Multi-Robot Cooperative Object Localization
Decentralized Bayesian Approach

João Santos and Pedro Lima

Institute for Systems and Robotics, Instituto Superior Técnico, 1049-001 Lisboa, Portugal
{jsantos,pal}@isr.ist.utl.pt

Abstract. We introduce a multi-robot/sensor cooperative object detection and tracking method based on a decentralized Bayesian approach which uses particle filters to avoid simplifying assumptions about the object motion and the sensors' observation models. Our method is composed of a local filter and a team filter. The local filter receives a reduced dimension representation of its teammates' sample belief about the object location, i.e., the parameters of a Gaussian Mixture Model (GMM) approximating the other sensors' particles, and mixes the particles representing its own belief about the object location with particles sampling the received GMM. All particles are weighted by the local observation model and the best ones are re-sampled for the next local iteration. The team filter receives GMM representations of the object in the world frame, from the sensor teammates, and fuses them all performing Covariance Intersection among GMM components. The local estimate is used when the sensor sees the object, to improve its estimate from the teammates' estimates. The team estimate is used when the sensor does not see the object alone. To prevent the fusion of incorrect estimates, the disagreement between estimates is measured by a divergence measure for GMMs. Results of the method application to real RoboCup MSL robots are presented.

1 Introduction

A team of robots cooperatively tracking an object becomes a team of sensors, each making observations to build a perception of reality that can be improved by the others. Multisensor fusion addresses the problem of combining all the information from multiple sensors in order to yield a consistent and coherent description of the observed environment. The problem itself comes from the fact that the sensors information is always uncertain, usually partial, occasionally incorrect and often geographically or geometrically incomparable with other sensor views [1].

A sensor model describes the uncertainty associated with each sensor observation and location allowing to extract relevant information. The models are often nonlinear resulting in non-Gaussian posterior distributions. However, a parametric (e.g. Gaussian) approximation of sensors information is usually a better choice given the low computational power and low communications bandwidth it requires. This is achieved at the cost of a limited representation of the sensors belief. On the other hand, non parametric discrete approximations, such as Particle Filters, are able to capture arbitrarily complex uncertainty, but are intractable when it comes to communicating the state distribution due to the necessity of transmitting a large sample-based representation.

Each sensor is part of a network node which has local computational power and is able to communicate with nearby nodes. In RoboCup, recent rules to forbid communications with exterior computers push the research towards decentralized sensor network topologies or centralized based topologies with a dynamic leader node [2]. Several teams have taken the decentralized way for a fully multi-agent approach [3] [4] [5] [6]. However, the implementations described rely mostly on parametric sensor models. We propose a decentralized approach based on a probabilistic framework from non-parametric sensors, where communication constraints must be taken into account.

This paper introduces a a cooperative perception model based on particle filters and a framework for representing and measuring disagreement of sensor information based on Gaussian Mixture Models. Our soccer robots (RoboCup Middle Size League (MSL) ISocRob team) are equipped with an omnidirectional camera with limited resolution that hardly provides a global view of the field. Our main motivation is to take real advantage of this team of mobile sensors scattered across the field, in order to provide a broader view while locating and tracking the ball. We are further motivated in benefiting from a multisensor system upon the challenges constantly imposed by RoboCup MSL such as the global localization in a symmetric environment or the tracking of the (yet to come) arbitrary color ball.

The paper is organized as follows. In Section 2 we review related work. Section 3 describes the implementation of a shape-based 3D tracker for the ball using a single camera. Section 4 presents a compact sensor information representation based on Gaussian Mixture Models (GMMs) and introduces a decentralized Bayesian approach to multisensor fusion that takes advantage of distributed particle filters and GMM modeling. In Section 5 we present experimental results to validate the presented methods. Section 6 outlines our conclusions.

2 Related Work

Most of the previous work focus on merging the ball localization estimates provided by several sensors to one consistent estimate among the team of robots. Lau et al. [7] calculate the mean and standard deviation of all ball estimates for discarding outliers and then assumes the ball information of the teammate closest to it. Ferrein et al. [8] describe a weighted mean of the estimates according to the distance from the robot to the ball and a time factor denoting how long ago the robot has seen the ball for the last time. On a more probabilistic approach, Stroupe et al. [9] represent ball estimates as a two-dimensional gaussian in canonical form, allowing to merge them by multiplication, and use a Kalman filter to predict the ball position. Pinheiro and Lima [10] implemented a multi-Bayesian team of robots as a direct application of the sensor fusion method introduced by Durrant-Whyte [1]. This approach detects sensors disagreement based on the Mahalanobis distance and achieve a team consensus faster. Other approaches also accounted for merging weighted gridcells from ball occupancy maps [11], Monte Carlo (ball) localization [12] or a combination of Kalman filter with Markov localization [13]. However, although mentioned in some approaches, none of these take into consideration the robots own localization uncertainty, frequently assuming a highly accurate self-localization method. This is problematic because fusion usually takes place in the global reference frame for the team, therefore local estimates must be transformed to global estimates before fusion, and the sensor localization uncertainty plays a major role in this. Pahliani and Lima [14] proposed a new cooperative localization algorithm that reduces the uncertainty of both self-localization and object localization. This method tries to overcome the performance of two popular algorithms for fusing sensor observations: Linear Opinion Pool and Logarithmic Opinion Pool. The implementation although, is based on multi-robot Markov Localization and assumes one can distinguish and locate different team-mates, which is a complex task given the current RoboCup environment.

On other domains, Rosencrantz et al. [15] introduced a scalable Bayesian technique for decentralized state estimation with distributed particle filters using a selective communication procedure over the particle set. On the other hand, instead of selecting which particles to communicate, Upcroft et al. [16] demonstrated the validity of approximating a particle set using Gaussian mixture models or Parzen representations in Decentralized Data Fusion (DDF) systems. However, at every given network node, all sensors are treated as equals, i.e., there is one data association proccess that is impartial to whether the current node is actually tracking the target or not. This means that we are implicitly assuming that the result of the fusion process is more relevant than the local sensor observations. Therefore, we present an approach where each node builds its perception from other sensor nodes observations, and yet relys on a fusion estimation proccess for critical situations, i.e., when the the target is out of the sensor range.

3 Ball Detection and Tracking

Our ball tracking observation model is based on Taiana et al. [17]. A 3D model of the ball is used to calculate it's 2D contour projected on the image. The expected ball contour on the image is computed from its 3D shape projection on the 2D image plane. The ball has rotational symmetry which reduces the problem dimension for there is no need to consider the object orientation. Given a 3-dimensional position, the projection model tell us how the ball contour is going to look like in the image. However, to track it, one needs to estimate the ball's location with respect to the robot. For that we use a particle filter to represent the ball's state space regarding position and velocity $\mathbf{x}_t = [x, y, z, \dot{x}, \dot{y}, \dot{z}]^T$. We start by assuming a simple Markov process for the underlying dynamics of the ball specified by a transition probability, from here and henceforth denoted as motion-model, $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, and that for every time step $t > 1$ a new observation z_t about the state $\mathbf{x}_t$ is made. Given the observation history at time t by $Z_t = [z_1, ..., z_t]$ our goal is to estimate the posterior distribution $p(\mathbf{x}_t|Z_t)$ for each time step. This can be done recursively over *Prediction* and *Update* steps:

$$Prediction : p(\mathbf{x}_t|Z_{t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|Z_{t-1})d\mathbf{x}_{t-1} \tag{1}$$

$$Update : p(\mathbf{x}_t|Z_t) \propto p(z_t|\mathbf{x}_t)p(\mathbf{x}_t|Z_{t-1}) \tag{2}$$

where $p(\mathbf{x}_{t-1}|Z_{t-1})$ is the previous estimate and $p(z_t|\mathbf{x}_t)$ is the observation model. At a given moment in time t, the particle filter represents the probability distribution of the state as a set of N weighted samples $\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}_{i=1}^{N}$, such that the posterior is approximated by an empirical estimate:

$$p(\mathbf{x}_t|Z_t) \approx \sum_{i=1}^{N} w_t^{(i)} \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)}) \tag{3}$$

where $\delta(.)$ is the Dirac delta function. The estimation of the best state is computed through a discrete Monte Carlo approximation of the expectation:

$$\hat{\mathbf{x}}_t \doteq \frac{1}{N} \sum_{i=1}^{N} w_t^{(i)} \mathbf{x}_t^{(i)} \tag{4}$$

Prediction computes an approximation of $p(\mathbf{x}_t|Z_{t-1})$ by moving each particle according to the ball motion model. We assume a constant velocity model where the motion equations correspond to a uniform acceleration during one time step:

$$\mathbf{x}_t = \begin{bmatrix} I & (\mathit{\Delta} t)I \\ 0 & I \end{bmatrix} \mathbf{x}_{t-1} + \begin{bmatrix} (\frac{\mathit{\Delta} t^2}{2})I \\ (\mathit{\Delta} t)I \end{bmatrix} \mathbf{a}_t \tag{5}$$

where I is the 3×3 identity matrix, $\mathit{\Delta} t$ in general represents the sampling time, and $\mathbf{a}_t$ is a 3×1 white zero mean random vector corresponding to an acceleration disturbance.

In the *Update* step, the particle's weights are updated according to the computed likelihood $p(z_t|\mathbf{x}_t^{(i)})$ for each hypothesis, from the observation model. We follow Taiana's [17] approach to compute the likelihood as a function of similarities between color histograms. We compute two YUV histograms for the inner and outer boundaries of the ball 2D projection contour and apply the Bhattacharyya [18] similarity metric. In order to track arbitrary color balls, we do not define a reference color model for the inner boundary and rely strictly on its mismatch to the outer boundary, that is the object to background dissimilarity. This is well suited given the RoboCup scenario, where the background is mostly the field color and the ball color, no matter what, will always have to contrast with it. The motion model described in Eq. (5) remains valid as long as we express the state of the ball in terms of the world reference frame W which, as opposed to the robot reference frame R, is inertial. As so, the robot pose must be taken into account in the observation model in order to project a 3D point M onto the image plane. This means that, at every time step, the coordinates expressed in the world reference frame ${}^{W}M = [{}^{W}X, {}^{W}Y, {}^{W}Z, 1]$ must be transformed to the robot reference frame ${}^{R}M = [{}^{R}X, {}^{R}Y, {}^{R}Z, 1]$ by means of a transformation matrix, which comprises a rotation matrix ${}^{R}R_W$ and a translation vector ${}^{R}t_W$:

$${}^{R}T_W = \begin{bmatrix} {}^{R}R_W & {}^{R}t_W \\ 0 & 1 \end{bmatrix} \tag{6}$$

The particles that have a higher weight are replicated in the *Resampling* step, and the rest of the particle set is discarded. To prevent the loss of diversity in the particle population, we use a low variance resampling technique.

We initialize our tracker by uniformly spreading a fixed number of ball hypothesis on the ground, in a 5 meter radius circle surrounding the robot. This enable us to reduce the search state space, as we assume the ball is on the floor, and constrains the detection according to the camera resolution.

4 Cooperative Perception in Mobile Sensor Networks

4.1 Information Representation

In order to communicate the ball location and the sensor uncertatinty to other teammates one cannot transmite the entire particle set that approximates the posterior in Eq. (3). The conversion of our sample-based non-parametric representation to a continuous distribution requires the use of methods such as kernel density estimation, but in order to achieve efficient communication a parametrization of the probability density function is, in fact, mandatory. A mixture model provides this type of representation and can also be viewed as a type of kernel method [19]. If the kernel function of the mixture model is Gaussian, the distribution is expressed as a Gaussian Mixture Model (GMM) of the form:

$$P(\mathbf{x}) = \sum_{k=1}^{N} w_k G(\mathbf{x}|\mu_k, \Sigma_k) \tag{7}$$

where **x** are the observations of the random variable **X**, w_k are positive weights such that $\sum_{i=k}^{N} w_k = 1$, G is a Gaussian probability density (Gaussian mixture component) with mean μ_k and covariance Σ_k, and N is the total number of mixture components. For the GMM to be of practical importance both for data fusion and communications, the density estimation technique, which will lead to the parametrization of the mixture model, must be computationally fast and accurate.

The Expectation Maximization (EM) algorithm is an efficient iterative method to the general approach of the maximum likelihood parameter estimation in the presence of missing data. Our main intuition while using EM is to alternate between estimating which sample from our sample-based representation belongs to which mixture component (missing data) and estimating the unknown parameters $\Theta_k = (w_k, \theta_k)$, where $\theta_k = (\mu_k, \Sigma_k)$, for each of those components. Each iteration of the EM consists of an expectation (E-step) and a maximization step (M-step). In the E-step we compute the expected likelihood for the complete data Γ (also known as Q-function) as the conditional distribution of the missing data Y, given the current settings of parameters Θ and the observed incomplete data **X**. So, using Bayes's rule, for each mixture component k:

$$p(y_i = k|x_i, \theta_k) = \frac{p(y_i = k, x_i|\theta_k)}{p(x_i|\theta_k)} = \frac{p(x_i|y_i = k, \theta_k)p(y_i = k|\theta_k)}{\sum_{k=1}^{N} p(x_i|y_i = k, \theta_k)p(y_i = k|\theta_k)} \tag{8}$$

where N is the total number of mixture components and $p(x_i|y_i = k, \theta_k)$ is, in our case, the multivariate Gaussian probability density function. One should also note that the probability of an observation being part of the k^{th} component $p(y_i = k|\theta_k)$ is actually its relative weight w_k in the mixture model. In the M-step we re-estimate the mixtures parameters Θ by maximizing the Q-function (see [19],[20] for the in-depth derivation). From here we can compute a new approximation Θ' for each component k:

$$\mu'_k = \frac{\sum_{i=1}^{M} x_i p(y_i = k|x_i, \theta_k)}{\sum_{i=1}^{M} p(x_i|y_i = k, \theta_k)}, \Sigma'_k = \frac{\sum_{i=1}^{M} p(x_i|y_i = k, \theta_k)(x_i - \mu'_k)(x_i - \mu'_k)^T}{\sum_{i=1}^{M} p(x_i|y_i = k, \theta_k)} \tag{9}$$

where M is the number of total observations. The relative weight of each Gaussian mixture is given by:

$$w'_k = \frac{1}{M}\sum_{i=1}^{M} p(y_i = k|x_i, \theta_k). \tag{10}$$

4.2 Cooperative Sensor Model

The decentralized sensor fusion typical approach is to build one single estimate of the target, regardless of whether it's being tracked by the local sensor or not, and always assume that in the worst case we filter out our individual local estimate and use the other robot fused estimate. We propose a different approach that consists of not taking other sensors beliefs for granted, and instead use them as if they were observations gathered by the local sensor (virtual observations).

From the previously described particle filter based perception framework in Section 3, we present herein a cooperative perception model that copes both with a local sensor-distributed estimate of the object and a fused team estimate, deals with the correlation between common information and can be used to improve self-localization. The model, based on sequential Bayesian filtering representation, is illustrated in Fig. 1.

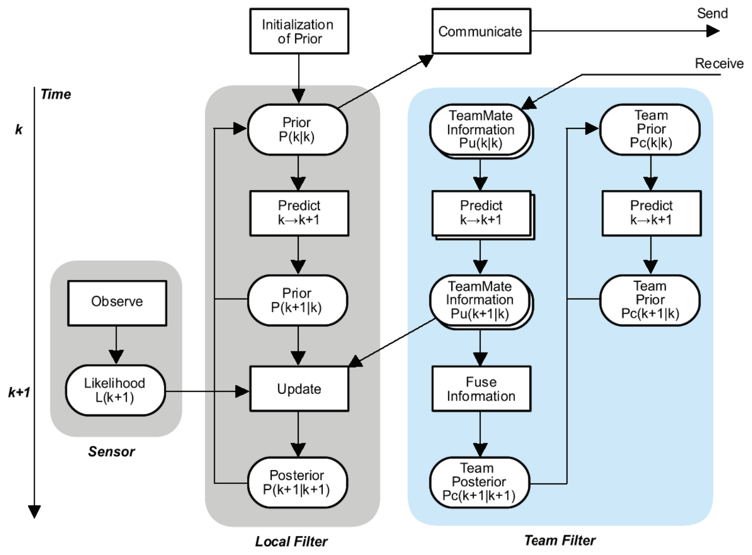

Fig. 1. Decentralized Mobile Cooperative Sensor Model (adapted from [16])

In the Local Filter, observations are made and used to compute the likelihood of the sensor model, which is then multiplied by the prior belief in the Update step. Both the *local* prior (before observation), predicted from the local posterior (after Update) over the previous state, and the *team* prior, predicted from the received posterior distributions of the teammates, are concurrently computed at each robot. This way, the other robots information will only influence the prior belief and the posterior will be determined according to the local sensor measurement model. In the update step, we sample from the prior distribution, denoted in particle filters as the proposal distribution $\overline{bel}(\mathbf{x}_t)$, and our goal is that the weighted particle set approximates the posterior, denoted as the target density $bel(\mathbf{x}_t)$. Upon resampling, the particles are distributed according to the posterior:

$$bel(\mathbf{x}_t^{[m]}) = \eta p(z_t|\mathbf{x}_t^{[m]})\overline{bel}(\mathbf{x}_t) \tag{11}$$

where $p(z_t|\mathbf{x}_t^{[m]})$ is the probability of measurement z_t under the mth particle $\mathbf{x}_t^{[m]}$. The target density is then transformed in a compact GMM representation and passed on to the other robots. When it is received, new samples will be drawn from it contributing for the proposal distribution. The ability to sample is not given for arbitrary distributions, however, since our distributions can actually be decomposed in a sum of Gaussians, we can draw a random vector $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)^T$ from each bivariate component k with mean μ_k and covariance matrix Σ_k from:

$$\mathbf{x}_k^{[n]} = A_k v^{[n]} + \mu_k \tag{12}$$

where v are n independent samples drawn from $N(0, I_2)$ and A_k is the Cholesky decomposition of Σ_k, such that $\Sigma_k = AA^T$. For each new particle $\mathbf{x}^{[n]}$ we then calculate the importance factor w as described in the ball tracking Update step, Section 3. As such, samples generated from received GMMs that do not follow the local observation model will have a low likelihood and will be discarded on resampling.

In the Team Filter we receive GMM representations of the ball's posterior in the world frame. Regarding information fusion, the Covariance Intersection (CI) filter yields consistent estimates to the problem of combining different Gaussian random vectors with unknown correlation between them. This can be extended to a GMM Covariance Intersection algorithm as in [16], by performing CI between each of the mixture components. The fusion between the ith component of a GMM and the jth component of another GMM will result in a Gaussian mixture with $N \times N$ components, such that:

$$\Sigma_{ij}^{-1} = \gamma\Sigma_i^{-1} + (1-\gamma)\Sigma_j^{-1} \tag{13}$$

$$\mu_{ij} = \Sigma_{ij}(\gamma\Sigma_i^{-1}\mu_i + (1-\gamma)\Sigma_j^{-1}\mu_j) \tag{14}$$

$$w_{ij} = \frac{1}{N}(\gamma w_i + (1-\gamma)w_j) \tag{15}$$

where $0 \leq \gamma \leq 1$ is a weighting parameter to minimize the determinant of the result. This parallel team estimate is to be used only in critical conditions when the target is out of the sensor field of view.

When associating data in distributed systems, an incorrect association decision leads to an incorrect fusion estimate, therefore ones needs to have the ability to measure agreement among disparate sensors before fusing their observations. A distance measure between Gaussian distributions can be defined as Kullback-Leiber distance [21], Bhattacharyya distance [18] and others. However there's no analytical solution of computing these measures to evaluate the distance between Gaussian mixture models. Therefore, we take Beigi et al. [22] approach to measure distances between collections of distributions in speech recognition, and define our measure of divergence between GMMs as:

$$D(G_1, G_2) = \frac{\sum_{i=1}^{N} W_i^1 + \sum_{j=1}^{N} W_j^2}{\sum_{i=1}^{N} c_i + \sum_{j=1}^{N} c_j} \leq \xi \tag{16}$$

and assume there is agreement if $D(G_1, G_2) \leq \xi$, where ξ is a positive threshold. Consider the matrix of distances between $N \times N$ mixture componentes:

$$T = \begin{bmatrix} d_{11} & d_{12} & ... & d_{1N} \\ d_{21} & d_{22} & ... & d_{2N} \\ ... & ... & ... & ... \\ d_{N1} & d_{N2} & ... & d_{NN} \end{bmatrix} \tag{17}$$

W_i^1 is the minima of the elements in the row times the row number c_i. Likewise, W_j^2 is the minima of the elements in the column times the column number c_j. We can compute d_{ij} from the above metrics for Gaussian distributions. We choose to apply the Bhattacharyya distance for multivariate Gaussian distributions.

4.3 Improving Self-Localization

Our current self-localization method is a combination of Monte Carlo Localization with gyrodometry and line points extraction. However, one of the issues that affects MCL performance is the difficulty to recover from failures. One typical recover approach consists in gradually augmenting the proposal distribution by systematically adding more and more particles until better observation likelihoods can be obtained. Two major drawbacks can put this approach at risk. One is the large amount of computational power required to draw and test samples from an augmented proposal distribution that can comprise the entire state space. The other drawback is the inability to deal with local maxima that are present in symmetric environments, such as the RoboCup field.

Instead, one can now see the problem as feature-based map localization with known correspondence, that is $p(\mathbf{r}_t | f_t^i, c_t^i, \mathbf{m})$, where $\mathbf{r}_t$ is the robot pose and f_t denotes a given feature that has a correspondence c_t in a list of landmarks $\mathbf{m}$. Let's consider the ball as a landmark $\mathbf{m}_1$. If some other robots are localized and tracking the ball, the coordinates $\mathbf{m}_{1,x}$ and $\mathbf{m}_{1,y}$ of our landmark in the world frame of the map are given by the Team Filter estimate. If the lost robot is tracking the ball relative to its local coordinate frame (Local Filter), it can make new guesses of its own whereabouts for it now knows it may be on a circle around the landmark. These new guesses represent new poses that incorporate the sensor measurement $p(f_t^i | c_t^i, \mathbf{r}_t, \mathbf{m})$. We can assume the robot is completely lost and therefore the prior $p(\mathbf{r}_t | c_t^i, \mathbf{m})$ is uniform. This assumption leads to:

$$\begin{aligned} p(\mathbf{r}_t | f_t^i, c_t^i, \mathbf{m}) &= \eta p(f_t^i | c_t^i, \mathbf{r}_t, \mathbf{m}) p(\mathbf{r}_t | c_t^i, \mathbf{m}) \\ &= \eta p(f_t^i | c_t^i, \mathbf{r}_t, \mathbf{m}) \end{aligned} \tag{18}$$

from where we concluded that sampling from $p(\mathbf{r}_t | f_t^i, c_t^i, \mathbf{m})$ can, in this particular case,

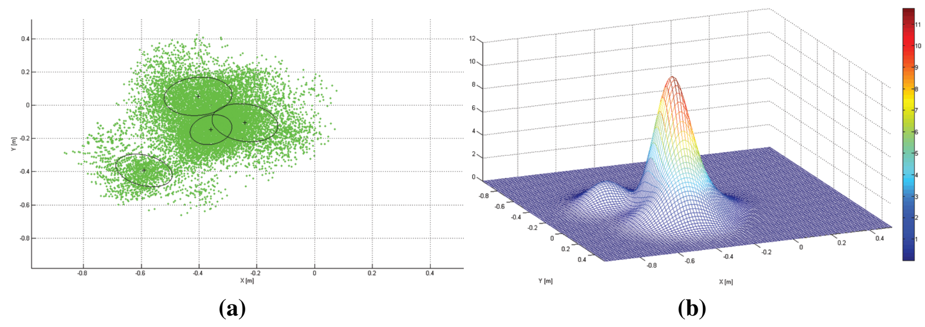

Fig. 2. Parametrization of the Ball Tracking particle set using Expectation Maximization. Particle set in green, and the respective Gaussian mixture components in black. (b) Density function.

be achieved from $p(f_t^i|c_t^i, \mathbf{r}_t, \mathbf{m})$. As so, we only add a limited amount of new sample poses that derive from a common target observation to the MCL proposal distribution.

5 Experimental Results

All experiments were made online, on a Intel Centrino 1.6 GHz processor, in real game situations. To achieve maximum processor performance, the implementation was done in C++ with extensive use of Intel Performance Primitives (IPP) for optimized vector and matrices operations and Intel Math Kernel Library (MKL) for statistics procedures.

5.1 Generating Compact Information Representations

In this experiment we tested the particle set approximation with EM. The purpose was to test the algorithm efficency and determine a good number of Gaussian components that would suit an aceptable EM convergence time. We ran our EM implementation with a different number of mixture components for the same scenario and registered the average run time of the algorithm in Table 1. As so, we choose to use GMMs with 4 components for it is enough to capture a good approximation of the particle set (Fig. 2) in an aceptable time. All the processing was made online with 1200 particles.

Table 1. Average execution time while computing GMMs with our EM implementation for 12000 particles, concurrently with other modules used to play soccer

Number of mixture components	**1**	**2**	**4**	**10**
Time taken [seconds]	0.0246	0.0690	0.1131	0.1924

5.2 Fusing Data

In this experiment three robots are able to localize the ball, while a fourth robot (the goalkeeper) cannot (see Fig. 3). The robots tracking the ball compute their GMM approximation and broadcast it to others. As the goalkeeper receives the teammates GMM estimates, it first tests it to see if there's agreement, and if there is, it proceeds to compute a team estimate by fusing the GMMs with CI.

Although its able to track the ball, robot4 is not able to localize itself correctly on the field (Fig. 3a). As such, it broadcasts GMM approximation of its erroneous ball localization belief, since it is corrupted by its self localization belief (Fig. 3b).

We show the results of the fusion estimate made by robot1 (Fig. 3g, 3h), which is not able to see the ball at all. The decision on which GMMs to fuse is based on the disagreement measurement Eq.(16) with $\xi = 30$. The computed distances between each of the received GMMs are shown is Table 2.

Table 2. Distance measurements between the GMMs received by robot1

$\mathbf{D(G_2, G_3)}$	9.6880
$\mathbf{D(G_3, G_4)}$	200.5146
$\mathbf{D(G_2, G_4)}$	162.7252

6 Conclusions

We presented a cooperative sensor fusion model based on a particle filter perception framework, for mobile robots operating in dynamic environments. We aim at taking advantage of a team of sensors to detect the ball on the field at all time.

For that we implemented a 3D shaped-based ball tracker that comprises a realistic dynamic motion model. The system is based on particle filters and also comprises an observation model that allow us to compute the likelihood of a ball hypothesis, given the ball shape model, the projection model for the omnidirectional camera and an acquired image. To acquaint for the robot motion in the tracker we take the robots pose into consideration in the observation model.

We presented a framework for representing and measuring disagreement of sensor information based on Gaussian Mixture Models. This representation allows to capture arbitrary complex uncertainty from nonlinear observation models, yet it's parametrization is simple and takes no overhead in communications. We implemented the Expectation Maximization algorithm for GMM parameter estimation to approximate the sample based ball posterior distribution.

The implemented cooperative perception model takes advantage of the GMM representation in two distinct forms. One is to improve the local ball particle filter in a distributed fashion way by injecting new particles drawn directly from the received GMMs. The other is to compute a ball team estimate directly from the received GMMs target distribution with Covariance Intersection if there's GMM agreemeant, when the the ball cannot be detected by the local sensor.

References

1. H. F. Durrant-Whyte, “Sensor models and multisensor integration,” *International Journal of Robotics Research*, vol. 7, no. 6, pp. 97–113, 1988.
2. F. Zhao, J. Shin, and J. Reich, “Information-driven dynamic sensor collaboration for tracking applications,” *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 61–72, 2002.
3. J. L. Azevedo, N. Lau, G. Corrente, A. Neves, M. B. Cunha, F. Santos, A. Pereira, L. Almeida, L. S. Lopes, P. Pedreiras, J. Vieira, D. Martins, N. Figueiredo, J. Silva, N. Filipe, and I. Pinheiro, “Cambada2008: Team description paper,” University of Aveiro, Tech. Rep., 2008.
4. O. Zweigle, U.-P. Kappeler, T. Ruhr, K. Haussermann, R. Lafrenz, F. Schreiber, A. Tamke, H. Rajaie, A. Burla, M. Schanz, and P. Levi, “Cops stuttgart team description 2007,” University of Stuttgart, Tech. Rep., 2007.
5. T. Hafner, S. Lange, M. Lauer, , and M. Riedmiller, “Brainstormers tribots team description,” University of Osnabruck, Tech. Rep., 2008.
6. T. Suzuki, N. Tomoyasu, M. Takafashi, and K. Yoshida, “Eigen keio univ. team description,” Keio University, Tech. Rep., 2008.
7. N. Lau, L. S. Lopes, and G. A. Corrente, “Cambada: Information sharing and team coordination,” in *Robótica 2008*, 2008.
8. A. Ferrein, L. Hermanns, and G. Lakemeyer, “Comparing sensor fusion techniques for ball position estimation,” in *RoboCup 2005 Symposium*, 2005.
9. A. W. Stroupe, M. C. Martin, and T. Balch, “Merging probabilistic observations for mobile distributed sensing,” Carnegie Mellon University, Tech. Rep., 2000.
10. P. Pinheiro and P. Lima, “Bayesian sensor fusion for cooperative object localization and world modeling,” in *8th Conference on Intelligent Autonomous Systems*, 2004.
11. A. Cai, T. Fakuda, and F. Arai, “Information sharing among multiple robots for cooperation in cellular robotic system,” in *Intelligent Robots and Systems*, 1997.
12. G. Steinbauer, M. Faschinger, G. Fraser, A. Muhlenfeld, S. Richter, G. Wober, and J. Wolf, “Mostly harmless team description,” Graz University of Technology, Tech. Rep., 2003.
13. M. Dietl, J.-S. Gutmann, and B. Nebel, “Cs freiburg: Global view by cooperative sensing,” in *International RoboCup Symposium*, 2001.
14. A. Pahliani and P. Lima, “Cooperative opinion pool: a new method for sensor fusion by a robot team,” in *Intelligent Robots and Systems*, 2007.
15. G. G. Matt Rosencrantz and S. Thrun, “Decentralized sensor fusion with distributed particle filters,” in *Conference on Uncertatinty in AI (UAI)*, 2003.
16. B. Upcroft, L. Ong, S. Kumar, M. Ridley, T. Bailey, S. Sukkarieh, and H. Durrant-Whyte, “Rich probabilistic representation for bearing only decentralized data fusion,” in *7th International Conference on Information Fusion*, 2005.
17. M. Taiana, J. Santos, J. Gaspar, J. Nascimento, A. Bernardino, and P. Lima, “Color 3d model-based tracking with arbitrary projection model,” in *SIMPAR Omnidirectional Vision Workshop*, 2008.
18. A. Bhattacharyya, “On a measure of divergence between two statistical populations defined by their probability distributions,” in *Bulletin Calcutta Mathematical Society*, 1943.
19. T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2003, pp. 165–190, 236–242.
20. J. A. Bilmes, “A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models,” U.C. Berkeley, Tech. Rep., 1998.
21. S. Kullback, *Information Theory and Statistics*. Dover Books, 1968.
22. H. Beiji, S. Maes, and J. Sorensen, “A distance measure between collections of distributions and its application to speaker recognition,” in *International Conference on Acoustics, Speech and Signal Processing*, vol. 2, 1998, pp. 753–756.

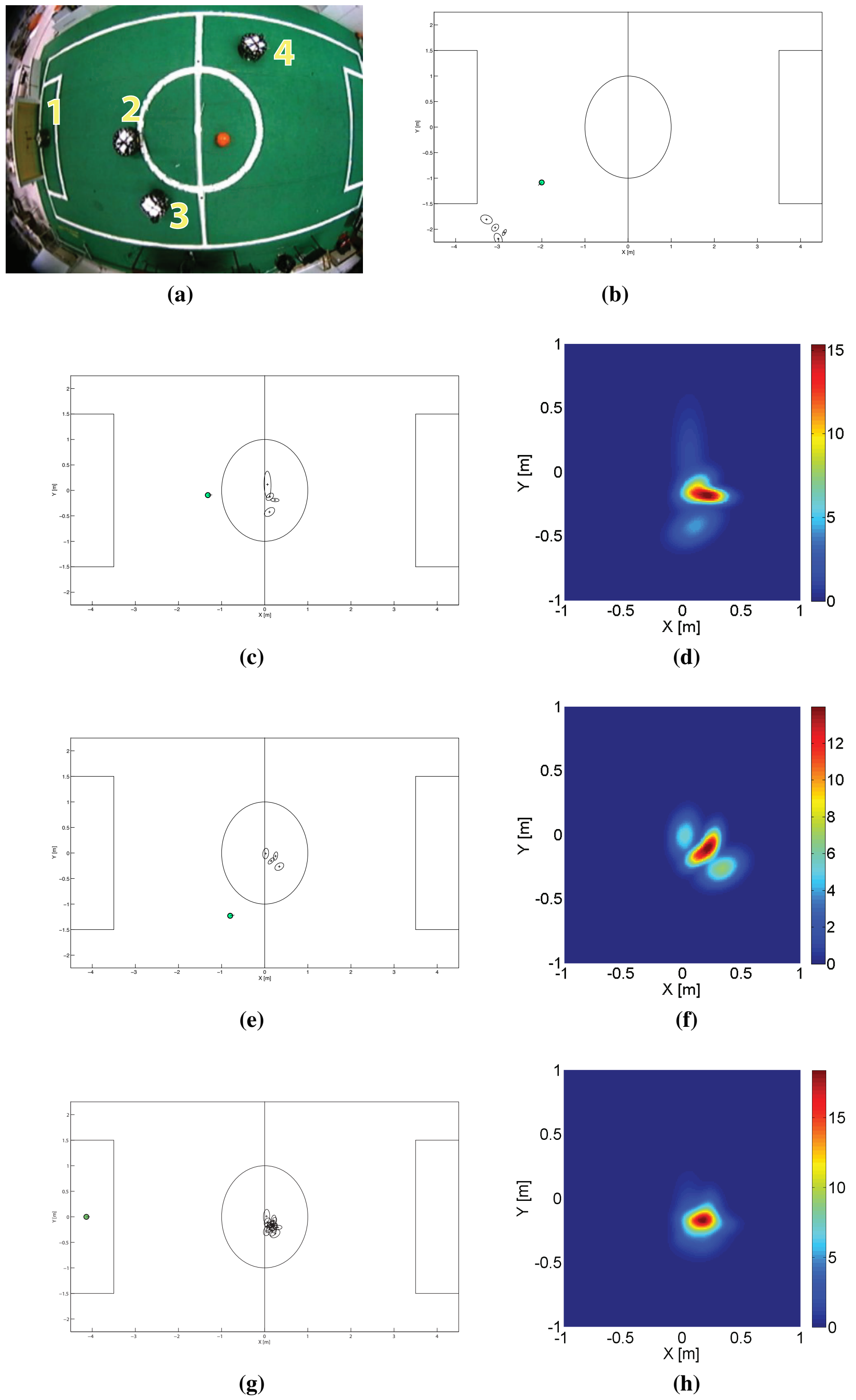

Fig. 3. GMM Data Fusion with Disagreement. (a) Top field camera view. (b) Robot4 tracks the ball and broadcasts its GMM, but is not well localized. (c-f) Robots 2 and 3 track the ball, compute their GMMs and broadcast them. (g-h) The goalkeeper (robot1) tests received GMMs for disagreement and computes GMM CI only for those that are in agreement.