})$. By substituting the above equation in equation (2) and the fact that the states are drawn from the proposal function q, the recursive equation for the prediction step can be written as follows:

$$\alpha_t^i = \alpha_{t-1}^i \frac{P(x_t^k|x_{t-1}^{k,i})}{q(x_t^k|x_{t-1}^{k,i}, y_{1:t})} \tag{5}$$

Where $x_{t-1}^{k,i}$ is the i^{th} sample of x_{t-1}^k. Now, by using equation (3) the update stage can be expressed as the recursive adjustment of importance weights as follows:

$$w_t^i = \alpha_t^i P(y_t|x_t^k) \tag{6}$$

By repeating the above procedure at each time step the sequential importance sampling (SIS) for online state estimation is presented as follows:

The SIS Algorithm

1. For i=1: N initialize the states x_0^i, prediction weights α_0^i and importance weights w_0^i.
2. At each time step t proceed the following stages:
3. Sample the states from the proposal density function as follows:

$$x_t^i \sim (x_t | x_{t-1}^i, y_{1:t}) \tag{7}$$

4. Update prediction weights as equation (5).
5. Update importance weights as equation (6).
6. Normalize the importance weights as follows:

$$\tilde{w}_t^i = \frac{w_t^i}{\Sigma_{i=1}^N w_t^i} \tag{8}$$

7. Set t=t+1 and go to step 3.

Where, for simplicity, the index, k, has been eliminated. The main failure of the SIS algorithm is the degeneracy problem. That is, after a few iterations one of the normalized importance ratios tends to 1, while the remaining ratios tend to zero. This problem causes the variance of the importance weights to increase stochastically over time [18]. To avoid the degeneracy of the SIS algorithm, a selection (resampling) stage may be used to eliminate samples with low importance weights and multiply samples with high importance weights. There are many approaches to implement resampling stage [18]. Among them, the residual resampling provides a straightforward strategy to solve the degeneracy problem in the SIS algorithm. Using the residual resampling joint with the SIS algorithm can be presented as the SIR algorithm as follows:

The SIR Algorithm

1. For i=1: N initialize the states x_0^i, prediction weights α_0^i and importance weights w_0^i.
2. At each time step t Do the following stages:
3. Do the SIS algorithm to sample states x_t^i and compute normalized importance weights $\tilde{w}_t^i$.
4. Check the resampling criterion:

   - If $N_{eff} > thresh$ follow the SIS algorithm.

   Else

   - Implement the residual resampling stage to multiply/suppress x_t^i with high/low importance weights.
   - Set the new normalized importance weights as $\tilde{w}_t^i = \frac{1}{N}$.

5. Set t=t+1 and go to step 3.

In the above algorithm, N_{eff} is a criterion checking the degeneracy problem which can be written as:

$$N_{eff} = \frac{1}{\sum_{i=1}^N (\tilde{w}_t^i)^2} \tag{9}$$

In [18], a comprehensive discussion has been made on how one can implement the residual resampling stage.
Besides the SIR algorithm, some other approaches have been proposed in the literature to enhance the quality of the SIR algorithm such as Markov Chain Monte Carlo particle filters, Hybrid SIR and auxiliary particle filters [19]. Although these methods are more accurate than the common SIR algorithm, some other problems such as the computational cost are the most significant reasons that, in many real time applications such as online tracking, the traditional SIR algorithm is applied to recursive state estimation.

2.2 Data Association

In the last section, the SIR algorithm was presented for online state estimation. However, the major problem of the proposed algorithm is how one can compute the likelihood function $P(y_t|x_t^k)$. To do so, an association should be made between the measurements and the targets. Briefly, the association can be defined as target to measurement and measurement to target association:
Definition 1: We will denote a target to measurement association (T→M) by $\tilde{\lambda} = \{\tilde{r}, m_c, m_T\}$ where $\tilde{r} = \tilde{r}_1, .., \tilde{r}_K$ and $\tilde{r}_k$ is defined as follows:

$$\tilde{r}_k = \begin{cases} 0 \text{ If the } k^{th} \text{ target is undetected} \\ j \text{ If the } k^{th} \text{ target generates the } j^{th} \text{ measurement} \end{cases}$$

Where j=1,2,...,m and m is the number of measurements at each time step and k=1,2,..,K which K is the number of targets.
Definition 2: In a similar fashion, the measurement to target association (M→T) is defined as $\lambda = \{r, m_c, m_T\}$ where $r = r_1, r_2, ..., r_m$ and r_j is defined as follows:

$$r_j = \begin{cases} 0 \text{ If the } j^{th} \text{ measurement is not associated} \\ \quad \text{to any target} \\ k \text{ If the } j^{th} \text{ measurement is associated} \\ \quad \text{to the } k^{th} \text{ target} \end{cases}$$

In both above equations, m_T is the number of measurements due to the targets and m_c is the number of clutters.
It is very easy to show that both definitions are equivalent but the dimension of the target to measurement association is less than the measurement to target association. Therefore, in this paper, the target to measurement association is used.
Now, the likelihood function for each target can be written as:

$$P(y_t|x_t^k) = \beta^0 + \sum_{i=1}^{m} \beta^i P(y_t^i|x_t^k) \tag{10}$$

In the above equation, β^i is defined as the probability that the i^{th} measurement is assigned to the k^{th} target. Therefore, β^i is written as:

$$\beta^i = P(\tilde{r}_k = i|y_{1:t}) \tag{11}$$

Before presenting the computation of the above equation, the following definition is explained:

Definition 3: We define the set $\tilde{\Lambda}$ as all possible assignments which can be made between the measurements and targets. For example, consider a 3-target tracking problem. Assume that the following associations are recognized between the targets and measurements:

$r_1 = 1,2, r_2 = 3, r_3 = 0.$

Now, the set $\tilde{\Lambda}$ can be shown in Table 1.

Table 1 Simulation Parameters

Target 1	**Target 2**	**Target 3**
0	0	0
0	3	0
1	0	0
1	3	0
2	0	0
2	3	0

In Table 1, 0 means that the target has not been detected.

By using the above definition, equation (11) can be rewritten as:

$$P(\tilde{r}_k = i|y_{1:t}) = \Sigma_{\tilde{\lambda}_t \in \tilde{\Lambda}_t, \tilde{r}_k = j} P(\tilde{\lambda}_t|y_{1:t}) \tag{12}$$

The right side of the above equation is written as follows:

$$P(\tilde{\lambda}_t|y_{1:t}) = \frac{P(y_t|y_{1:t-1}, \tilde{\lambda}_t)P(y_{1:t-1}, \tilde{\lambda}_t)}{P(y_{1:t})} = P(y_t|y_{1:t-1}, \tilde{\lambda}_t)P(\tilde{\lambda}_t) \tag{13}$$

In the above equation, we have used the fact that the association vector $\tilde{\lambda}_t$ is not dependent on the history of measurements. Each of the density functions in the right side of the above equation can be presented as follows:

- $P(\tilde{\lambda}_t)$
  The above density function can be written as:

  $$P(\tilde{\lambda}_t) = P(\tilde{r}_t, m_c, m_T) = P(\tilde{r}_t|m_c, m_T)P(m_c)P(m_T) \tag{14}$$

  The computation of each density function in the right side of the above equation is straightforward and can be found in [16].

- $P(y_t|y_{1:t-1},\tilde{\lambda}_t)$
  Because the targets are considered independent, the above density function can be written as follows:

$$P(y_t|y_{1:t-1},\tilde{\lambda}_t) = (V_{max})^{-m_c}\Pi_{j\in\Gamma}P^{r_t^j}(y_t^j|y_{1:t-1}) \tag{15}$$

  Where V_{max} is the maximum volume which is in the line of sight of the sensors, Γ is the set of all valid measurement to target associations and $P^{r_t^j}(y_t|y_{1:t-1})$ is the predictive likelihood for the $(r_t^j)^{th}$ target. Now, consider $r_t^j = k$. The predictive likelihood for the k^{th} target can be formulated as follows:

$$P^k(y_t^j|y_{1:t-1}) = \int P(y_t^j|x_t^k)P(x_t^k|y_{1:t-1})dx_t^k \tag{16}$$

  Both density functions in the right side of the above equation are estimated using the samples drawn from the proposal density function. However, the main problem is how one can determine the association between the measurements (y_t) and the targets. To do so, the soft gating method is proposed as the following algorithm:

Soft Gating

1. Consider $x_t^{i,k}$, i=1,2,..,N, as the samples drawn from the proposal density function.
2. For j=1:m do the following steps for the j^{th} measurement:
3. For k=1:K do the following steps:
4. Compute μ_j^k as follows:

$$\mu_j^k = \sum_{i=1}^{N}\tilde{\alpha}_t^{i,k}g(x_t^{i,k}) \tag{17}$$

   Where g is the sensor model and $\tilde{\alpha}_t^k$ is the normalized weight as presented in the SIR algorithm.
5. Compute σ_j^k as follows:

$$\sigma_j^k = R + \sum_{i=1}^{N}\tilde{\alpha}_t^{i,k}\big(g(x_t^{i,k}) - \mu_j^k)(g(x_t^{i,k}) - \mu_j^k\big)^T \tag{18}$$

6. Compute the distance to the j^{th} target as follows:

$$d_j^2 = \frac{1}{2}(y_t^j - \mu_j^k)^T(\sigma_j^k)^{-1}(y_t^j - \mu_j^k) \tag{19}$$

7. If $d_j^2 < \varepsilon$, assign the j^{th} measurement to the k^{th} target.

It is easy to show that the predictive likelihood presented in equation (16) can be approximated as follows:

$$P^k(y_t^j|y_{1:t-1}) \simeq N(\mu^k, \sigma^k) \tag{20}$$

Where $N(\mu^k, \sigma^k)$ is a normal distribution with mean μ^k and the covariance matrix σ^k. By computing the predictive likelihood, the likelihood density function is easily estimated. In the next subsection, the JPDAF algorithm is presented for multi-target tracking.

2.3 The JPDAF Algorithm

The mathematical foundations of the JPDAF algorithm were discussed in the last sections. Now, we are ready to propose the full JPDAF algorithm for the problem of multiple target tracking. To do so, each target is characterized by the dynamical model introduced by equation (1). The JPDAF algorithm is presented as follows:

The JPDAF Algorithm

1. **Initialization:** initialize the states for each target as $x_0^{i,k}$ where i=1,2,..,N and k=1,2,...,K, the predictive importance weights $\alpha_0^{i,k}$ and importance weights $w_0^{i,k}$.
2. At each time step t proceed through the following stages:
3. For i=1:N do the following steps:
4. For k=1:K conduct the following procedures for each target:
5. Sample the new states from the proposal density function as follows:

$$x_t^{i,k} \sim q(x_t|x_{t-1}^{i,k}, y_{1:t}) \tag{21}$$

6. Update the predictive importance weights as follows:

$$\alpha_t^{i,k} = \alpha_{t-1}^{i,k} \frac{P(x_t^{i,k}|x_{t-1}^{i,k})}{q(x_t^{i,k}|x_{t-1}^{i,k}, y_{1:t})} \tag{22}$$

   Then, normalize the predictive importance weights.
7. Use the sampled states and new observations, y_t, to constitute the association vector for each target as $R^k = \{j|0 \leq j \leq m, y_j \rightarrow k\}$ where $(\rightarrow k)$ refers to the association between the k^{th} target and the j^{th} measurement. To do so, use the soft gating procedure described in the last subsection.
8. Constitute all of possible association for the targets and make the set, $\tilde{\Gamma}$, as described by definition 3.

9. Use equation (13) and compute β^l for each measurement where l=1,2,..,m and m is the number of measurements.
10. By using equation (10) compute the likelihood ratio for each target as $P(y_t|x_t^{i,k})$.
11. Compute the importance weights for each target as follows:

$$w_t^{i,k} = \alpha_t^{i,k} P(y_t|x_t^{i,k}) \tag{23}$$

Then, normalize the importance weights as follows:

$$\tilde{w}_t^{i,k} = \frac{w_t^{i,k}}{\Sigma_{i=1}^{N} w_t^{i,k}} \tag{24}$$

12. Proceed the resampling stage. To do so, implement the similar procedure described in the SIR algorithm. Afterwards, for each target the resampled states can be presented as follows:

$$\{\tilde{w}_t^{i,k}, x_t^{i,k}\} \rightarrow \{\frac{1}{N}, x_t^{m(i),k}\} \tag{25}$$

13. Set the time as t=t+1 and go to step 3.

The above algorithm can be used for multiple target tracking issue. In the next section, we show how the JPDAF algorithm can be used for multiple robot tracking. In addition, some well-known motion models are presented to track the motion of a mobile robot.

3 The JPDAF Algorithm for Multiple Robot Tracking

In the last section, the JPDAF algorithm was completely discussed. Now, we want to use the JPDAF algorithm to the problem of multiple robot tracking. To do so, consider a simple 2-wheel differential mobile robot whose dynamical model is represented as follows:

$$\begin{aligned} x_{t+1} &= x_t + \frac{\Delta s_r + \Delta s_l}{2} cos(\theta_t + \frac{\Delta s_r - \Delta s_l}{2b}) \\ y_{t+1} &= y_t + \frac{\Delta s_r + \Delta s_l}{2} sin(\theta_t + \frac{\Delta s_r - \Delta s_l}{2b}) \\ \theta_{t+1} &= \theta_t + \frac{\Delta s_r - \Delta s_l}{b} \end{aligned} \tag{26}$$

Where $[x_t, y_t]$ is the position of the robot, θ_t is the angle of robot's head, Δs_r and Δs_l are the distances traveled by each wheel, and b refers to the distance between two wheels of the robot. The above equation describes a simple model presenting the motion of a differential mobile robot. For a single mobile robot localization the most

straightforward way is to use this model and the data collected from the sensors set on the left and right wheels measuring Δs_r and Δs_l at each time step. But, the above method does not satisfy the suitable results because the data gathered from the sensors are dotted with the additive noise and, therefore, the estimated trajectory does not match the actual trajectory. To solve this problem, measurements obtained from the sensors are used to modify the estimated states. Therefore, Kalman and particle filters have been greatly applied to the problem of mobile robot localization [2, 6]. Now, suppose the case in which the position of other robots should be identified by a reference robot. In this situation, the dynamical model discussed previously is not applicable because the reference robot does not have access the internal sensors of the other robots such as the sensors measuring the movement of each wheel. Therefore, a motion model should be defined for the movement of each mobile robot. The simplest method is near constant velocity model presented as follows:

$$\begin{aligned} X_{t+1} &= AX_t + Bu_t \\ X_t &= [x_t, \dot{x}_t, y_t, \dot{y}_t]^T \end{aligned} \tag{27}$$

Where the system matrixes are defined as:

$$\begin{aligned} A &= \begin{pmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{pmatrix} \\ B &= \begin{pmatrix} \frac{T_s^2}{2} & 0 \\ T_s & 0 \\ 0 & \frac{T_s^2}{2} \\ 0 & T_s \end{pmatrix} \end{aligned} \tag{28}$$

Where T_s refers to the sample time. In the above equations, u_t is a white noise with zero mean and an arbitrary covariance matrix. Because the model is considered as a near constant velocity, the covariance of the additive noise can not be so large. Indeed, this model is suitable for the movement of the targets with a constant velocity which is common in many applications such as aircraft path tracking [5] and people tracking [11].

The movement of a mobile robot can be modeled by the above model in many conditions. However, in some special cases the robots' movement can not be characterized by a simple near constant velocity model. For example, in a robot soccer problem, the robots conduct a maneuvering movement to reach a special target ordinary to the ball. In these cases, the robot changes its orientation by varying the input forces imposed to the right and left wheels. Therefore, the motion trajectory of the robot is so complex that the elementary constant velocity does not result in satisfactory responses. To overcome the mentioned problem the variable velocity model is proposed. The key idea behind this model is using the robot's acceleration as another

state variable which should be estimated as well as the velocity and position of the robot. Therefore, the new state vector is defined as, $X_t = [x_t, \dot{x}_t, a_t^x, y_t, \dot{y}_t, a_t^y]$, where a_t^x and a_t^y are the robot's acceleration along with the x and y axis, respectively. Now, the near constant acceleration model can be written as what mentioned in equation (27), except for the system matrixes are defined as follows [13]:

$$A = \begin{pmatrix} 1 & 0 & T_s & 0 & a_1 & 0 \\ 0 & 1 & 0 & T_s & 0 & a_1 \\ 0 & 0 & 1 & 0 & a_2 & 0 \\ 0 & 0 & 0 & 1 & 0 & a_2 \\ 0 & 0 & 0 & 0 & exp(-T_s) & 0 \\ 0 & 0 & 0 & 0 & 0 & exp(-T_s) \end{pmatrix}$$

$$B = \begin{pmatrix} b_1 & 0 \\ 0 & b_1 \\ b_2 & 0 \\ 0 & b_2 \\ b_3 & 0 \\ 0 & b_3 \end{pmatrix} \tag{29}$$

Where the parameters of the above equation are defined as follows:

$$\begin{aligned} b_3 &= \frac{1}{c}\big(1 - exp(-cT_s)\big) \\ a_2 = b_3, b_2 &= \frac{1}{c}(T_s - a_2), a_1 = b_2 \\ b_1 &= \frac{1}{c}(\frac{T_s^2}{2} - a_1) \end{aligned} \tag{30}$$

In the above equation, c is a constant value. The above model can be used to track the motion trajectory of a maneuvering object, such as the movement of a mobile robot. Moreover, combining the results of the proposed models can enhance the performance of tracking. This idea can be presented as follows:

$$\tilde{X}_t = \alpha \tilde{X^a}_t + (1 - \alpha)\tilde{X^v}_t \tag{31}$$

Where $\tilde{X^a}_t$ and $\tilde{X^v}_t$ are the estimation of the near constant acceleration and near constant velocity model, respectively, and α is a number between 0 and 1.
Besides the above approaches, some other methods have been proposed to deal with tracking of maneuvering objects such as IMM filters [14]. But, these methods are applied to track the movement of the objects with sudden maneuvering targets which is common in aircrafts. However, in mobile robots scenario, the robot's motion is not so erratic for the above method to be necessary. Therefore, the near constant velocity model, near constant acceleration model or a combination of the proposed models can be considered as the most suitable structures which can be used to represent the

dynamical model of a mobile robot. Afterwards, the JPDAF algorithm can be easily applied to the multiple robot tracking problem.

4 Simulation Results

To evaluate the efficiency of the JPDAF algorithm, a 3-robot tracking scenario is considered. Fig.1 shows the initial position of the reference and target robots. To prepare the scenario and implement the simulations, the parameters are presented for the mobile robots structure and simulation environment as Table 2. Now, the following experiments are conducted to evaluate the performance of the JPDAF algorithm in various situations.

Table 2 Simulation Parameters

Parameters	**Description**
v_l	the robot's left wheel speed
v_r	the robot's right wheel speed
b	the distance between the robot's wheels
n_s	number of sensors placed on each robot
R_{max}	the maximum range of simulation environment
Q_s	the covariance matrix of the sensors' additive noise
t_s	Sample time for simulation running
t_{max}	Simulation maximum running time

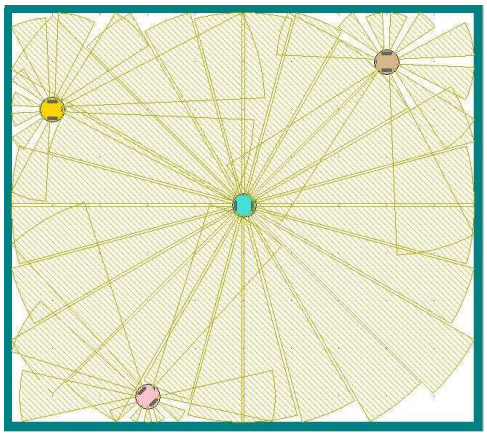

Fig. 1 Initial Position of Reference and Target Robots and Simulation Environment, non-maneuvering case

4.1 Local Tracking for Non-maneuvering Movement

For each mobile robot, the speed of each wheel is determined as shown in Table 3.

Table 3 Simulation Parameters

Robot Number	Left Wheel Speed	Right wheel Speed
1	$.2(\frac{Rad}{S})$	$.2(\frac{Rad}{S})$
2	$.3(\frac{Rad}{S})$	$.3(\frac{Rad}{S})$
3	$.1(\frac{Rad}{S})$	$.1(\frac{Rad}{S})$

The initial position of the reference robot is set as $[5,5,\frac{\Pi}{2}]$. Also, the position of the target robots are considered as $[8,2,\pi]$,$[1,3,\frac{\Pi}{3}]$ and $[1,9,\frac{\Pi}{4}]$, respectively. To run the simulation, sample time, t_s, and simulation maximum running time, t_{max}, are set as 1s and 200s, respectively. Fig. 2 shows the trajectories generated for each mobile robot. To probe the physical environment, 24 sensors are placed on the reference robot. The sensors provide the distance from the static objects, walls or obstacles, and the dynamic targets. Because the position of each sensor to the robot's head is fixed, the sensors' data can be represented as a pair of $[r,\phi]$ where ϕ is the angle between the sensor and the reference coordination. Also, the covariance of the sensors' noise is supposed to be:

$$\begin{pmatrix} .2 & 0 \\ 0 & 5\times 10E-4 \end{pmatrix} \tag{32}$$

Now, the JPDAF framework is used to track the trajectory of each robot. In this simulation, the clutters are assumed to be the data collected from the obstacles such as walls. To apply the near constant velocity model as described in equation (27), the initial value of the velocity is chosen uniformly between 1 and -1 as follows:

$$\dot{x}_t = \mathscr{U}(-1,1) \tag{33}$$

$$\dot{y}_t = \mathscr{U}(-1,1) \tag{34}$$

Where $\mathscr{U}(.)$ is the uniform distribution function. Fig. 3 presents the simulation results using the JPDAF algorithm with 500 particles for each robot's tracking. Simulations justify the ability of the JPDAF algorithm in modifying the intrinsic error embedded in the data collected from the measurement sensor.

4.2 Local Tracking for Maneuvering Movement

Now, consider a maneuvering movement for each robot. Therefore, a variable velocity model is considered for each robot's movement. Fig. 4 shows the generated trajectory for each mobile robot. Now, we apply the JPDAF algorithm for track-

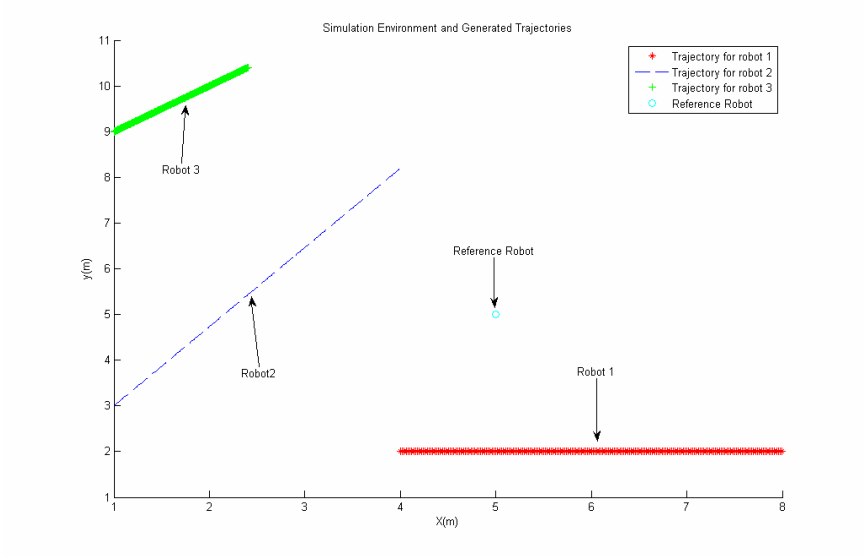

Fig. 2 The generated trajectory for target robots

ing each robot's position. To do so, 3 approaches are implemented. Near constant velocity model, near constant acceleration model and combined model described in the last section are used to estimate the states of the mobile robots. Fig. 5-7 shows the results for each robot separately. To compare the efficiency of each strategy, the obtained error is presented for each approach in Tables 4, 5 where the following criterion is used for estimating the error:

$$e_z = \frac{\sum_{t=1}^{t_{max}} (z_t - \tilde{z}_t)^2}{t_{max}}, z_t = \{x_t, y_t\} \tag{35}$$

The results show better performance of the combined method relative to the other approaches.

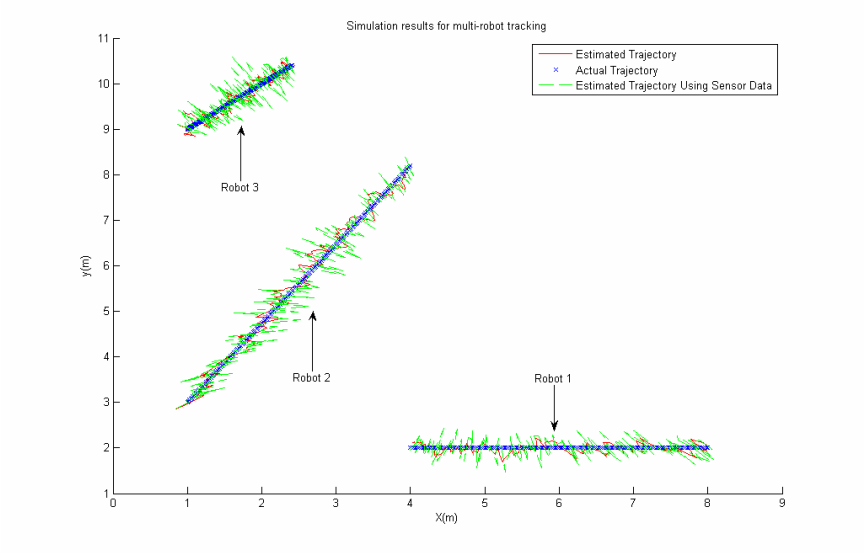

Fig. 3 The estimated trajectory using the JPDAF algorithm

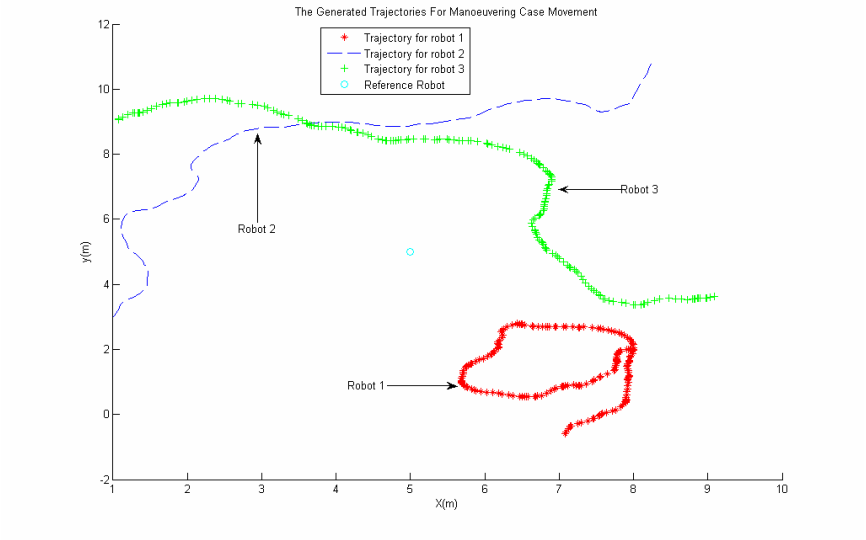

Fig. 4 The generated trajectory for manoeuvering target robots

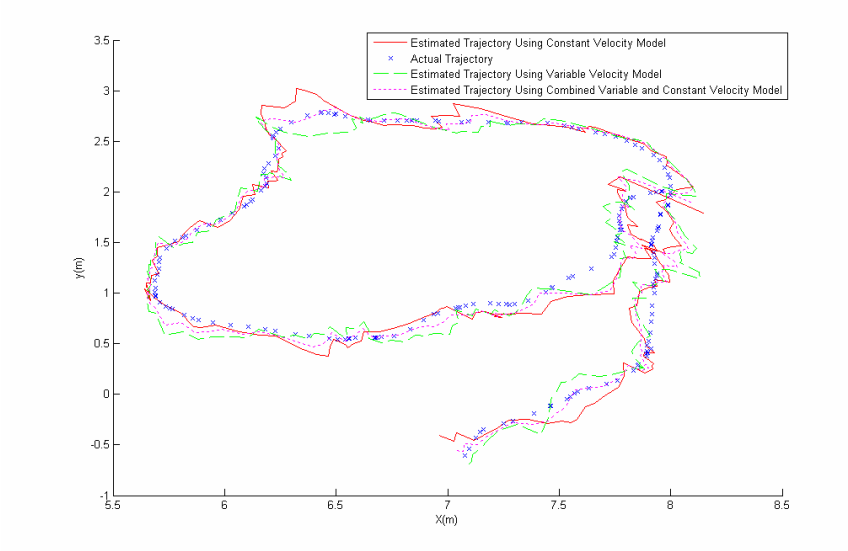

Fig. 5 The estimated trajectory for robot 1

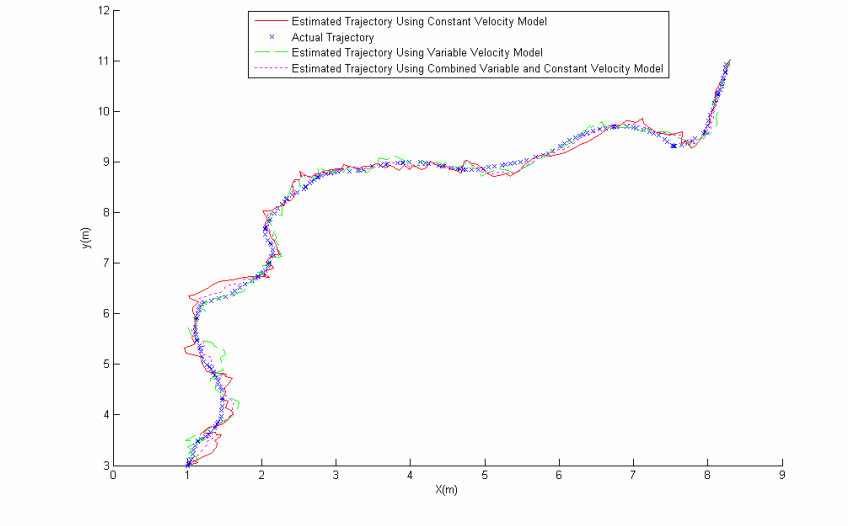

Fig. 6 The estimated trajectory for robot 2

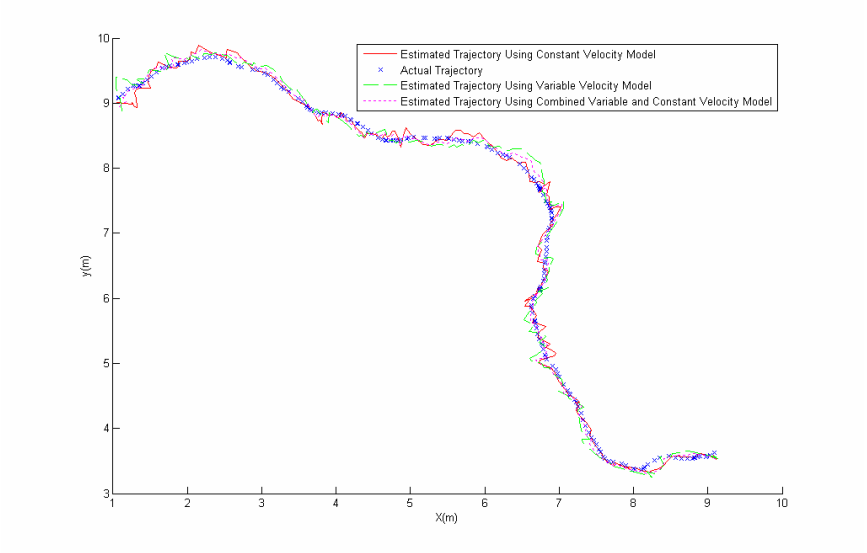

Fig. 7 The estimated trajectory for robot 3

Table 4 The error for estimating x_t

Robot Number	Constant Acceleration	Constant Velocity	Combined
1	.0045	.0046	.0032
2	.0112	.0082	.005
3	.0046	.0067	.0032

Table 5 The error for estimating y_t

Robot Number	Constant Acceleration	Constant Velocity	Combined
1	.0082	.009	.0045
2	.0096	.0052	.005
3	.0073	.0077	.0042

5 Conclusion

In this paper, the JPDAF algorithm was presented for multi robot tracking. After introducing the mathematical basis of the JPDAF algorithm, we showed how one can apply the JPDAF algorithm to the problem of multiple robot tracking. To do so, three different models were represented. The near constant velocity model and near constant acceleration model were first described and, then, combining two former approaches were presented as a modification to enhance the accuracy of multi robot tracking. Afterwards, simulation results were carried out in two different stages.

First, tracking the non-maneuvering motion of the robots were explored. Later, the JPDAF algorithm was tested on tracking the motion of the robots with maneuvering movement. The simulations show that the combined approach provided better performance than the other presented strategies. Although the simulations were explored in the case in which the reference robot was assumed to be stationary, this algorithm can be easily extended to multi robot tracking as well as mobile reference robot. Indeed, robot's self localization can be used to obtain the mobile robot's position. Next, the JPDAF algorithm is easily applied to the target robots according to the current estimated position of the reference robot.

References

1. R. E. Kalman, and R. S. Bucy, New Results in Linear Filtering and Prediction, Trans. American Society of Mechanical Engineers, Series D, Journal of Basic Engineering, Vol. 83D, pp. 95–108, 1961.
2. R. Siegwart and I. R. Nourbakhsh, Intoduction to Autonomous Mobile Robots, MIT Press, 2004.
3. A. Howard, Multi- robot Simultaneous Localization and Mapping using Particle Filters, Robotics: Science and Systems I, pp. 201–208, 2005.
4. N.J. Gordon, D.J. Salmond, and A.F.M. Smith, A Novel Approach to Nonlinear/non-Gaussian Bayesian State Estimation, IEE Proceedings F, 140(2):107–113, 1993.
5. M. Isard, and A. Blake, Contour Tracking by Stochastic Propagation of Conditional Density, In Proc. of the European Conference of Computer Vision, 1996.
6. D. Fox, S. Thrun, F. Dellaert, and W. Burgard, Particle Filters for Mobile Robot Localization, Sequential Monte Carlo Methods in Practice. Springer Verlag, New York, 2000.
7. C. Hue, J. P. L. Cadre, and P. Perez, Sequential Monte Carlo Methods for Multiple Target Tracking and Data Fusion, IEEE Transactions on Signal Processing, Vol. 50, NO. 2, February 2002.
8. B. Ristic, S. Arulampalam, and N. Gordon, Beyond the Kalman Filter, Artech House, 2004.
9. K. Kanazawa, D. Koller, and S.J. Russell, Stochastic Simulation Algorithms for Dynamic Probabilistic Networks, In Proc. of the 11th Annual Conference on Uncertainty in AI (UAI), Montreal, Canada, 1995.
10. F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. J. Nordlund, Particle Filters for Positioning, Navigation, and Tracking, IEEE Transactions on Signal Processing, Vol. 50, No. 2, February 2002
11. D. Schulz, W. Burgard, D. Fox, and A. B. Cremers, People Tracking with a Mobile Robot Using Sample-based Joint Probabilistic Data Association Filters, International Journal of Robotics Research (IJRR), 22(2), 2003.
12. M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking, IEEE Transactions on Signal Processing, Vol. 50, No. 2, February 2002.
13. N. Ikoma, N. Ichimura, T. Higuchi, and H. Maeda, Particle Filter Based Method for Maneuvering Target Tracking, IEEE International Workshop on Intelligent Signal Processing, Budapest, Hungary, May 24-25, pp.3–8, 2001.
14. R. R. Pitre, V. P. Jilkov, and X. R. Li, A comparative study of multiple-model algorithms for maneuvering target tracking, Proc. 2005 SPIE Conf. Signal Processing, Sensor Fusion, and Target Recognition XIV, Orlando, FL, March 2005.
15. T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association, IEEE Journal of Oceanic Engineering, Vol. 8, pp. 173–184, 1983.

16. J. Vermaak, S. J. Godsill, and P. Perez, Monte Carlo Filtering for Multi-Target Tracking and Data Association, IEEE Transactions on Aerospace and Electronic Systems, Vol. 41, No. 1, pp. 309–332, January 2005.
17. O. Frank, J. Nieto, J. Guivant, and S. Scheding, Multiple Target Tracking Using Sequential Monte Carlo Methods and Statistical Data Association, Proceedings of the 2003 IEEE/IRSJ, International Conference on Intelligent Robots and Systems, Las Vegas, Nevada, October 2003.
18. Jao. F. G. de Freitas, Bayesian Methods for Neural Networks, PhD thesis, Trinity College, University of Cambridge, 1999.
19. P. D. Moral, A. Doucet, and A. Jasra, Sequential Monte Carlo Samplers, J. R. Statist. Soc. B, 68, Part 3, pp. 411–436, 2006.