International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:11, No:7, 2017

Distributed Coverage Control by Robot Networks in Unknown Environments Using a Modified EM Algorithm

Mohammadhosein Hasanbeig, Lacra Pavel

***Abstract*—**In this paper, we study a distributed control algorithm for the problem of unknown area coverage by a network of robots. The coverage objective is to locate a set of targets in the area and to minimize the robots' energy consumption. The robots have no prior knowledge about the location and also about the number of the targets in the area. One efficient approach that can be used to relax the robots' lack of knowledge is to incorporate an auxiliary learning algorithm into the control scheme. A learning algorithm actually allows the robots to explore and study the unknown environment and to eventually overcome their lack of knowledge. The control algorithm itself is modeled based on game theory where the network of the robots use their collective information to play a non-cooperative potential game. The algorithm is tested via simulations to verify its performance and adaptability.

***Keywords*—**Distributed control, game theory, multi-agent learning, reinforcement learning.

I. Introduction

MULTI-AGENT Coverage Control (MCC) is concerned with the design of rules for the action coordination of a set of agents towards achieving a desired coverage objective. MCC covers an extensive range of applications, both with static and dynamic natures. A static study is often concerned with efficient sensor placement for area coverage [1] or parameter estimation [2]. On the other hand, the dynamic version allows the agents to have a coordinated movement within the environment [3]. The main focus in the dynamic study is on optimal motion control of agents in various applications such as tracking problems [4], odor detection [5], search and rescue [6], and collision avoidance [7].

The major challenge in MCC is how to design an efficient coordination between the agents such that the system's coverage objective is met. In the last few years, game theory has drawn significant attention in providing such efficient coordination between the agents in MCC systems. Most of the studies done on MCC, within the game theory framework, are in the class of potential games (e.g., [8]-[10]). Potential game is a suitable class due to its useful features including finite improvement property. By this property, players are able to improve their decisions and to eventually reach a point beyond which no improvement can be achieved. This point is called potential game's Nash equilibrium.

M. Hasanbeig and L. Pavel are with the Edward S. Rogers Department of Electrical and Computer Engineering, University of Toronto, Toronto, M5S 3G4, Canada (e-mail: m.hasanbeig@mail.utoronto.ca, pavel@control.utoronto.ca).

The concept of "learning" in potential games is a critical notion which can provide such improvement for agents when the game is unknown. Learning schemes assume that players learn over time about the environment and also about the behavior of other players by constantly interacting with them. Such learning mechanisms agree with the social and biological situations in which players imitate, reinforce or update their belief. There are a variety of approaches to learning in games depending on the number of players, the information players are expected to know and the importance of the communication with other players [11].

In a game theoretic setup of MCC, it is common to assume that the agents have a level of autonomy, i.e., the control scheme is usually distributed. However, there are centralized control approaches to solving the MCC problem (e.g. [12]). In a centralized control method, a base unit controls agents in their searching task and agents are assumed not to be able to act autonomously. Although, a centralized scheme may achieve the desired objective, a distributed approach can demonstrate higher robustness, stability, adaptability, and scalability.

Apart from game theory, there exist different methods to solve the MCC problem. A decentralized gradient-search-based algorithm that is proposed in [13] is an example of such methods. In this method, a greedy algorithm is used to associate agents with a set of targets in the environment while the gradient-based algorithm provides agents with an optimal motion planning with respect to the targets' positions. Alternatively, in [14] the search area is divided into Voronoi regions and each region is assigned to one agent. The partitions between the Voronoi regions will change over time based on a probability map. This method is further extended with limited sensing and communications in [15] and limited power in [16].

In this paper, a network of robots is considered as a set of agents whose coverage objective is to localize a set of hidden targets while the robots' movement energy consumption is minimized. The set of targets are distributed across a geographical area and it is assumed that each of them emits a signal that is detectable by the robots. As an example, consider the multi-robot search-and-rescue problem in which a number of robots are employed to find a number of victims in an area. The area could be unknown and the robots have to trace any kind of signal emitted from the victims in order to locate them. This signal, in this case, can be an audio signal or other vital signs.

In [10] the agents are assumed to know the approximate location of the targets which limits practical applicability of [10]'s results. Although this assumption is relaxed in [8], one of the underlying assumptions of [8]'s method is agents' knowledge about the number of targets. With this assumption if there is more than one target in the area and if the targets are dispersed then [8]'s algorithm may fail to find the targets. The present work extends the results of [3], [8], [10] by assuming a completely unknown search area. We use the Akaike information criterion to develop an algorithm such that even if the number of targets is unknown the robots are able to estimate the number of the targets and efficiently locate the targets.

The paper is organized as follows: In Section II, a game theoretic solution is taken as the main control scheme. Next, a Gaussian mixture model estimation algorithm is introduced to estimate the targets' distribution parameters. Section III presents the Akaike information criterion which is used to optimize the estimated number of the targets. A communication mechanism is further investigated to let the agents share their information about the environment. In Section V simulation results are presented as well as a discussion and comparison of the performance of different methods.

II. Problem Formulation

Consider a set of finite number of robots $\mathcal{I}$ modeled as a network of agents that can move through a geographic area. A two-dimensional grid is applied over this area and each cell l in this grid has a square shape of unit dimensions. A cell l has the center coordinates $(l_x, l_y) \in \mathcal{L}$, where $\mathcal{L}$ is defined as the collection of the centroid of all square cells. We may use l to refer to the cell l's coordinates (l_x, l_y). The agents can only move between the centroids of the grid cells and the location of agent i at time step n is denoted by $\alpha^i(n) \in \mathcal{L}$, $i = 1, ..., |\mathcal{I}|$. We may refer to $\alpha^i(n)$ as agent i's action at time step n.

Let the set of agent i's neighbor cells at time step n be defined as $N_R^i(\alpha^i(n)) = \{l \in \mathcal{L} \; s.t. \; |\alpha^i(n) - l| \leq R\}$ where R is the neighborhood radius. The agent i's available action set at time step n is denoted by $\mathcal{A}_n^i \subset \mathcal{L}$, and is defined as $\mathcal{A}_n^i = N_\delta^i(\alpha^i(n))$ where δ is the action selection radius. The total available action set can be defined as $\mathcal{A}_n = \mathcal{A}_n^1 \times \mathcal{A}_n^2 \times ... \times \mathcal{A}_n^{|\mathcal{I}|}$ and the action profile of all agents is denoted by $\alpha(n) = (\alpha^1(n), \alpha^2(n), ..., \alpha^{|\mathcal{I}|}(n))$. We may show the action profile of all agents as $\alpha(n) = (\alpha^i(n), \alpha^{-i}(n))$ where $\alpha^{-i}(n) = (\alpha^1(n), \alpha^{i-1}(n), ..., \alpha^{i+1}(n), \alpha^{|\mathcal{I}|}(n))$.

The worth of a cell l is defined as the probability that a target exists in that cell and its distribution over $\mathcal{L}$ is denoted by $f : \mathcal{L} \mapsto [0, 1]$. The worth of each cell is directly proportional to the strength of the sensed signal on that cell. The f distribution over $\mathcal{L}$ is constant in time since we assume that targets are stationary. An agent $i \in \mathcal{I}$ can determine the worth of a cell l when it is over that cell. We also assume that each agent i is able to determine the total worth over its neighborhood N_δ^i. We denote this sensed worth by $C^i : \mathcal{A}_n^i \mapsto \mathbf{R}$ which is defined as $C^i(\alpha^i(n)) = \sum_{l \in \mathcal{A}_n^i} f(l)$. Thus, at each time step n, each agent i can determine the worth of its current cell $f(\alpha^i(n))$ and also the "cumulative" worth of its sensing neighborhood $C^i(\alpha^i(n))$. Note that other than its current location $\alpha^i(n)$, agent i can not determine the worth of other cells in $\mathcal{A}_n^i$, and $C^i(\alpha^i(n))$ is only a summation of the worth over $\mathcal{A}_n^i$. The coverage objective is to find the set of targets while the energy consumption is minimized. Technically speaking this means that the robot network has to maximize the total worth of the covered cells and to minimize the energy consumption due to movement.

Each agent i lays a flag at each cell once he observes that cell and the set of its flags Υ^i are detectable by other agents. The agents can detect these flags from the maximum distance of 2δ. By using this technique, the agents do not need to have access to each other's observation memories and the observed cells are locally detectable.

A limited communication setting is considered to let the agents fuse their information. In this setting, each agent i can communicate with its set of neighbor agents, defined as $\mathcal{N}_n^i = \{j \in \mathcal{I} \setminus i \; s.t. \; |\alpha^j(n) - \alpha^i(n)| \leq R_{com}\}$ where R_{com} is communication radius. At each time step n agents are picked pairwise to check if they are within each other's communication radius. Thus, all the neighbor robots have the chance to make a contact with other robots in their neighborhood and share their information.

A. Game Formulation

The first step to establish a game is to define an appropriate utility function for each agent. Considering the coverage objective in this work, the utility function should take the reward of covering worthwhile cells into account and should penalize the robots for the consumed energy. The energy consumption of agent i due to its motion is denoted by $E_{move}^i = K^i(|\alpha^i(n) - \alpha^i(n-1)|)$ where $K^i > 0$ is a coefficient and $\alpha^i(n-1)$ is the location of agent i at time step $n-1$. We now formulate the coverage problem as a game by defining the following utility function which is inspired by [8]:

$$u^i(\alpha(n), \alpha(n-1)) = \varrho^i[C^i(\alpha^i(n)) - C_n^i(\alpha^i(n))] - K^i(|\alpha^i(n) - \alpha^i(n-1)|), \quad (1)$$

where $C_n^i(\alpha^i(n)) = \sum_{j \in \mathcal{I} \setminus i} \sum_{l \in N_j^\delta \cap N_i^\delta} f(l)$ and ϱ^i is defined as:

$$\varrho^i = \begin{cases} 1 & : \alpha^i \notin \Upsilon^j, \; \forall j \in \mathcal{I} \setminus i \\ 0 & : otherwise \end{cases}$$

Note that $C^i(\alpha^i(n)) - C_n^i(\alpha^i(n))$ is actually the worth of the cells that are only covered by agent i. Parameter ϱ^i prevents players to pick their next actions from the cells that are previously observed by other agents.

Lemma 1: The coverage game $\mathcal{G} = \langle \mathcal{I}, \mathcal{L}, u^i \rangle$ is a potential game where the potential function is defined as:

$$\Phi(\alpha(n), \alpha(n-1)) = \sum_{j=1}^{N} u^j(\alpha(n), \alpha(n-1)). \quad (2)$$

(see Appendix I)

B. Learning Process

This section reviews Binary Log-Linear Learning (BLLL) as a model-based learning scheme to provide the iterative improvement in players' utilities in the MCC problem. A BLLL scheme, as used in [8], is a modified version of standard log-linear learning which allows players to have a time-varying action set (e.g. A_n^i) in a potential game. It can be shown that in a potential game, if all players adhere to BLLL, the joint action will stochastically converge to a steady state point. This stable joint action maximizes the potential under the following conditions [17]:

1. For any agent $i \in \mathcal{I}$ and for any action pairs $\alpha^i(1), \alpha^i(\eta) \in \mathcal{L}$, there exists a sequence of actions from $\alpha^i(1)$ to $\alpha^i(\eta)$ for all $n = 2, ..., \eta$ such that $\alpha^i(n) \in N_\delta^i(\alpha^i(n-1))$.

2. For any agent $i \in \mathcal{I}$ and for any action pair α_1^i and $\alpha^i{}_2$, α_1^i is in the available action set of α_2^i if and only if α_2^i is in the available action set of α_1^i.

It is easy to show that that the above conditions hold for the current problem and therefore stable joint actions in BLLL scheme maximize the potential function.

In BLLL, at each time step n, one random agent i is selected to change its action while other agents repeat their action (i.e. $\alpha^{-i}(n) = \alpha^{-i}(n-1)$). Without loss of generality we assume that $i = (n \ mod \ |\mathcal{I}|) + 1$. Thus, the agents do not need to all agree on one random player since such agreement needs a wide communication within the robot network. Next, agent i chooses a trial action α_T^i uniformly randomly from its available action set $\mathcal{A}_{n-1}^i$. Recall that in our case, the available action set for player i is the set of his neighbor cells $N_\delta^i(\alpha^i(n-1))$. Finally, according to the following Boltzmann sampling method, agent i either selects action α_T^i as its next action or remains on its previous action $\alpha^i(n-1)$:

$$P^i_{\alpha^i(n-1)}(n) = \frac{\exp(\frac{1}{\tau}u^i(\alpha(n-1), \alpha(n-2)))}{D_n^i(\alpha_T^i)}, \tag{3}$$

$$P^i_{\alpha_T^i}(n) = \frac{\exp(\frac{1}{\tau}u^i(\alpha_T^i, \alpha^{-i}(n), \alpha(n-1)))}{D_n^i(\alpha_T^i)}, \tag{4}$$

where $D_n^i(\alpha_T^i) = \exp(1/\tau \ \ u^i(\alpha(n-1), \alpha(n-2))) + \exp(1/\tau \ \ u^i(\alpha_T^i, \alpha^{-i}(n), \alpha(n-1)))$. $P^i_{\alpha^i(n-1)}(n)$ is the probability that agent i repeats its action, $P^i_{\alpha_T^i}(n)$ is the probability that agent i selects α_T^i as its next move, and the coefficient τ (often referred to as the temperature coefficient) specifies how likely it is that each agent i chooses a sub-optimal action with respect to its utility function. Note that each agent i must have a posteriori knowledge about $u^i(\alpha_T^i, \alpha^i(n-1))$ to be able to determine (3) and (4). Although it may be impossible to determine the actual value of $u^i(\alpha_T^i, \alpha^i(n-1))$, given the fact that some parts of the area may be unobserved, an estimation algorithm can be used to estimate the utility of the trial action α_T^i and to enable players to predict the probabilities in (3) and (4). In the following we first introduce the Gaussian model which we have considered to model the environment and then we move forward to the estimation algorithm.

C. Gaussian Mixture Model

Recall that each target is assumed to emit a signal that is detectable by the agents. It is reasonable to assume that the shape of this signal is a symmetric Gaussian function whose mean is on the target. Therefore, the worth distribution over the area is basically a Gaussian Mixture Model (GMM). A GMM is a probability density function defined as a weighted sum of a number of Gaussian functions

$$GMM = \sum_{j=1}^{M} \omega_j \ g(x|\mu_j, \Sigma_j), \tag{5}$$

where M is the number of components (i.e. targets), ω_j is the weight of jth component and

$$g(x|\mu_j, \Sigma_j) = \frac{1}{2\pi|\Sigma_j|^{1/2}} \ exp[\frac{-1}{2}(x-\mu_j)^T\Sigma_j^{-1}(x-\mu_j)]. \tag{6}$$

$g(x|\mu_j, \Sigma_j)$ is a two-variable Gaussian function where x is the continuous-valued location vector, μ_j is the mean vector (i.e. location of the target j) and Σ_j is the covariance matrix of component j. The weights have to satisfy $\sum_{j=1}^{M} \omega_j = 1$. We summarize GMM's parameters into the set $\lambda = \{\omega_j, \mu_j, \sigma_j | j = 1, ..., M\}$. Note that $f(l)$ can be determined by substituting l into the GMM, i.e.

$$f(l) = GMM\big|_{x=l} = \sum_{j=1}^{M} \omega_j \ g(l|\mu_j, \Sigma_j). \tag{7}$$

Therefore, by knowing (or estimating) the GMM, we are able to determine (or estimate) the worth distribution over the area.

D. Estimation Scheme

During the search task, the robots keep their observations of sensed regions in their memories. This aggregated memory can be used to form an estimation of the worth distribution which enables agents to take optimized actions once they encounter undiscovered areas. Let agent i's estimation of the mixture parameters be $\hat{\lambda}^i = \{\hat{\omega}_j^i, \hat{\mu}_j^i, \hat{\sigma}_j^i | j = 1, ..., M\}$. We then denote the estimation model by

$$\widehat{GMM} = \sum_{j=1}^{M} \hat{\omega}_j \ g(x|\hat{\mu}_j, \hat{\Sigma}_j). \tag{8}$$

Agent i's memorized observation vector at iteration n is a sequence of its sensed cells from time step 1 to n denoted by $O^i = \{O_1^i, O_2^i, O_3^i, ..., O_n^i\}$ where O_n^i is the corresponding coordinates of the sensed cell by agent i at time step n.

Expectation Maximization (EM) is an iterative algorithm used to find maximum likelihood parameters of a statistical model when there are missing data points from the model. The EM algorithm is able to determine the parameter set $\hat{\lambda}^i, \ i \in \mathcal{I}$

through the following iterative algorithm [18]:

$$\begin{aligned}
\hat{\omega}_j^i &= \frac{1}{n}\sum_{\eta=1}^{n} P^i(j|O_\eta^i, \hat{\lambda}^i),\\
\hat{\mu}_j^i &= \frac{\sum_{\eta=1}^{n} P^i(j|O_\eta^i, \hat{\lambda}^i) O_\eta^i}{\sum_{\eta=1}^{n} P^i(j|O_\eta^i, \hat{\lambda}^i)},\\
(\hat{\sigma}_j^i)^2 &= \frac{\sum_{\eta=1}^{n} P^i(j|O_\eta^i, \hat{\lambda}^i)(O_\eta^i - \hat{\mu_j^i})(O_\eta^i - \hat{\mu_j^i})^T}{\sum_{\eta=1}^{n} P^i(j|O_\eta^i, \hat{\lambda}^i)},\\
P^i(j|O_\eta^i, \hat{\lambda}^i) &= \frac{\hat{\omega_j^i}\ g(O_\eta^i|\hat{\mu_j^i}, \hat{\sigma_j^i})}{\sum_{k=1}^{M} \hat{\omega_k^i}\ g(O_\eta^i|\hat{\mu_k^i}, \hat{\sigma_k^i})}
\end{aligned} \tag{9}$$

where $P^i(j|O_\eta^i, \hat{\lambda}^i)$ is often referred to as posteriori probability of the observation vector O_η^i for the jth component of Gaussian distribution. Through (9), given the current estimated parameters, the EM algorithm estimates the likelihood that each data point belongs to each component. Next, the algorithm maximizes this likelihood with respect to each parameter to find a new set of parameters.

Recall that O^i is the sequence of agent i's observed coordinates within the area. Since EM only considers O^i as the input it actually disregards the sensed worth. This issue is pointed out in [8] and the proposed solution as introduced in [19] is to repeat the iterative algorithm for m times in worthwhile cells. The parameter m is chosen as:

$$m = \begin{cases} 1 + V\ round(\dfrac{f(l)}{f_{mode}}) & : f(l) \geq f_{mode} \\ 1 & : f(l) < f_{mode} \end{cases}$$

where $f(l)$ is the worth at coordinate l, f_{mode} is the threshold above which the observation is considered worthwhile for EM repetition and V is the correction factor which regulates the number of algorithm repetitions for the worthy cells. The algorithm will run once for cells with a worth value of less than the threshold.

The variation rate of the estimation parameters $\hat{\lambda}^i$ is relatively low due to the rate of update of the observation vector and the nature of the EM algorithm. The case of slow variation of Gaussian distribution is investigated in [20]. By using BLLL and assuming a slow changing distribution, [20] showed that the agents' estimation error $|\hat{\lambda}(n) - \lambda(n)|$ decreases as the observations vector expands; consequently, the agents will stochastically converge to a Nash equilibrium. If we assume that the number of the targets (M) is a known parameter, we can use the iterative algorithm discussed in this section. We call this algorithm EM_N and it is the algorithm that is used in [8].

III. Merge and Split Mechanism

Recall that the agents do not have any prior knowledge either about the worth distribution function over the area or about the number of targets. As we discussed in Section II, we need to know the number of the targets M in (9) in order to use the EM algorithm. In this section we propose a modified EM algorithm which lets the robots estimate the number of the targets.

The Akaike information criterion (AIC) introduced in [21] is considered as the main verification method to help the agents select the best guess for the number of the targets. The Akaike information criterion is a measure of the quality of a distribution model for a set of data points. Given a collection of models for a data set, AIC estimates the relative quality of each model by

$$AIC = 2k - 2ln(L), \tag{10}$$

where L is the maximized value of the likelihood and k is the number of estimated parameters in the model. The Akaike criterion considers the goodness of the fit as the likelihood function L and penalty of complexity of the model as k (i.e. the number of model parameters). Given a collection of statistical models, the one with minimum AIC value is the best model [21].

Let the estimated number of the targets at time step n each robot i be denoted by $\hat{M}^i(n)$. In our proposed algorithm, after each n_{AIC} time steps, each agent $i \in \mathcal{I}$ randomly picks a number T_{AIC} from the set $\mathcal{M}(n_{AIC}) = \{\hat{M}^i(n_{AIC}) + 1, \hat{M}^i(n_{AIC}) - 1\} \subset \mathbf{N}$. If $\hat{M}^i(n_{AIC}) - 1 = 0$ then $\mathcal{M}(n_{AIC})$ reduces to $\{\hat{M}^i(n_{AIC}) + 1\}$. Next, player i decides between its current estimated Gaussian component number $\hat{M}^i(n_{AIC})$ and T_{AIC} according to the following probabilities:

$$P^i_{\hat{M}^i(n_{AIC})} = \frac{\exp(\frac{1}{\tau} IAIC(\hat{M}^i(n_{AIC})))}{\exp(\frac{1}{\tau} IAIC(\hat{M}^i(n_{AIC}))) + \exp(\frac{1}{\tau} IAIC(T_{AIC}))}, \tag{11}$$

$$P^i_{T_{AIC}} = \frac{\exp(\frac{1}{\tau} IAIC(T_{AIC}))}{\exp(\frac{1}{\tau} IAIC(\hat{M}^i(n_{AIC}))) + \exp(\frac{1}{\tau} IAIC(T_{AIC}))}, \tag{12}$$

where $P^i_{\hat{M}^i(n_{AIC})}$ is the probability that player i keeps its current estimation $\hat{M}^i(n_{AIC})$ and $P^i_{T_{AIC}}$ is the probability that player i chooses T_{AIC} as its estimation of the number of the components. $IAIC(\hat{M}^i(n_{AIC}))$ is the inverse AIC value for agent i's estimation distribution when agent i's estimated number of the components is $\hat{M}^i(n_{AIC})$ and $IAIC(T_{AIC})$ is the inverse AIC value when agent i's estimation is T_{AIC}.

Since the number of the components changes every n_{AIC} iterations, the next step is to determine an appropriate method for merging and splitting Gaussian components. The method proposed in [22] incorporates the split and merge operations into the EM algorithm for GMM estimations. Moreover, efficient criteria are proposed in [22] to decide which components have to be merged or split. In the following the merge and split algorithms are briefly discussed.

A. Merge

In order to reduce the number of components in a Gaussian mixture, some of the components should be merged together. However, an effective criterion is needed to pick the optimal pairs.

a) Criterion: The posteriori probability of a data point gives a good estimation about to which Gaussian component that

data point belongs. It can be concluded that if for many data points, the posteriori probabilities are almost equal for two components, then these two components can be merged. To mathematically implement this, the following criterion for jth and j'th Gaussian components is proposed in [22]:

$$J_{merge}(j,j';\hat{\lambda}) = \mathbf{P}_j(\hat{\lambda})^T \mathbf{P}_{j'}(\hat{\lambda}), \tag{13}$$

where $\mathbf{P}_j(\hat{\lambda}) = (P(j|O_1,\hat{\lambda}), P(j|O_2,\hat{\lambda}), ..., P(j|O_n,\hat{\lambda}))^T$ is a n-dimensional vector consisting of posteriori probabilities of all data points for jth Gaussian component. The criterion $J_{merge}(j,j';\hat{\lambda})$ must be calculated for all possible pairs and the pair with the largest value is selected to be merged. As the next step, the selected pairs will be merged by a modified EM algorithm.

b) Merging Procedure: In order to merge two Gaussian components, the distribution model parameters must be re-estimated. A modified EM algorithm is investigated in [22] that re-estimates Gaussian parameters corresponding to former distribution parameters ($\hat{\lambda}$). If the merged Gaussian from the pair of j and j' is denoted by j'' then the initial parameters for the merged Gaussian would be [22]

$$\begin{aligned} \omega^0_{j''} &= \omega_j + \omega_{j'}, \\ \mu^0_{j''} &= \frac{\omega_j \mu_j + \omega_{j'} \mu_{j'}}{\omega_j + \omega_{j'}}, \\ \Sigma^0_{j''} &= \frac{\omega_j \Sigma_j + \omega_{j'} \Sigma_{j'}}{\omega_j + \omega_{j'}}. \end{aligned} \tag{14}$$

After calculating the initial values, an EM iterative algorithm will run to re-estimate the distribution parameters. The main steps are the same as (9) except the posteriori probability [22]:

$$P(j''|O_\eta,\hat{\lambda}) = \frac{\hat{\omega}_{j''}\ g(O_\eta|\hat{\mu}_{j''},\hat{\sigma}_{j''})\ \sum_{k=j,j'} P(k|O_\eta,\hat{\lambda})}{\sum_{k=j''} \hat{\omega}_k\ g(O_\tau|\hat{\mu}_k,\hat{\sigma}_k)}. \tag{15}$$

By using this modified EM algorithm, the parameters of j''th Gaussian are re-estimated without affecting the other Gaussian components. The initial parameter values calculated by (14) are often poor. Hence, the newly generated Guassians should be first trained by fixing the other Gaussians through the modified EM.

B. Split

In case of a need to increase the number of components in the estimation distribution, we use the split algorithm to split one or more Gaussians. As for the merging process, an appropriate criterion is necessary as well as an efficient splitting method.

a) Criterion: As the split criterion of kth component, the local Kullback-Leibler divergence is proposed in [22] as:

$$J_{split}(k;\hat{\lambda}) = \int p_k(x,\hat{\lambda}) log(\frac{p_k(x,\hat{\lambda})}{g(x|\hat{\mu_k},\hat{\Sigma_k})})\ dx, \tag{16}$$

where $p_k(x,\hat{\lambda})$ is the local data density around kth component and is defined as:

$$p_k(x,\hat{\lambda}) = \frac{\sum_{m=1}^{n}\ \delta(x-x_m) P(k|x_m,\hat{\lambda})}{\sum_{m=1}^{n} P(k|x_m,\hat{\lambda})}. \tag{17}$$

Equation (16) actually represents the distance between two Gaussian components where the kth Gaussian is characterized by $\hat{\mu_k}$ and $\hat{\Sigma_k}$. As for the merge criterion, $J_{split}(k,\hat{\lambda})$ must be applied on all candidates and the one with the largest criterion value will be selected to be split; the larger the $J_{split}(k,\hat{\lambda})$, the worse the local density, the more appropriate candidate to split.

b) Splitting Procedure: As in the merging process, a modified EM algorithm is used to re-estimate the Gaussians parameters after the initial process of split. If the split candidate is the kth Gaussian component and the two resulting Gaussians are denoted by j' and k' the initial conditions to start the modified EM algorithm are calculated as follows:

$$\begin{aligned} \omega^0_{j'} &= \omega^0_{k'} = \frac{1}{2}\ \omega_k, \\ \Sigma^0_{j'} &= \Sigma^0_{k'} = det(\Sigma_k)^{1/d}\ I_d, \end{aligned} \tag{18}$$

where I_d is the d-dimensional unit matrix and d is the dimension of Gaussian function $g(x|\mu_k,\Sigma_k)$. The mean vectors $\mu^0_{j'}$ and $\mu^0_{k'}$ are determined by applying random perturbation vector ϵ_m, $m = 1,2$ on μ_k as $\mu^0_{j'} = \mu_k + \epsilon_1$ and $\mu^0_{k'} = \mu_k + \epsilon_2$ where $||\epsilon_m|| \ll ||\mu_k||$ and $\epsilon_1 \neq \epsilon_2$. The parameters re-estimation for j' and k' is done using a modified EM algorithm similar to the merge EM algorithm where the modified posteriori probability is:

$$P(m'|O_\eta,\hat{\lambda}) = \frac{\hat{\omega}_{m'}\ g(O_\eta|\hat{\mu}_{m\prime},\hat{\sigma}_{m'})\ \sum_{l=k} P(l|O_\eta,\hat{\lambda})}{\sum_{l=j',k'} \hat{\omega}_l\ g(O_\eta|\hat{\mu}_l,\hat{\sigma}_l)}, \tag{19}$$

where $m' = j', k'$. The parameters of j' and k' are re-estimated without affecting other Gaussians. By means of AIC and split-merge technique, the agents are able to estimate the number of targets. We call this new integrated algorithm EM_SM.

IV. Communication

In this section a mutual information-sharing mechanism is proposed to improve the robot network knowledge about the environment. The communication cost between agent i and agent i' is defined as:

$$J_{com} = K_{com}\ D_{trans}(i,i',n)\ (|\alpha^i(n) - \alpha^{i'}(n)|), \tag{20}$$

where K_{com} is the regulation coefficient and $D_{trans}(i,i',n)$ is the volume of transmitted data between agent i and agent i' at time step n. This communication cost is modeled as the power consumption needed to transmit data between the robot network nodes in a wireless communication setting. At each iteration, agents that are in each other's communication radius (R_{com}) are able to contact each other. In order to make a more efficient communication mechanism, a robot pair (i and i') are allowed to communicate with each other if the following conditions are satisfied:

1. Agent i is within agent i' communication range.

2. One of the agents (i) has a larger cumulative covered worth (with a certain ratio) than the other member of the

communication pair (i'). In other words:

$$\sum_{\eta=1}^{n} f(O_{\eta}^{i}) \geq K'_{com} \sum_{\eta=1}^{n} f(O_{\eta}^{i'}), \tag{21}$$

where K'_{com} is a pre-set coefficient (ratio).

3. The length of the observation vector of agent i is larger than that of agent i' with an specified ratio

$$num^{i} \geq K''_{com}\ num^{i'}, \tag{22}$$

where K''_{com} is the ratio coefficient and num^{l} is the frequency with which agent l selected a trial action up to time step n where $l = i, i'$ (i.e. number of unique observations of agent l). Clearly, based on these conditions we may assume that agent i is more experienced than agent i' in the network and a communication between i and i' could improve the knowledge of agent i' about the area which results in an enhancement in the overall performance of the whole system. During communication, agent i' that is inexperienced and possesses a lower data volume, transmits its observation vector along with its distribution estimation parameters (i.e. $\hat{\omega}^{i'}$,$\hat{\mu}^{i'}$ and $\hat{\Sigma}^{i'}$) to agent i. Hence, in this scenario, agent i carries out the computation burden. The computation consists of re-estimation of the Gaussian distribution based on the observation vectors of agent i and agent i' together; the aggregated vector of O^{i} and $O^{i'}$ is denoted by $O^{i+i'}$. The re-estimation algorithm is basically an EM algorithm with the following initial condition:

$$\hat{\Theta}_{0}^{r} = \frac{E^{i}\hat{\Theta}^{i} + E^{i'}\hat{\Theta}^{i'}}{E^{i} + E^{i'}}, \tag{23}$$

where $\hat{\Theta}$ could be $\hat{\omega}$, $\hat{\mu}$ or $\hat{\sigma}$, and E^{l} is defined as:

$$E^{l} = num^{l} \sum_{\eta=1}^{n} f(O_{\eta}^{l}),\ l = i, i'. \tag{24}$$

Equation (23) actually represents a weighted average of the estimation parameters of agent i and agent i' which takes the experience of each agent into account. In order to re-estimate the Gaussian parameters an EM algorithm is used with iterative steps as discussed in (9). However it should be noted that the initial conditions are calculated based on (23) and the training vector which is fed into the algorithm is $O^{i+i'}$. Finally, the resulting parameters $\hat{\omega}^{r}$, $\hat{\mu}^{r}$ and $\hat{\sigma}^{r}$ will replace old estimation parameters of both agent i and agent i'. In this study, the communication cost is not implemented in the agents' utility function and is only considered as a performance comparison mean between different methods.

V. Simulation and Discussion

In this section we provide a numerical example to verify the performance of the proposed algorithm. A 40×40 square area is considered as the environment. The Gaussian distribution in this area is chosen randomly with different but reasonable weight, mean and covariance values (Fig. 1). The number of the targets (Gaussian components) is between 1 to 5 and the robot network has no prior knowledge about this number.

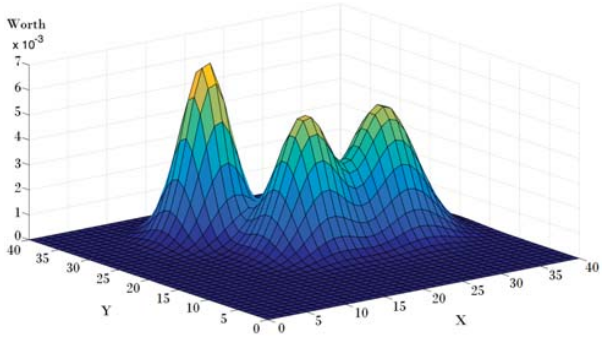

Fig. 1 Worth distribution over the area

A group of five robots ($|\mathcal{I}| = 5$) scatters in the area to maximize their utility function. Note that the number of the targets could be higher than the number of the robots but in that case the robots need more time to explore the area and gather enough data.

The simulation parameters are chosen as $K^{i} = 3 \times 10^{-5}$ for $i = 1, ..., |\mathcal{I}|$, $\tau = 5 \times 10^{-4}$, $f_{mode} = 10^{-5}$, $V = 0.1$, $\delta = 1.5$, $n_{AIC} = 100$, $K_{com} = 1$, $K'_{com} = K''_{com} = 1.5$ and $R_{com} = 5$. An increase in K^{i} results in agents not to significantly deviate from their current location. Parameter τ controls how rational the robots are, in their learning process. The importance of worth distribution from agents' point of view is regulated by f_{mode} and V. The parameter n_{AIC} controls the split and merge process and its computation burden on the agents. Communication parameters K_{com}, K'_{com}, K''_{com} and R_{com} can be regulated according to the on-board power limitations. However, it is obvious that the wider the communication the more accurate the estimation.

The worth of the covered area using a normal EM algorithm (EM_N, Section II) and our proposed algorithm (EM_SM, Section III) is illustrated in Fig. 2. In EM_N, and as in [8], the agents do not have a prior knowledge about the location of the targets but the number of the targets is known to the agents. In EM_SM, the number of the targets is unknown. As in Fig. 2, EM_SM outperformed EM_N in terms of the coverage worth. However, the convergence rate of EM_N is slightly better. The fluctuation in real-time covered worth is due to the robots' exploration attempts.

Fig. 3 shows the final configuration of the robots for both EM_N and EM_SM. In EM_N, the robots failed to locate all the targets while in EM_SM all the targets are successfully located. This actually explains the difference between the covered worth in EM_N and EM_SM in Fig. 2.

The real-time estimated value of the number of Gaussian components in EM_SM is shown in Fig. 4. As in Fig. 4, the estimated number of components oscillates between 3, 4 and 5 where the initial value is a random number between 1 and 5.

Unlike most of previous studies, in which the mobile sensors start their searching task from a completely random or from a pre-specified initial locations, here, an optimized and dynamic initialization is proposed. In order to initialize the robots, a 10×10 square box at the middle of the environment is defined

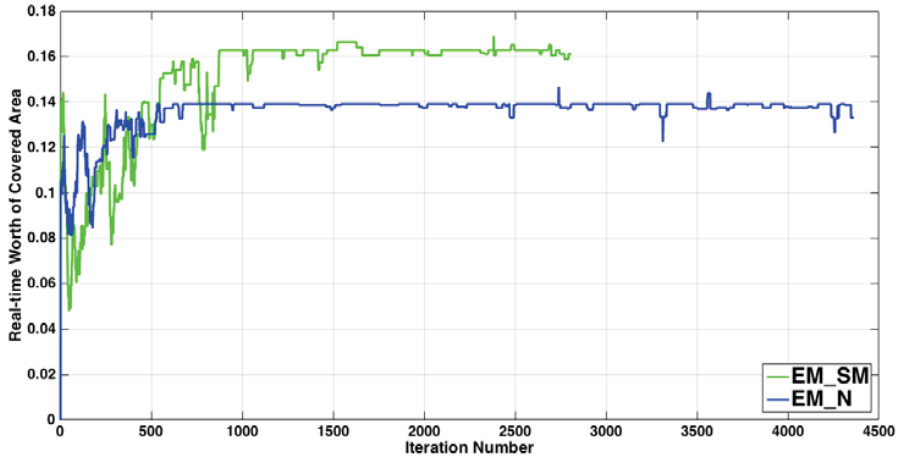

Fig. 2 Evolution of the real-time covered worth

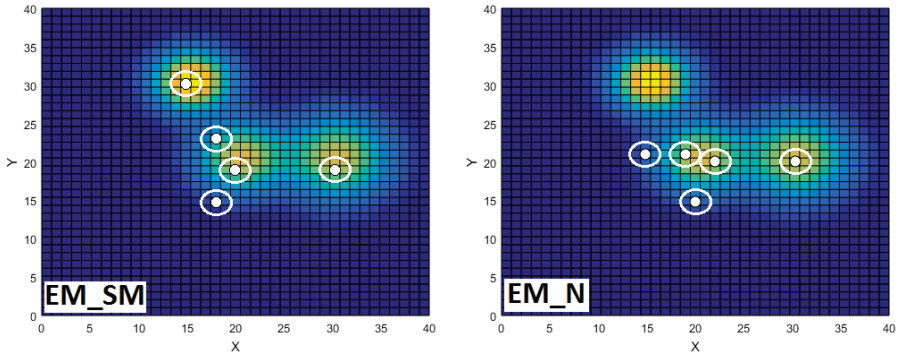

Fig. 3 Final configuration of the robots in EM_N and EM_SM

where the agents pick their initial locations randomly inside this box. Agents are not allowed to exceed the borders of this square while they are searching for the targets. However, the square is continuously expanding with respect to time until it reaches to the size of the mission space (40×40).

The slope of trending of total covered worth is considered as the main tool to optimize the expansion rate. Fig. 5 demonstrates the expansion concept in which S1 is the slope of the trend line of total cumulative worth versus total time and S2 is the slope of trend line of last $\beta\%$ of cumulative worth versus last $\beta\%$ of time. If S2 is lower than S1 then, it means that the worthwhile points of the area is covered (or the area's worth is low) so the expansion rate of searching area would be increased (β is currently 5 and in the case of expansion, the area expansion rate will be multiplied by S1/S2). On the other hand, if S1 is lower than S2, which means the area is worthwhile, the expansion rate would be decreased (multiplied by S1/S2). The expansion algorithm will start after 20 iterations and will stop if the borders reach the 40×40.

The communication cost of both methods (EM_N and EM_SM) is shown in Fig. 6. In EM_SM, agents attempted to communicate with each other 45 times while in EM_N it was 54. One interpretation is that all the agents gained a uniform average experience through EM_SM; so the number of communication attempts is less than EM_N. Furthermore, the amount of data bits transferred in EM_SM is less than EM_N upto 45th attempt which means the agents consume less energy to send data bits in EM_SM while maintained their level of collaboration (data sharing) in an acceptable range.

VI. Conclusions

In this paper, a mobile sensor coverage problem is investigated in which a finite number of robots seek to cover the areas with the highest probability of existence of a set of targets. The robots have no familiarity with the area as they do not have any information either about the location or the number of the targets. A state-based potential game is formulated to relate the robots together and to control the robots' actions. The reward of sensing valuable areas is considered in the utility function as well as the penalty of energy consumption due to the agents' movement. Update of agents' action profile is based on BLLL in which the agents need to know an estimation of the outcome of their future actions. Hence, an estimation algorithm has been utilized to assist the agents in anticipating the probability of the targets' existence in undiscovered areas. A modified EM algorithm is introduced to estimate the number of the targets as well as other parameters of the probability distribution. Furthermore, a dynamic and restricted search zone is proposed to force the agents to remain in worthwhile areas as much as possible.

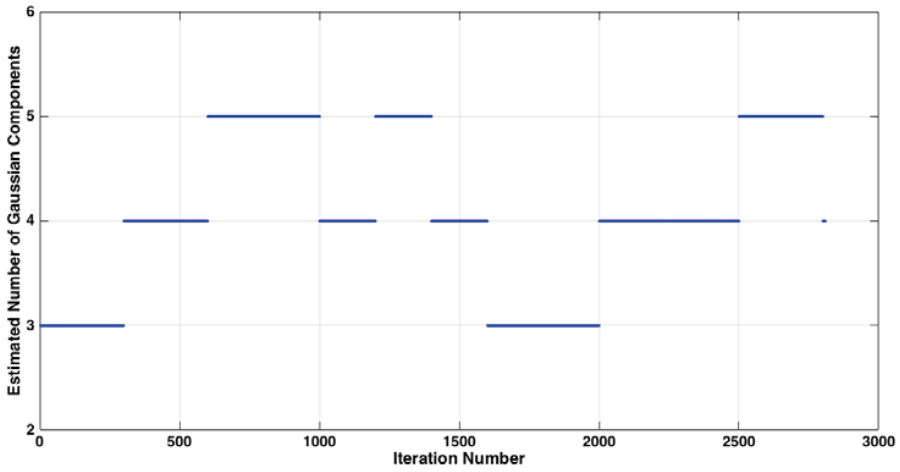

Fig. 4 Online estimation of number of the Gaussian components

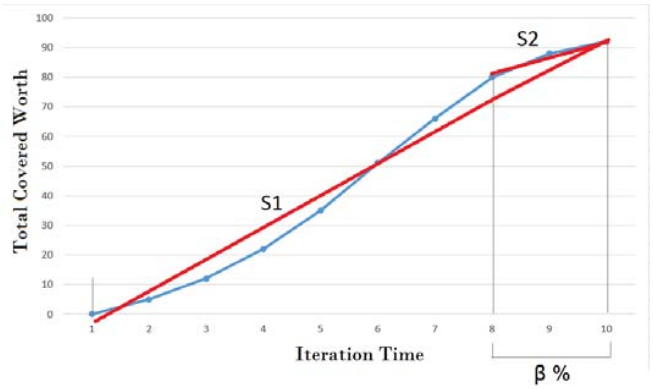

Fig. 5 Online expansion of initialization square

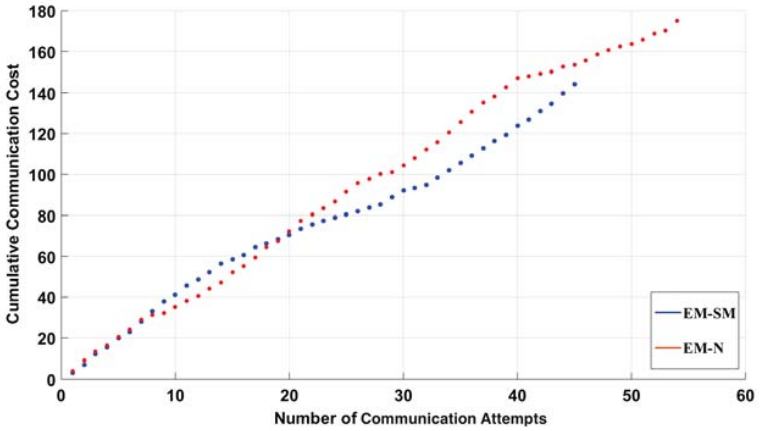

Fig. 6 Cumulative communication cost

Lastly, we discussed a communication scheme that allows the agents to fuse their knowledge of the unknown environment which presumably improves system's overall performance.

APPENDIX

Proof of Lemma 1: We have to show that for any agent $i \in \mathcal{I}$, for every $\alpha^{-i}(n) \in \mathcal{A}^{-i}$:

$$\Phi(\alpha_2^i(n), \alpha^{-i}(n), \alpha(n-1)) - \Phi(\alpha_1^i(n), \alpha^{-i}(n), \alpha(n-1)) = u^i(\alpha_2^i(n), \alpha^{-i}(n), \alpha(n-1)) - u^i(\alpha_1^i(n), \alpha^{-i}(n), \alpha(n-1)), \tag{25}$$

By (2) we have:

$$\Phi(\alpha_2^i(n), \alpha^{-i}(n), \alpha(n-1)) = [\Sigma_{j=1,j\neq i}^N u^j(\alpha^j(n), \alpha^j(n-1))] + u^i(\alpha_2^i(n), \alpha^i(n-1)) = \Big[\Sigma_{j=1,j\neq i}^N \varrho^j[C^j(\alpha^j(n)) - C_n^j(\alpha^j(n))] - \Sigma_{j=1,j\neq i}^N [K^i(|\alpha^j(n) - \alpha^j(n-1)|)]\Big] + \Big[\varrho^i[C^i(\alpha_2^i(n)) - C_n^i(\alpha_2^i(n))] - K^i(|\alpha_2^i(n) - \alpha^i(n-1)|)\Big],$$

and

$$\Phi(\alpha_1^i(n), \alpha^{-i}(n), \alpha(n-1)) = [\Sigma_{j=1,j\neq i}^N u^j(\alpha^j(n), \alpha^j(n-1))] + u^i(\alpha_1^i(n), \alpha^i(n-1)) = \Big[\Sigma_{j=1,j\neq i}^N \varrho^j[C^j(\alpha^j(n)) - C_n^j(\alpha^j(n))] - \Sigma_{j=1,j\neq i}^N [K^i(|\alpha^j(n) - \alpha^j(n-1)|)]\Big] + \Big[\varrho^i[C^i(\alpha_1^i(n)) - C_n^i(\alpha_1^i(n))] - K^i(|\alpha_1^i(n) - \alpha^i(n-1)|)\Big],$$

Hence,

$$\Phi(\alpha_2^i(n), \alpha^{-i}(n), \alpha(n-1)) - \Phi(\alpha_1^i(n), \alpha^{-i}(n), \alpha(n-1)) = \varrho^i[C^i(\alpha_2^i(n)) - C_n^i(\alpha_2^i(n))] - K^i(|\alpha_2^i(n) - \alpha^i(n-1)|) - \varrho^i[C^i(\alpha_1^i(n)) - C_n^i(\alpha_1^i(n))] - K^i(|\alpha_1^i(n) - \alpha^i(n-1)|) = u^i(\alpha_2^i(n), \alpha^{-i}(n), \alpha(n-1)) - u^i(\alpha_1^i(n), \alpha^{-i}(n), \alpha(n-1)),$$

□

REFERENCES

[1] S. Dhillon, K. Chakrabarty, S. Iyengar, "Sensor Placement for Grid Coverage under Imprecise Detections", in Proc. Intl. Conf. on Information Fusion, pp. 1571-1587, 2002.
[2] D. Ucinski, "Measurement Optimization for Parameter Estimation in Distributed Systems", CRC Press, 1999.
[3] Li, W., Cassandras, C.G., "Distributed Cooperative coverage Control of Sensor Networks", 44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC '05), pp. 2542-2547, 2005.
[4] J. R. Spletzer and C. J. Taylor, "Dynamic Sensor Planning and Control for Optimally Tracking Targets", International Journal of Robotics Research, vol. 22, no. 1, pp. 7-20, 2003.
[5] T. Nakamoto, H. Ishida, and T. Moriizumi, "Active Odor Sensing System", in Proc. Intl. Symposium on Industrial Electronics, vol. 1, pp. SS128-SS133, 1997.
[6] J.R. Frost and L.D. Stone, "Review of Search Theory: Advances and Applications to Search and Rescue Decision Support", Technical Report CG-D-15-01, U.S. Coast Guard Research and Development Center, Gronton, CT, 2001.
[7] T. Heng, Y. Kuno, and Y. Shirai, "Active Sensor Fusion for Collision Avoidance", in Proc of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, vol. 3, pp. 1244-1249, 1997.
[8] Rahili, S., Ren, W., "Game Theory Control Solution for Sensor Coverage Problem in Unknown Environment", 53rd IEEE Conference on Decision and Control (CDC), pp.1173-1178, 2014.
[9] W. Yatao and L. Pavel,"A Modified Q-Learning Algorithm for Potential Games", Proceedings of the 19th IFAC World Congress, vol.19, pp. 8710-8718, 2014.
[10] J. Marden and A. Wierman, "Distributed Welfare Games with Applications to Sensor Coverage", 47th IEEE Conference on Decision and Control, pp. 1708-1713, 2008.
[11] Pavel, L.,"Game Theory and Evolutionary Games", ECE1657, University of Toronto, Toronto, 2014.
[12] H. Bayram and H. Bozma, "Multi-robot Communication Network Topology via Centralized Pairwise Games", 2013 IEEE International Conference on Robotics and Automation, 2013.
[13] Chung, T.H., Gupta, V., Burdick, J.W., Murray, R.M.,"On a Decentralized Active Sensing Strategy using Mobile Sensor Platforms in a Network", 43rd IEEE Conference on Decision and Control (CDC), pp.1914-1919, Vol. 2, 2004.
[14] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage Control for Mobile Sensing Networks", IEEE Transactions on Robotics and Automation, vol. 20, no. 2, pp. 243-255, 2004.
[15] J. Cortes, S. Martinez and F. Bullo, "Spatially-distributed Coverage Optimization and Control with Limited-range Interactions", ESAIM: Control, Optimisation and Calculus of Variations, vol. 11, no. 4, pp. 691-719, 2005.
[16] A. Kwok and S. Martinez, "Deployment Algorithms for A Power-constrained Mobile Sensor Network", IEEE International Conference on Robotics and Automation, 2008.
[17] J. R. Marden and J. S. Shamma, "Revisiting Log-linear Learning: Asynchrony, Completeness and Payoff-based Implementation", Games and Economic Behavior, vol. 75, no. 2, pp. 788-808, 2012.
[18] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via The EM Algorithm", Journal of the Royal Statistical Society Series B, vol. 39, no. 1, pp. 1-38, 1977.
[19] V. Lakshmanan and J. S. Kain, "A Gaussian Mixture Model Approach to Forecast Verification", Weather and Forecasting, vol. 25, pp. 908-920, 2010.
[20] Y. Lim and J. S. Shamma, "Robustness of Stochastic Stability in Game Theoretic Learning", ACC, pp. 6145-6150, 2013.
[21] Akaike, H., "A New Look at The Statistical Model Identification", IEEE Transactions on Automatic Control, vol.19, no.6, pp. 716-723, 1974
[22] N. Ueda, R. Nakano, Z. Ghahramani and G. Hinton, "Split and Merge EM Algorithm for Improving Gaussian Mixture Density Estimates", The Journal of VLSI Signal Processing, vol. 26, no. 12, pp. 133-140, 2000.