# Multi-Robot Cooperative Object Tracking Based on Particle Filters

Aamir Ahmad* Pedro Lima*
*Institute for Systems and Robotics, Instituto Superior Técnico, Lisboa, Portugal

***Abstract*— This article presents a cooperative approach for tracking a moving object by a team of mobile robots equipped with sensors, in a highly dynamic environment. The tracker's core is a particle filter, modified to handle, within a single unified framework, the problem of complete or partial occlusion for some of the involved mobile sensors, as well as inconsistent estimates in the global frame among sensors, due to observation errors and/or self-localization uncertainty. We present results supporting our approach by applying it to a team of real soccer robots tracking a soccer ball.**

## I. Introduction and Related Work

This paper deals with cooperative tracking of an object by a team of robots. Object tracking is a field of research with multiple techniques being researched and developed extensively [1]. In recent years RoboCup Soccer has laid down a common platform for various research areas in robotics, object tracking being a predominant and crucial one. This involves tracking the soccer ball by the robots during the game play. The complexity of tracking has risen from small, orange colored balls to standard sized, random/multi-colored balls and from 2D to 3D [2]. The problem can be formulated as tracking a moving object of known dimensions by a moving robot. We use RoboCup Soccer as an ideal testbed for novel methods that can be used outside soccer.

Particle Filters (PF) are one of the most popular methods employed for tracking [7]. PF is a non-parametric filter. Non-parametric filters can efficiently handle multi-modal beliefs. In a generic tracker, the motion model of object being tracked can be completely unknown and might change over time hence using a parametric filter can lead to failures quite often. This is because if we use any standard motion model for the object, the tracker can quickly result in low confidence on the posterior when the object motion changes to a different model or switches randomly. This makes it essential to have beliefs with multiple modes scattered over the whole state space which makes the use of a non-parametric filter appropriate. In the RoboCup scenario, PF based trackers are dominant tools currently being used by most of the teams. An interesting approach of fusing the Extended Kalman Filter (EKF) and Monte Carlo PF has been described in [4] where an integrated self-localization and ball tracking method is presented. In [5] a method for simultaneously estimating ball position and velocity using Monte Carlo Localization (MCL) is developed. An efficient implementation of Rao-Blackwellised PF which was successfully demonstrated on Sony AIBO robots in the four-legged league of RoboCup is presented in [7]. None of these works use the information from more than one sensor/robot, therefore being less robust to occlusion and very dependent on the relative state of the robot and the object tracked.

The field of sensor fusion, including its use for single and multiple target tracking [12,13], is now very mature. However, it does frequently address situations where the sensors are static, know their location in a global frame with no uncertainty, and occlusions occur rarely. When sensors are mobile, e.g., mounted on the top of mobile robots, their knowledge of their own localization may degrade over time and/or during time periods due to a number of reasons (e.g., absence of known environment features, bad odometry) and this impacts the uncertainty in the determination of the target position in the global frame, where it is fused with the estimates from the other sensors. Furthermore, occlusions can occur more frequently, as they are due not only to the target(s) path but also to the motion of the different sensors/robots. Therefore, the problem of cooperatively tracking a moving object by a team of mobile sensors is an extension of sensor fusion, designated here as cooperative perception, in which one has to handle occlusions, disagreements between sensors, and dynamic changes of the observation models due to frequent spatial changes.

Efficient solutions for multiple static platforms and a moving target [14] or a single moving platform and moving target(s) [15] have been introduced. Our approach to combine both challenges, i.e., track a moving target using multiple moving platforms, is novel.

In [10] relationship between fixed world objects and moving objects is explored for global object localization. These relationships are communicated to teammates where they form a set of constrained relations, solving which gives object location estimates. The authors in [6] present a cooperative PF based tracker for Sony AIBO robots, where the fusion of information involves communicating a reduced set of particles between the robots over the wireless network, which still remains a huge data set causing inefficient communication. Our approach overcomes this problem which is explained further in this paper. Outside the RoboCup scenario, in [11] a new cooperative perception architecture is developed and tested on multiple UAVs for forest fire detection and localization. A substantial effort is put on developing the fire detector and fusion of data from various sensors used on-board a single and multiple UAVs. The errors that creep in due to the self-localization of the UAVs themselves are unaccounted for, which we address in our work.

In [12,13] a decentralized PF for multiple target tracking is developed and deployed on flight vehicles. The communication bandwidth problem is solved by transforming the particle

This work was supported by FCT (ISR/IST plurianual funding) through the PIDDAC Program funds, and by project FCT PTDC/EEA-CRO/100692/2008 PCMMC.

set into a Gaussian Mixture Model (GMM) which seems to be an efficient way. In our work we communicate a single parametrized observation probability density function between two robots. This not only further reduces the bandwidth usage but also prevents the prediction model errors of the PF to be propagated to team-mates which happens when sharing of particles (or of a parametrized form of it) is done.

Our work builds mainly upon [2] and [3], carried out in the direction of object tracking and sensor fusion among teammates respectively. In [2], a PF based tracker is presented with a unique and novel 3-D observation model based on color histogram matching. Each robot has an individual tracker and its most notable feature is that the tracking could be performed in 3-D space without the object color information, but at the cost of computational expense. In [3] a sensor fusion technique for cooperative object localization using particle filters is presented. Parameters of a GMM approximating a teammate's tracker's particles are communicated to the other robots. Particles at a robot's tracker are then sampled using own belief and the received GMM.

In this paper we introduce an approach to cooperative perception where we implement a PF-based tracker. For each observing robot (i.e., a mobile robot with a sensor), we determine confidence factors associated to the tracked target from two origins: i) the confidence on the observation itself and ii) the confidence on the self-localization estimate of the observing robot. Note that the self-localization method itself is completely decoupled from the PF-based tracker. The observation model of each mobile sensor is a parametrized probability density function (e.g., a Gaussian centered on the observation). The probability density functions associated to the observations of the team robots are shared by all of them in a pool. Each robot selects the best function, i.e., the one with higher confidence factors, from the pool, and uses it to assign weights to the particles in the traditional PF update step. The parametrization of the observation models intends to reduce significantly the amount of data communicated to teammates, since the probability density function can be univocally represented by its communicated parameters. The method handles, within a single unified framework, inconsistencies (disagreements) between sensors due to observation errors and/or self-localization uncertainty. In order to achieve near real-time tracking, we track the object in 2-D space only and use the object color information. These will be relaxed in the future work, as they depend mostly on the available computing power.

## II. Particle Filter based Tracker

In this section we briefly explain a standard PF and how it is used to construct a tracker. Our tracker will estimate the 2-D pose of an object assumed to be moving on a known ground plane. The state vector of the object will be denoted by $\mathbf{x}_t$. We assume that the object's state evolution over time is an unobserved Markov process with a uniform initial distribution $p(\mathbf{x}_0)$ and a transition distribution $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. The observations $\{\mathbf{y}_t; t \in \mathbb{N}\}$ are conditionally independent given the process $\{\mathbf{x}_t; t \in \mathbb{N}\}$ with distribution $p(\mathbf{y}_t|\mathbf{x}_t)$.

For the estimation of the *a posteriori* distribution *i.e.* belief of the state given all observations $p(\mathbf{x}_t|\mathbf{y}_{1:t})$, the problem under Markov assumption is formulated as:

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\mathbf{x}_t) \int p(\mathbf{x}_t|\mathbf{x}_{t-1}, u_t) p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) d\mathbf{x}_{t-1} \tag{1}$$

In the rest of paper we refer to $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ as the motion model and to $p(\mathbf{y}_t|\mathbf{x}_t)$ as the observation model. $u_t$ is the odometry input to the motion model.

For solving the problem as formulated above we use a PF-based tracker. A PF is a non-parametric Bayesian filter where, contrary to a parametric Bayesian filter, the state's probability distribution is represented by a set of M weighted particles $\mathbf{X}_t \triangleq \{\mathbf{x}_t^i, w_t^i\}_{i=1}^M$ where each particle is an instantiation of the state itself [8].

## III. The Cooperative Tracker

In this section we present our PF-based cooperative tracker algorithm (see Algorithm 1) which involves the classical PF augmented with the fusion step which we introduce as a novel contribution.

**Algorithm 1** PF_Cooperative_Tracker($\mathbf{X}_{t-1}, u_t, M, k, N$)

$\bar{\mathbf{X}}_t = \mathbf{X}_t = \phi$
**for** $m = 1$ to M **do**
  $x_t^{[m]} =$ ***sample_motion_model***$(u_t, x_{t-1}^{[m]})$
**end for**
{The Fusion step starts here}
Perform sensor observation and generate Observation Model $\mathcal{M}_{klocal}$
$\mathcal{M}_{klocal} \rightarrow \mathcal{M}_{kworld}$ {Frame Transformation using self posture. In rest of the algorithm the Observation models' parameters are in world frame and are denoted by $\mathcal{M}_k$ }
Compute and Send $\mathcal{M}_k, \mathcal{C}(\mathcal{M}_k), \mathcal{C}_{LOCk}$ to teammates.
$\alpha_k = \mathcal{C}(\mathcal{M}_k)$ {Observation confidence for robot $k$}
**for** $r = 1$ to $N$ **do**
  **if** $r \neq k$ **then**
    receive $\mathcal{M}_r, \mathcal{C}(\mathcal{M}_r), \mathcal{C}_{LOCr}$ from $R_r$
    $\alpha_r = \mathcal{C}(\mathcal{M}_r) * \mathcal{C}_{LOCr}$
  **end if**
**end for**
**for** $r = 1$ to $N$ **do**
  $\alpha_r = \frac{\alpha_r}{\sum_{r=1}^{N} \alpha_r}$ {Weight normalization step}
**end for**
**for** $m = 1$ to M **do**
  $i = r$ ; $r \in [1:N]$ *sampled with probability* $\alpha_r$
  $w_t^{[m]} \sim \mathcal{M}_i$
  $\bar{\mathbf{X}}_t = \bar{\mathbf{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
**end for**
{The Fusion step ends here}
$\mathbf{X}_t$ = Low_Variance_Sampler($\bar{\mathbf{X}}_t$)
return $\mathbf{X}_t$

The prediction step and the resampling step of the PF based cooperative tracker inherit directly from the original PF [8]. We introduce a *fusion step* which modifies the observation

model's mechanism which in turn modifies the way in which the particles are assigned weights.

The first input to the algorithm, $\mathbf{X}_{t-1}$ is the set of particles, which initially could be distributed on the state space according to any choice. Here we consider a uniform distribution. $u_t$ denotes the input to the motion model of the tracked object. $M$ is the total number of particles which depends on the available computational resource. The larger the value of $M$, the better is the approximation of the target probability density function (PDF). A standard practice is to keep $M$ close to the order of $500 - 1000$. $k$ is the robot number on which the algorithm is running in a team of $N$ robots. We denote the $r^{th}$ robot as $R_r$.

First, we assume that each robot can communicate with every other robot in the team. After the prediction step of PF the robot makes an observation and generates an observation PDF, denoted by $\mathcal{M}$, approximating the observation over the whole state space. This PDF could be any parametrized function in general based on the model chosen. $\mathcal{M}_k$ is the observation model PDF by robot $k$, which can be reduced to the parameters of the PDF representing the observation model, e.g., mean and covariance matrix for the Gaussian case. $\mathcal{C}(\mathcal{M}_k)$ represents the robot $k$'s confidence on its observation which can be calculated in various ways depending on the implementation and scenario. We calculate $\mathcal{C}(\mathcal{M}_k)$ as :

$$\mathcal{C}(\mathcal{M}_k) = \frac{\mathcal{A}_{obs}}{\mathcal{A}_{exp}}, \tag{2}$$

where $\mathcal{A}_{obs}$ is the area of the tracked object observed in the camera image and $\mathcal{A}_{exp}$ is the expected area of the object since we know *a priori* the real dimensions of the tracked object.

$\mathcal{C}_{LOCk}$ represents the robot $k$'s confidence on its own localization. The self-localization confidence factor is determined from the particle filter set. One good approach to do this is to consider the *number of effective particles* $n^k_{eff}$ [8]

$$\mathcal{C}_{LOCk} = n^k_{eff} = \frac{1}{\sum_{m=1}^{M}(w_t^{[m]})^2}, \tag{3}$$

which assumes normalized weights of the particles.

The observation PDF's parameters are converted into the world frame at this point, using the robot's self-localization estimate. Next, we receive $\mathcal{M}_r, \mathcal{C}(\mathcal{M}_r), \mathcal{C}_{LOCr}$ from every other robot $r$ in the team. It is important to note here that these parameters obtained from the teammate robots are already expressed in the world frame. This leads us to form a set $\mathbf{P}_k = \{\mathcal{M}_r \mid 1 \leqslant r \leqslant N\}$ which we call an observation model pool (OMP) for the robot $k$ in the world frame whose elements represent each robot's individual observation PDF.

We now associate a weight to each element in $\mathbf{P}_k$ as mentioned in Algorithm 1. For the robot $k$'s OMP, only the elements due to other teammates are weighed using their self-localization confidence except the element due to $k$'s own observation which is weighed only by its observation confidence. The reason for this will become clear later.

$$\alpha_k = \mathcal{C}(\mathcal{M}_k) \tag{4}$$

$$\alpha_r = \mathcal{C}(\mathcal{M}_r) * \mathcal{C}_{LOCr} \quad 1 \leqslant r \leqslant N; r \neq k \tag{5}$$

Furthermore, we normalize the weights to :

$$\alpha_r = \frac{\alpha_r}{\sum_{r=1}^{N}\alpha_r} \quad 1 \leqslant r \leqslant N \tag{6}$$

The most crucial step of fusion is here. For every particle that we have after the prediction step we do the following :

*Step 1* : Sample an element $\mathcal{M}_i$ from the OMP $\mathbf{P}_k$ with probability $\alpha_i$. Recall that $\mathcal{M}_i$ is the parametrized observation PDF generated by robot $i$'s observation of the target.

*Step 2* : Calculate the weight of the particle using the PDF represented by $\mathcal{M}_i$ sampled in the previous step. For instance if $\mathcal{M}_i$ is parametrized as a Gaussian over the target space, the closer the particle lies to the mean of this Gaussian, the higher is its weight. These two steps are further clarified in Fig. 1 with the help of an example OMP.

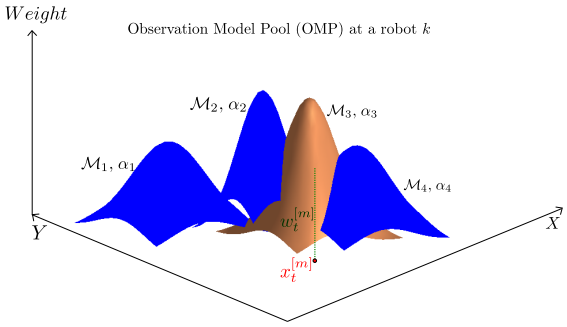


Fig. 1. For an $m^{th}$ particle $x_t^{[m]}$ at time step $t$, $\mathcal{M}_3$ is sampled with a probability $\alpha_3$ from the OMP which consists of the observation model PDFs $\mathcal{M}_1 \cdots \mathcal{M}_4$ which in this figure are shown as Gaussians for the sake of an example (but in practice it can be any parametrized PDF). $\mathcal{M}_3$ is then used to generate the weight $w_t^{[m]}$ of the particle $x_t^{[m]}$. Sampling $\mathcal{M}$ and then using it for generating the particle's weight is done for every particle at each time step.

Since the above two steps are performed for every particle at a given iteration of the Algorithm 1, it fuses the information from all the elements of the OMP according to their respective importance. After this, resampling is performed which can be done by any established method. The association of OMP elements to particles is done in every iteration of the PF-based tracker. Due to the sampling process in step 1 as mentioned above a few particles get associated with low weight elements of OMP to maintain the spread of the particles.

The most relevant problem which this PF-based cooperative tracker solves is that of partial or complete occlusion. A partially occluded object's observation by a robot $k$ may lead to a low confidence in its own observation PDF, but in case the same object is being fully observed by another teammate robot $r$, it will lead to a high confidence in the element contributed by it to $\mathbf{P}_k$, and hence a greater chunk of particles will get associated to it. This way the robot $k$ would still be able to make a good approximation of the target distribution. The OMP elements refer to the world frame. If a robot is wrongly localized, its observation of the object in the world frame will be incoherent with another correctly localized robot's observation of the same object in the world frame, although both might be observing the object correctly in their respective local frames. The incoherency here is due to the frame transformation carried out by the wrongly

localized robot, of its observation in local frame to the global frame. In our approach this problem is solved by weighing the OMP elements by the associated robot's self localization confidence multiplied by its observation confidence. Also, this is done differently for the recipient robot and the sender robots (4),(5). By doing this we ensure 2 major advantages: i) a wrongly localized robot's good observation hardly influences other robot's OMP; ii) a wrongly localized robot would still have a high confidence in its own good observation which is necessary if we want to carry out visual tracking in its local frame. It will not be affected by the observations of well localized teammates which are incoherent with the wrongly localized robot's local frame. This enables the robot to keep tracking the object with its local information during visual tracking without relying on incorrect global frame information.

We do not have to deal with the robot's motion directly. This is taken care *in situ* when we construct the OMP in world frame, using the self-posture of the robot which inherently involves odometry or motion update of the robot, assuming that the robot's acceleration is not very high. Otherwise the frame transformation functions should include the non-inertial terms.

Robot motion control is handled assuming that the robot's acceleration is not very high. Otherwise the frame transformation functions should include the non-inertial terms.

## IV. Implementation and Results

### A. Test Bed

We applied the proposed algorithm here to the robot soccer scenario. Our testbed is the RoboCup Middle Sized League(MSL) robot team. In robot soccer, one of the basic necessities is to continuously track the ball. A major concern here is that the robots tend to lose their localization on the soccer field because of low-range vision and field symmetry. Moreover, since the field is too large as compared to a robot's camera vision field, a ball far from the robot ($> 5m$) is scarcely visible, and very often a nearby lying ball is occluded by a teammate or an opponent robot. Thus it becomes a very interesting and appropriate testbed for our proposed algorithm.

Fig. 2. Omni directional robots of MSL on the soccer field

Each of our 5 fully autonomous robots (Fig. 2) in the team is characterized by an omni directional drive and a dioptric vision system consisting of a fish-eye lens facing downwards. All high level algorithms are coded using C/C++ programming language and run on a Pentium IV 1.6 Ghz CPU laptop always mounted on the robot itself. All implementations and results presented further involve tracking the ball by 4 of these robots in one half of the MSL field.

### B. Implementation

In our implementation, we model individual robot's observation of the ball as a bivariate Gaussian distribution over the soccer field. The confidence of the observation is calculated based on the observed ball size and expected ball size as mentioned in (2). This is because in the image frame the expected ball size is a fixed function of the distance of ball (*fixed radius* $= 10cm$) from the image center. Images from the camera are streamed at 15 fps.

For the prediction step of the tracker we use the same approach (7) as in [2]

$$\mathbf{X}_{t+1} = \begin{bmatrix} \mathbf{I} & (\Delta t)\mathbf{I} \\ 0 & \mathbf{I} \end{bmatrix} \mathbf{X}_t + \begin{bmatrix} (\frac{\Delta t^2}{2})\mathbf{I} \\ (\Delta t)\mathbf{I} \end{bmatrix} a_t, \tag{7}$$

which is a constant velocity model with normally distributed acceleration noise about zero mean. $I$ is a $3\times3$ identity matrix, $\Delta t = 1$, and $a_t$ is a $3 \times 1$ white zero mean random vector corresponding to an acceleration disturbance. For resampling we apply the Low Variance Sampling method [8].

### C. Robot Localization

Since the PF based tracker is closely interlaced with the localization uncertainties of individual robots, it is worth mentioning that the localization method implemented here is Monte Carlo Localization, and is independent of the ball tracking particle filter.

### D. Results

In this sub-section we present a set of plots for each experiment we made using 4 robots represented as $R_1 - R_4$. In Figs. 3 and 5 robot's self localization confidence and individual robot's ball observation confidence is shown. The plots in Figs. 4 and 6 represent the spatial variance and the variance of weights of the particles at every iteration step for the tracker running on $R_3$ in each experiment which are calculated as in (8) and (9). Every iteration of the PF-based tracker consisting of the prediction step and the observation-fusion-update step takes $\sim 0.1$ second. Note that comparing with the ground truth is rather difficult in scenarios like this because an overhead field camera will have tracking errors in itself.

$$\sigma^2_{\text{spatial}} = \frac{1}{M} \sum_{m=1}^{M} \|\mathbf{x}_t^{[m]} - \bar{\mathbf{x}}_t\|^2, \tag{8}$$

$$\sigma^2_{\text{weight}} = \frac{1}{M} \left( \sum_{m=1}^{M} \left(w_t^{[m]}\right)^2 - \frac{1}{M} \sum_{m=1}^{M} \left(w_t^{[m]}\right)^2 \right). \tag{9}$$

In the accompanying video [http://www.youtube.com/watch?v=m_a4DuWagYI], the left side of the frame shows the self localization estimates of all robots and $R_3$'s estimate of the tracked ball's global position. The right side of the frame shows the simultaneously shot overhead footage of the experiment in which $R_3$ can be seen reaching for the ball.

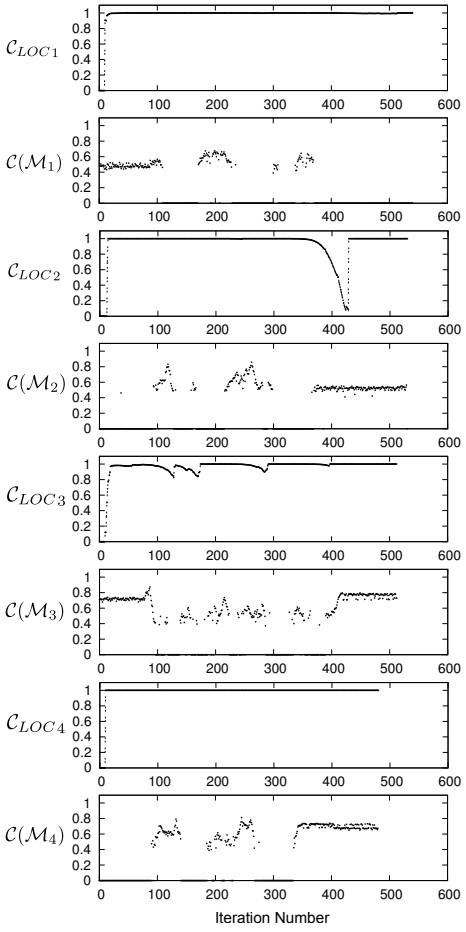


Fig. 3. Confidence plots of experiment 1 (Each iteration is performed every 0.1 second)

*1) Experiment 1:* In this experiment we have 3 robots which are forced to stay at their initial positions on the field and $R_3$ is supposed to reach for the ball. Tracking by $R_3$ has been analyzed in the plots of Fig. 4.

Confidence plots for $R_3$ in Fig. 3 show that intermittently between iterations 100 and 400 the ball was not directly observed by it. During this period, the ball was directly observed by either $R_1$, $R_2$ or $R_4$ (see other plots in Fig. 3). As a result of fusion of the observation models from teammates as explained previously in this paper, the cooperative tracker on $R_3$ was able to track the ball consistently during this period with only a few breaks where it momentarily lost the ball. This is supported by the low spatial variance of the $R_3$'s tracker particles and their converged weight variance in Fig. 4 with few small periods of peaks in variances denoting instances when the ball was completely lost. Also, the robot kept following the correct ball visible in the video accompanying this paper.

The spatial variance of the particles in $R_3$'s tracker reach a high peak for a small duration after iteration 300 in Fig. 4 which indicates that the ball was not tracked during that period. This was not only because the ball was lost from $R_2$, $R_3$ and $R_4$'s vision field but also only $R_1$ was observing the ball with a low observation confidence ($\sim 0.4$). During most part of this experiment all four robots stay well localized.

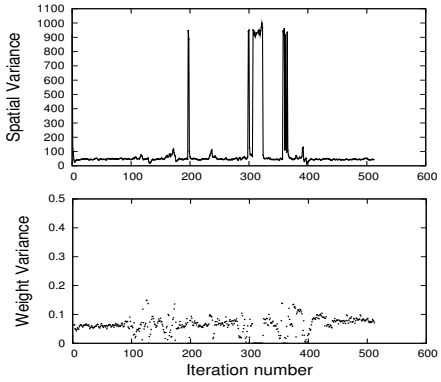


Fig. 4. Spatial Variance and Variance of the weights of the Particles of the cooperative tracker at $R_3$ in experiment 1

*2) Experiment 2:* Similar to experiment 1, in this setup too there are 3 static robots ($R_1, R_2, R_4$) and one dynamic robot ($R_3$). The major difference here is that $R_1$, $R_2$ and $R_3$ do not localize well for most of the time.

The first visible conclusion from the confidence plots in Fig. 5 is that the robot $R_3$ directly observes the ball for a very short period (iterations $\sim 300 - 350$, near 450 and near 700). Up to iteration 300, the spatial variance of $R_3$'s tracker's particles remain low and their weight variance has also converged indicating that $R_3$ was able to track the ball, which is supported by the accompanying video. This is because $R_4$ (see Fig. 5) was directly observing the ball and since it was well localized, $R_3$ was relying on its observation model and hence tracking the ball in the correct position.

Close to iteration 300 the ball is lost by all the robots for a very short instance (peak in spatial variance in Fig. 6) and then $R_3$ and $R_4$ observe the ball alternatively between iterations ($\sim 300$ & $420$) (Fig. 5) leading to a converged spatial variance during that time supported by the video showing that $R_3$ is consistently following the ball.

During iterations ($\sim 420 - 650$) the ball is seen directly only by $R_1$ (Fig. 5) which has an almost zero confidence on its self localization estimate (Fig. 5). The cooperative tracker at $R_3$ takes this into account and prevents itself from tracking the ball in a wrong position, hence the high spatial variance in Fig. 6 during that time period.

Another notable instance is at iteration 400 where $R_4$ was directly observing the ball and was well localized (Fig. 5), $R_1$ and $R_3$ were not seeing the ball and $R_2$ was observing the ball but was not localized (low self-localization confidence in Fig. 5). At that instance, $R_3$ was following the correct ball (supported by low spatial variance in Fig. 6 and the video) indicating that the fusion in $R_3$'s cooperative tracker diminished the influence of $R_2$'s observation model because of its low self localization confidence hence protecting itself from tracking the ball in a wrong position.

## V. Conclusion and Future Work

After performing a considerable number of tests on real robots we conclude that our algorithm provides a robust approach for tracking an object cooperatively by a team of robots and support this on a set of real robot experimental

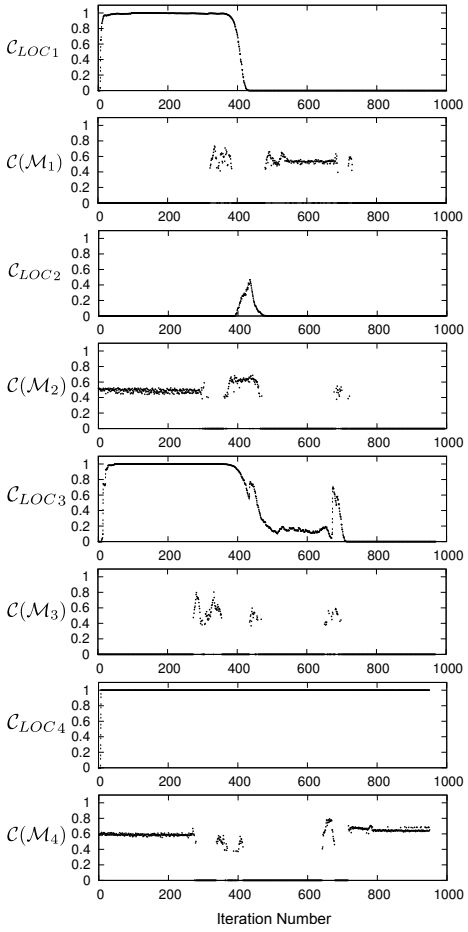


Fig. 5. Confidence plot of experiment 2 (Each iteration is performed every 0.1 second)

data. However, a few major points can be enumerated: the approach is a solution for tracking an object which is likely to be occluded or partially occluded quite often. If the object is confidently located by a wrongly localized robot, after fusion it would track it correctly in its own local frame and affect other teammates' fused observation model quite insignificantly. By sharing an observation PDF (in which only the models' parameters are enough to construct the whole PDF), we significantly reduce the use of bandwidth and communication time which leads to real-time tracking of the object. We are working further to extend our model of cooperative tracking to include multiple sensors on the same robot as well as to relax the 2-D space and colored object tracking assumptions. Tracking a random colored ball in 3-D space will require a new observation model and an extension in the state space without any changes in our current tracker's algorithm because it uses a generalized observation model and is independent of the state space dimensions. Furthermore we intend to extend our current approach to active cooperative tracking where we dynamically change the geometry of the robot formation so as to reduce the uncertainty of the belief in the object recognition.

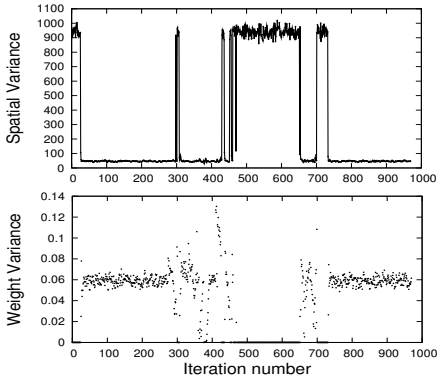


Fig. 6. Spatial Variance and Variance of the weights of the Particles of the cooperative tracker at $R_3$ in experiment 2

## REFERENCES


[1] Alper Yilmaz, Omar Javed and Mubarak Shah *Object Tracking: A Survey,* ACM Computing Surveys, Vol.38, No.4, Article 13, December 2006.

[2] M. Taiana, J. Santos, J. Gaspar, J. Nascimento, A. Bernardino and P. Lima *Tracking objects with generic calibrated sensors : an algorithm based on color and 3D shape features,* Robotics and Autonomous Systems, special issue on Omnidirectional Robot Vision, Vol. 58, Issue 6, 30 June, pp. 784-795, 2010.

[3] J. Santos and Pedro Lima *Multi-Robot Cooperative Object Localization - Decentralized Bayesian Approach,* Proc. of RoboCup 2009 Symposium, Graz, Austria, 2009.

[4] Raúl A. Lastra, Paul A Vallejos and Javier Ruiz-del-Solar *Integrated Self-Localization and Ball Tracking in the Four-Legged Robot Soccer League,* 1st IEEE Latin American Robotics Symposium, Mexico city, 2004.

[5] Jun Inoue, Akira Ishino and Ayumi Shinohara *Ball tracking with velocity based on Monte-Carlo localization,* 9th International Conference on Intelligent Autonomous Systems, Tokyo, Japan, March 2006.

[6] Walter Nisticó, Matthias Hebbel, Thorsten Kerkhof and Christine Zarges *Cooperative Visual Tracking in a Team of Autonomous Mobile Robots,* RoboCup 2006: Robot Soccer World Cup X. Lecture Notes in Artificial Intelligence, Springer 2007.

[7] Cody Kwok and Dieter Fox *Map-based Multiple Model Tracking of a Moving Object,* RoboCup 2004: Robot Soccer World Cup VIII, Lecture Notes in Computer Science 2005.

[8] Thrun, Sebastian and Burgard, Wolfram and Fox, Dieter *Probabilistic Robotics,* The MIT Press, 2005.

[9] Hastie Trevor, Tibshirani Robert and Friedman Jerome *The Elements of Statistical Learning,* Springer, 2001.

[10] Daniel Göhring and Hans-Dieter Burkhard *Multi Robot Object Tracking and Self Localization Using Visual Percept Relations,* Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 9 - 15, 2006, Beijing, China.

[11] Luis Merino, Fernando Caballero, J. R. Martínez-de Dios, Joaquín Ferruz, and Aníbal Ollero *A Cooperative Perception System for Multiple UAVs: Application to Automatic Detection of Forest Fires,* Journal of Field Robotics 23(3/4), 165-184; 2006

[12] Ong, L.-L., Upcroft, B., Bailey, T., Ridley, M., Sukkarieh, S., and Durrant-Whyte, H. (2006a). A decentralised particle filtering algorithm for multi-target tracking across multiple flight vehicles. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4539 –4544.

[13] Ong, L.-L., Upcroft, B., Ridley, M., Bailey, T., Sukkarieh, S., and Durrant-Whyte, H. (2006b). Consistent methods for decentralised data fusion using particle filters. In *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, pages 85 –91.

[14] Sheng, Xiaohong and Hu, Yu-Hen and Ramanathan, Parameswaran. Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network. *Proceedings of the 4th international symposium on Information processing in sensor networks* , IPSN 2005.

[15] Michael D. Breitenstein and Fabian Reichlin and Bastian Leibe and Esther Koller-Meier and Luc Van Gool. Online Multi-Person Tracking-by-Detection from a Single, Uncalibrated Camera. in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010