Decentralized Multi-target Search and Coverage Using Hyperparameter Consensus

Haruki Nishimura – 06045657
AA277 Multi-robot Control, Communication and Sensing
Prof. Mac Schwager

March 10, 2016

Abstract

We present a linear-consensus-based algorithm for estimating the unknown positions of multiple targets using a team of sensing agents. Our approach integrates the hyperparameter consensus protocol with the conventional coverage cost function to deploy the agents and minimize the expected minimum distances from the targets. The simultaneous learning and control algorithms are provided for the one-target and the multi-target cases under several assumptions on the observation model. We empirically demonstrate via a set of simulations that our heuristic method appropriately balances exploration and exploitation in the target search and realizes a configuration to persistently monitor the targets.

1 Introduction

The field of multi-agent sensor networks has recently obtained extensive attention of researchers. The potential applications of multi-agent networks include environmental monitoring, cooperative exploration, and search and rescue. One of the challenging tasks for multi-agent systems is to provide coverage of a certain environment to meet the demands of users. This problem is known as the coverage control problem, which can be classified into two subcategories in general. The first category is the static coverage control, which aims to optimally scatter a set of sensors in an environment to maximize the service quality as a whole network. The other is the dynamic coverage control, in which the sensors need to keep moving in the environment to maintain a prescribed coverage level. For the static coverage control, one well-studied approach is to partition the environment according to the positions of agents and drive each agent towards the centroid of the partitioned space where it is involved. This partitioning is known as the Voronoi tessellation and is first applied to the distributed coverage control in [1]. Various extensions and modifications are presented thereafter, such as [2][3].

Although these Voronoi based coverage control methods have made significant contributions, they generally require prior knowledge about the information distribution function, which represents the state of the environment and assigns relative importance to each point in it. In order to relax this assumption, Schwager et al. [4] present a decentralized, adaptive coverage controller to achieve optimal static coverage while learning the distribution function simultaneously. This type of simultaneous coverage and learning task has also been studied in the context of the dynamic coverage problem in [5][6]. All of the aforementioned three works employ control-theory-based methods to estimate the distribution function.

On the other hand, many multi-agent sensing problems, including the simultaneous coverage and learning mentioned above, can be formulated as a parallelization of machine learning algorithms. One can think of two different classes of these parallel learning problems: those with a central information fusion processer and those without one. Capmbell and How [7] refer to these problems as distributed learning and decentralized learning, respectively. For example, distributed classification and regression in sensor networks discussed in [8] falls into the first category. Decentralized learning seems to have gained much more attention from researchers, however. Many decentralized approaches involve agents forming a consensus on the model parameters either in the maximum-likelihood framework [9] or in the Bayesian scheme [10][11][12].

In the present work we reformulate the coverage control problem with an unknown distribution as the simultaneous target search and coverage problem. The main objective of this paper is to reinterpret the coverage cost function as a statistical measure and incorporate the recursive Bayesian filter into the coverage controller to estimate the positions of the multiple static targets. We design a decentralized algorithm which appropriately deploys the agents to persistently monitor the targets. Particularly, we focus on the hyperparameter consensus algorithm introduced by Fraser et al. [12] and show that their approach can be extended to the decentralized multi-target search and coverage problem under certain assumptions. The proposed framework has the potential to perform well in practical situations. For instance, our method could be applied to a team of unmanned marine vehicles searching for unconfirmed underwater volcanoes using onboard sonar sensors.

The rest of the paper is organized as follows. In section 2, we formulate the problem and derive the control law for the coverage deployment. In section 3, we impose several assumptions on our model and adapt the hyperparameter consensus protocol for belief synchronization among the agents. The integrated algorithms for the one-target case and the multi-target case are presented in section 4. Finally, we demonstrate the proposed algorithms with simulations in section 5, and provide discussions in section 6.

2 Problem Formulation and Control Law

Let $Q \subset \mathbb{R}^2$ be a convex and bounded domain. The N agents deployed in this domain are denoted by $p_i \in \mathbb{R}^2 \ \ \forall i \in \{1, \ldots, N\}$. The number of static targets K in the domain is known to all the agents, but the true positions $x_k \in \mathbb{R}^2, \forall k \in \{1, 2, \ldots, K\}$ are unknown. Instead, each agent can use its onboard sensors to make measurements and have a belief about the target positions in the form of a probability density function $\phi(q)$, where $q \in Q$ and $\int_Q \phi(q)dq = 1$. This function provides the probability density that a target is positioned at q.

Given this probability distribution, we are interested in minimizing the expected squared euclidean distance between the targets and the agents which are already closest to them. Formally,

$$\mathbb{E}_\phi \left[\min ||q - p_i||^2\right]. \tag{1}$$

This expectation takes the form

$$\mathcal{H}(p_1, p_2, \ldots, p_N) = \int_Q (\min ||q - p_i||^2)\phi(q)dq, \tag{2}$$

which can be interpreted as a cost function of ϕ and $(p_1, \ldots, p_N)$. In the static target detection scenario, one can think of $||q - p_i||^2$ as a surveillance cost or a motion cost for the agent p_i to deal with a potential state change of the target in future.

We need to further decompose this cost function in order to minimize it in a distributed fashion. Indeed, $\mathcal{H}$ can be decoupled as follows.

$$\mathcal{H}(p_1, \ldots, p_N) = \sum_{i=1}^{N} \int_{V_i} ||q - p_i||^2 \phi(q)dq. \tag{3}$$

V_i in (3) is called the Voronoi region of p_i, which is a subset of Q such that all the points in it is closer to p_i than to all other p_js.

$$V_i = \{q \in Q \mid ||q - p_i|| \leq ||q - p_j|| \ \forall j \neq i\} \tag{4}$$

V_i is bounded by the bisecting normal hyperplanes between p_i and its neighboring agents or the Voronoi neighbors. The computation of Voronoi regions has been extensively studied in the literature [13][14]. Furthermore, [1] proposes a distributed control algorithm to minimize (3) by gradient descent.

$$\dot{p_i} = -\frac{\partial \mathcal{H}}{\partial p_i} \tag{5}$$

$$\frac{\partial \mathcal{H}}{\partial p_i} = \eta(p_i - C_{V_i}), \tag{6}$$

where η is the control gain and C_{V_i} represents the centroid of the Voronoi region:

$$C_{V_i} = \frac{1}{\int_{V_i} \phi(q)dq} \int_{V_i} q\phi(q)dq. \tag{7}$$

From a geometrical perspective, this control law drives each agent towards the centroid of its Voronoi region. The positions of the agents are proven to converge to a centroidal Voronoi configuration, which corresponds to an extremum point of the cost function (3). The reader is referred to [1] for the complete proof.

Although this continuous algorithm works well in practice, we will modify it to consider the discrete time setting for implementation in this paper. The position of the agent p_i at time $t+1$ is given by p_i and C_{V_i} at time t.

$$p_i[t+1] = p_i[t] + \eta(C_{V_i}[t] - p_i[t])\Delta t, \tag{8}$$

where Δt is the timestep per iteration. This is a hybrid method of [1] and the Lloyd's algorithm [15].

3 Observation Model and Belief Synchronization

So far we have assumed that the belief about the target positions have been computed and synchronized among all the agents. However, synchronization of the belief can be challenging due to the possibly high dimensional belief space and the communication constraints. sharing the same belief among the agents is critical to the control algorithm we derived in the previous section since the computation of Voronoi regions depends on the consistency of $\phi_i(q) = \phi(q) \;\; \forall i \in \{1, \ldots, N\}$. Therefore, the quality of the belief uniformity is crucial to our approach. In this section we will discuss a linear-consensus-based method to realize the belief synchronization.

First, we will define the observation model. We impose the following assumptions in the present work.

Assumption 1 (Linear Observation Model) *The observation is linear and subject to a Gaussian noise v with zero mean and a known covariance R. Let z_{ki} denote a measurement of the target x_k taken by the agent p_i. Then z_{ki} is given by the formulae below.*

$$\begin{aligned} z_{ki} &= x_k + v \qquad (9) \\ v &\sim \mathcal{N}(0, R). \qquad (10) \end{aligned}$$

We assume that this model is known to the agents.

Assumption 2 (Limited Sensing Range) *The agents can observe a target only if it exists within the sensor's range $b \in \mathbb{R}_{>0}$. This range may or may not be known to the agents. We also assume that the agents are homogeneous. Accounting for the sensor heterogeneity could be left for future work.*

Assumption 3 (Data Association) *The agents know which targets they are observing. In other words, we assume that the data association is completed upon the observation.*

The third assumption may be strong and not always be true depending on the scenario, but for simplicity we omit the complexity related to the data association in this work.

Given the observation model, we employ the Bayesian approach to keep track of the posterior belief. Before considering the multi-agent and multi-target case, we will provide a brief review of the recursive Bayesian estimation for the one-agent, one-target case. Let $X \in \mathbb{R}^2$ be a continuous random variable following a distribution $p(x|\theta) = P(X = x|\theta)$, where θ is a parameter determining the form of the distribution. In our case, $p(x|\theta)$ corresponds to $\phi(q)$ and the governing parameter θ is the unknown target position x_k. Our purpose is to estimate this parameter θ in the Bayesian framework, so we will create a belief state $p(\theta)$ and update it based on a set of m_k observations $z_k = \{z_{k1}, z_{k2}, \ldots, z_{km_k}\}$ of the target x_k. Furthermore, we introduce a *hyperparameter* ω, which defines the distribution over θ. From the Bayes' Theorem, the *posterior* distribution over θ after observing z is:

$$p(\theta|z, \omega) = \frac{p(z|\theta)p(\theta|\omega)}{\int p(z, \theta|\omega)d\theta} \propto p(z|\theta)p(\theta|\omega) \tag{11}$$

where $p(z|\theta)$ is the measurement likelihood function and $p(\theta|\omega)$ is the *prior* distribution conditioned on ω. If the observations are uncorrelated with one another, or conditionally independent given θ, the term $p(z|\theta)$ can be decomposed as

$$p(z|\theta) = p(z_{k1}|\theta)p(z_{k2}|\theta)\ldots p(z_{km_k}|\theta). \tag{12}$$

(11) and (12) illustrate how we can compute the posterior from the prior, the measurement likelihood function and the actual observations. However, numerically computing the integration of the denominator in (11) can be computationally expensive. We therefore introduce the *conjugate prior distribution* to circumvent this problem. If the likelihood is conjugate to the prior, then the posterior is of the same form as the prior and the hyperparameter can be updated in a closed form.

$$p(\theta|z, \omega) = p(\theta|\omega_{pos}) \tag{13}$$

For the observation model given by (9) and (10), the conjugate prior is a Gaussian distribution $\mathcal{N}(x_k|\mu_k, \Sigma_k)$ with the mean μ_k and the covariance matrix Σ_k. From [12], the hyperparameters are the information form of the Kalman filter:

$$\Lambda_k = \Sigma_k^{-1} \tag{14}$$
$$y_k = \Lambda_k \mu_k. \tag{15}$$

The measurement updates are additive:

$$\Lambda_k \leftarrow \Lambda_k + R^{-1} \tag{16}$$
$$y_k \leftarrow y_k + R^{-1} z_k. \tag{17}$$

Thus, the recursive Baysian estimation with m_k observations $\{z_{k1}, z_{k2}, \ldots, z_{km_k}\}$ is done by iteratively updating the hyperparamaters in an additive form.

$$\Lambda_k \leftarrow \Lambda_k + m_k R^{-1} \tag{18}$$

$$y_k \leftarrow y_k + R^{-1} \sum_{j=1}^{m_k} z_{ki} \tag{19}$$

The hyperparameter updates can be also applied to multi-agent situations. [12] proposes a hyperparameter consensus algorithm leveraging this additive structure for belief synchronization and we adapt their method to this project. The belief synchronization consists of the following two stages.

1. Local Update Stage
   Starting from the same prior hyperparameters, each agent independently updates them based on the new observations unique to it and subtract the prior values.

   $$\Delta\Lambda_{ki}[0] \leftarrow m_{ki} R^{-1} \tag{20}$$

   $$\Delta y_{ki}[0] \leftarrow R^{-1} \sum_{j=1}^{m_{k_i}} z_{ki}, \tag{21}$$

   where m_{ki} denotes the number of new measurements taken by the agent p_i.

2. Linear Consensus Stage
   The linear consensus protocol runs an iterative loop until convergence and fuses $\Delta\Lambda_{ki}$ and Δy_{ki} respectively. Each iterative step is given by

   $$\Delta\Lambda_{ki}[l+1] = \Delta\Lambda_{ki}[l] + \epsilon \sum_{j \in \mathcal{N}_i} \left(\Delta\Lambda_{kj}[l] - \Delta\Lambda_{ki}[l]\right) \tag{22}$$

   $$\Delta y_{ki}[l+1] = \Delta y_{ki}[l] + \epsilon \sum_{j \in \mathcal{N}_i} \left(\Delta y_{kj}[l] - \Delta y_{ki}[l]\right), \tag{23}$$

   where $\mathcal{N}_i$ is the set of neighbors of the agent p_i in the communication graph. After the convergence, each agent updates the fused posterior hyperparameters as follows.

   $$\Lambda_k \leftarrow \Lambda_k + N\Delta\Lambda_{ki} \tag{24}$$

   $$y_k \leftarrow y_k + N\Delta y_{ki} \tag{25}$$

If the communication graph is undirected, (22) and (23) is guaranteed to reach an asymptotic consensus as long as the graph is connected and $\epsilon \in (0, 1/\max|\mathcal{N}_i|]$. The proof is presented in [16]. The group decision values are the averages of the initial states:

$$\lim_{l\to\infty} \Delta\Lambda_{ki}[l] = \frac{1}{N}\sum_{i=1}^{N} \Delta\Lambda_{ki}[0] \tag{26}$$

$$\lim_{l\to\infty} \Delta y_{ki}[l] = \frac{1}{N}\sum_{i=1}^{N} \Delta y_{ki}[0]. \tag{27}$$

From (20) - (27), the fused hyperparameters are

$$\Lambda_k \leftarrow \Lambda_k + \left(\sum_{i=1}^{N} m_{ki}\right) R^{-1} \tag{28}$$

$$y_k \leftarrow y_k + R^{-1}\left(\sum_{i=1}^{N}\sum_{j=1}^{m_{ki}} z_{ki}\right), \tag{29}$$

which correspond to the hyperparameters updated using all the new observations taken by the agents. Therefore, belief synchronization is achieved after the consensus loop.

The hyperparameter consensus algorithm discussed above is focused on the one-target belief update, but it can be also applied to the multi-target scenario. In the next section we propose the complete algorithm for the multi-target case as well as for the one-target case.

4 Proposed Algorithms

4.1 One-Target Search and Coverage

The one-target search and coverage algorithm is presented in Algorithm 1. First, the Voronoi cells are computed according to the current agent positions. Lines 3-15 are the hyperparameter consensus algorithm consisting of the local update stage and the linear consensus stage. In line 16, the mean μ_t of the current distribution is extracted. This mean is used as an estimate for the target position x and governs the belief $\phi(q)$. If the target has been observed by any of the agents, then $\phi(q) = \mathcal{N}(q|\mu_t, R)$ gives the current belief. If not, we simply assume that $\phi(q)$ is the uniform distribution, meaning that the target is equally likely to be positioned at any points in Q. Note that we also need to normalize $\phi(q)$ so that $\int_Q \phi(q)dq = 1$ holds. Lastly, the Voronoi centroids are computed and the agents are moved to the next positions by (8).

One advantage of this algorithm is that the agents do not need to broadcast the observations to others, which could be difficult under communication constraints. The only required information to be exchanged is the agent positions and the locally updated hyperparameters among the Voronoi neighbors.

```
input : Prior hyperparameters: $y_0$, $\Lambda_0$
        Initial agent positions: $\{p_1[0], \dots, p_N[0]\}$
        Connected undirected communication graph: $\mathcal{G}$
        True target position: $x$
        Observation noise covariance: $R$
1  for each time step $t > 0$ do
2      $ComputeVoronoiCells(p_1[t], \dots, p_N[t])$
3      for each agent $i \in \{1, \dots, N\}$ do /* Local update stage */
4          $\Delta\Lambda_{it}[0] \leftarrow 0$
5          $\Delta y_{it}[0] \leftarrow 0$
6          if $||x - p_i[t]|| \leq b$ then
7              $z_{it} = x + v,\ v \sim \mathcal{N}(0, R)$
8              $\Delta\Lambda_{it}[0] \leftarrow R^{-1}$
9              $\Delta y_{it}[0] \leftarrow R^{-1} z_{it}$
           end
       end
10     for each agent $i \in \{1, \dots, N\}$ do
11         repeat/* Linear consensus stage */
12             $\Delta\Lambda_{it}[l+1] \leftarrow \Delta\Lambda_{it}[l] + \epsilon \sum_{j \in \mathcal{N}_i} (\Delta\Lambda_{jt}[l] - \Delta\Lambda_{it}[l])$
13             $\Delta y_{it}[l+1] \leftarrow \Delta y_{it}[l] + \epsilon \sum_{j \in \mathcal{N}_i} (\Delta y_{jt}[l] - \Delta y_{it}[l])$
           until convergence;
14         $\Lambda_t \leftarrow \Lambda_{t-1} + N\Delta\Lambda_{it}[end]$
15         $y_t \leftarrow y_{t-1} + N\Delta y_{it}[end]$
16         $\mu_t \leftarrow \Lambda_t^{-1} y_t$
17         if $\mu_t \neq \mu_0$ then
18             $\phi(q) = \mathcal{N}(q|\mu_t, R)$
           end
19         else
20             $\phi(q) = 1/\int_Q dq$
           end
21         normalize $\phi(q)$ $q \in Q$
22         compute $C_{V_i} = \int_{V_i} q\phi(q)dq / \int_{V_i} \phi(q)dq$
23         $p_i[t+1] = p_i[t] + \eta(C_{V_i}[t] - p_i[t])\Delta t$
       end
   end
```

Algorithm 1: One-target search and coverage algorithm

```
input  : Prior hyperparameters: $y_{k0} = y_0,\ \Lambda_{k0} = \Lambda_0\ \forall k \in \{1, \dots, K\}$
         Initial agent positions: $\{p_1[0], \dots, p_N[0]\}$
         Connected undirected communication graph: $\mathcal{G}$
         True target positions: $\{x_1, \dots, x_K\}$
         Observation noise covariance: $R$
1  for each time step $t > 0$ do
2      $ComputeVoronoiCells(p_1[t], \dots, p_N[t])$
3      for each agent $i \in \{1, \dots, N\}$ do /* Local update stage */
4          for each target $k \in \{1, \dots, K\}$ do
5              $\Delta\Lambda_{kit}[0] \leftarrow 0$
6              $\Delta y_{kit}[0] \leftarrow 0$
7              if $||x_k - p_i[t]|| \leq b$ then
8                  $z_{kit} = x_k + v,\ v \sim \mathcal{N}(0, R)$
9                  $\Delta\Lambda_{kit}[0] \leftarrow R^{-1}$
10                 $\Delta y_{kit}[0] \leftarrow R^{-1} z_{kit}$
               end
           end
       end
11     for each agent $i \in \{1, \dots, N\}$ do
12         for each target $k \in \{1, \dots, K\}$ do
13             repeat/* Linear consensus stage */
14                 $\Delta\Lambda_{kit}[l+1] \leftarrow \Delta\Lambda_{kit}[l] + \epsilon \sum_{j \in \mathcal{N}_i} (\Delta\Lambda_{kjt}[l] - \Delta\Lambda_{kit}[l])$
15                 $\Delta y_{kit}[l+1] \leftarrow \Delta y_{kit}[l] + \epsilon \sum_{j \in \mathcal{N}_i} (\Delta y_{kjt}[l] - \Delta y_{kit}[l])$
               until convergence;
16             $\Lambda_{kt} \leftarrow \Lambda_{kt-1} + N\Delta\Lambda_{kit}[end]$
17             $y_{kt} \leftarrow y_{kt-1} + N\Delta y_{kit}[end]$
18             $\mu_{kt} \leftarrow \Lambda_{kt}^{-1} y_{kt}$
19             if $\mu_{kt} \neq \mu_0$ then
20                 $\phi_k(q) = \mathcal{N}(q|\mu_{kt}, R)$
               end
21             else
22                 $\phi_k(q) = 1/\int_Q dq$
               end
           end
23         $\phi(q) = \frac{1}{K}\sum_{k=1}^{K} \phi_k(q)$
24         normalize $\phi(q)\ q \in Q$
25         compute $C_{V_i} = \int_{V_i} q\phi(q)dq / \int_{V_i} \phi(q)dq$
26         $p_i[t+1] = p_i[t] + \eta(C_{V_i}[t] - p_i[t])\Delta t$
       end
   end
```

Algorithm 2: Multi-target search and coverage algorithm

4.2 Multi-Target Search and Coverage

The same algorithm for the multi-target case is presented in Algorithm 2. Lines 3-22 are the hyperparameter consensus algorithm for each target. This decomposition is possible due to Assumption 3. Compared to the one-target algorithm, the computational complexity of this part is proportional to the number of the targets K.

Another notable difference is the expression of $\phi(q)$ in line 23. $\phi(q) = \frac{1}{K}\sum_{k=1}^{K}\phi_k(q)$ is a mixture of Gaussians with the uniform mixture ratio $1/K$, where $\phi_k(q)$ is the probability density that the target k is positioned at q. This form of $\phi(q)$ equally weighs the influence of each target. We could also assign distinct weights in order to account for the difference in the relative importance of the targets.

5 Simulation Results

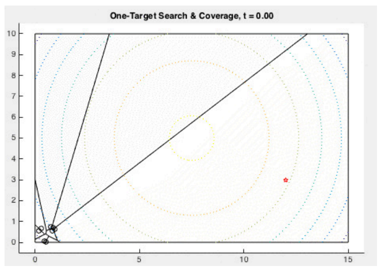
In this section we demonstrate the performance of the proposed algorithms in the one-target and the multi-target scenarios. For both cases, the working domain of the agents is bounded by a 10x15 rectangular boundary. Each agent is assumed to be a point in the domain and initially positioned randomly within the 1x1 squared region on the bottom left corner. The prior hyperparameters are $\Lambda_0 = diag((10^{-9}, 10^{-9}))$ and $y_0 = \Lambda_0(7.5, 5)^{\mathrm{T}}$. The sensing range is set to be 3 with the noise covariance $R = diag((0.5, 0.5))$. We used the Multi-Parametric Toolbox 3 [17] on MATLAB for the computation of Voronoi cells.

5.1 One-target Case

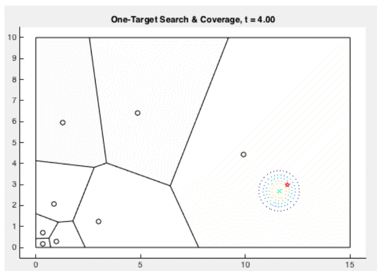
Figure 1 presents the deployment process of the 8 agents represented by the circles. The target is positioned at $(12, 3)$. At $t = 4$, an agent observes the target. The cross mark corresponds to the observation and the dashed contours illustrate the fused belief. Having found the target, all the agents move towards it and converge to a final configuration as shown in Figure 2. Figure 3 shows the transition of the minimum distance between the target and the agents. Although the distance does not converge to zero due to the stochasticity of the observation model, it decreased from 11.3 at $t = 0$ to 0.2 at $t = 46$.

5.2 Multi-target Case

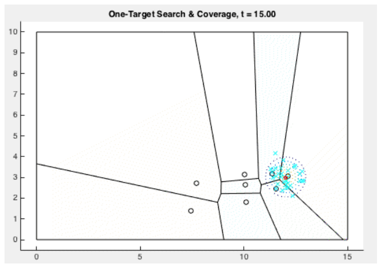
Figure 4 shows the deployment process of the 8 agents searching for 3 targets, which are placed near the corners of the domain. + marks indicate the Voronoi centroids. At $t = 5.5$, an agent observes the first target at $(1.5, 9)$ and move towards it. Unlike in the one-target case, however, the other agents keep exploring in the domain. This is because of the uniform beliefs about the two unobserved targets; the uniform beliefs encourage the agents to deploy uniformly across the domain, thus enhancing exploration.



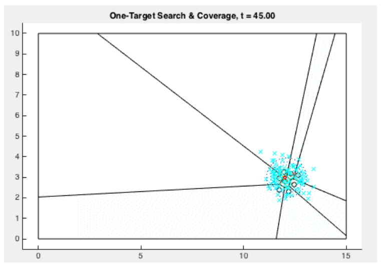
(a) $t = 0$



(b) $t = 4$



(c) $t = 15$



(d) $t = 45$

Figure 1: One-target search and coverage deployment process with 8 agents. The target is positioned at $(12, 3)$.

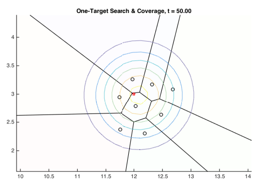


Figure 2: Converged configuration (one-target)

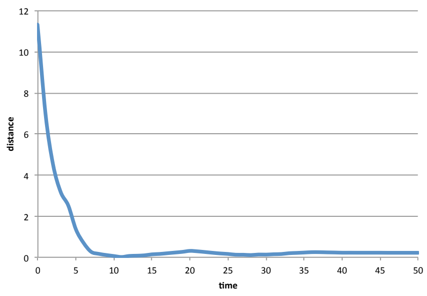
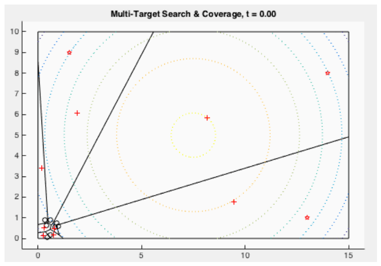
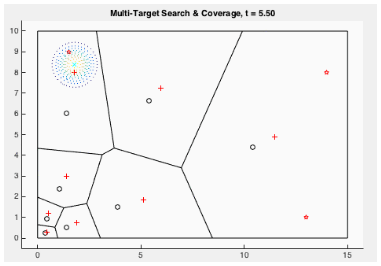


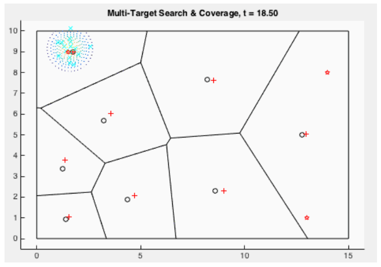
Figure 3: Minimum distance from the target



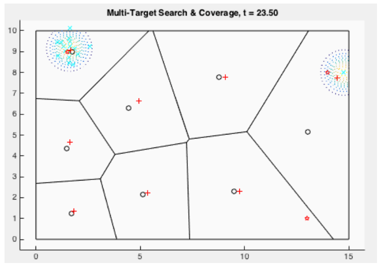
(a) $t = 0$



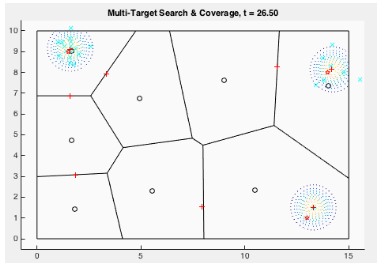
(b) $t = 5.5$



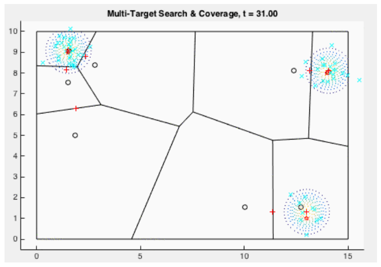
(c) $t = 18.5$



(d) $t = 23.5$

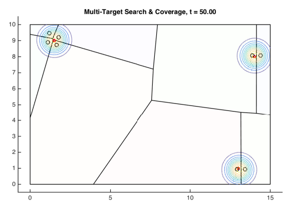


(e) $t = 26.5$



(f) $t = 31$

Figure 4: Multi-target search and coverage deployment process with 8 agents. The targets are positioned at $(13, 1)$, $(1.5, 9)$ and $(14, 8)$.



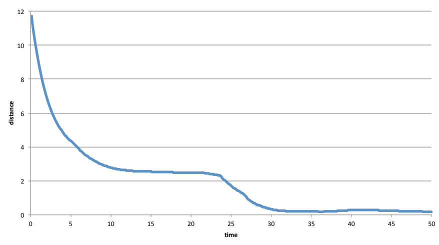


Figure 5: Converged configuration (multi-target)

Figure 6: Average of the minimum distances from the three targets

An interesting phenomenon occurs at $t = 26.5$. At this time, the last unseen target is observed and the Voronoi centroids have shifted to the boundaries. Since all the uniform beliefs have disappeared, exploration is no longer encouraged and all the agents move towards the targets. The final configuration is presented in Figure 5. As can be seen in Figure 6, the average of the three minimum distances from each target to the agents is reduced from 11.7 to 0.2. The sudden decrease around $t = 24$ in Figure 6 corresponds to the configuration in Figure 4d, when the second target is found.

6 Conclusion

In this paper we have integrated the hyperparameter consensus protocol with the conventional coverage cost function to operationalize decentralized multi-target search with a team of networked sensing agents. The proposed algorithms work online and only require information exchange between the Voronoi neighbors. We have demonstrated the feasibility of our approach via a set of simulations. Although this heuristic method does not guarantee that all the targets will be found, it is worthwhile to note that the uniform beliefs resulting from the unseen targets contribute to balancing exploration and exploitation.

One limitation of our approach is that the hyperparameter update must be in an additive form. This limits the class of distributions applicable to the proposed framework. In addition, the resource allocation optimality is not considered in the controller design. The data association problem is also excluded. We are interested in extending this research to address these problems as well as providing rigorous mathematical analysis of the proposed algorithms.

References

[1] 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.

[2] J. W. Durham, R. Carli, P. Frasca, and F. Bullo, "Discrete Partitioning and Coverage Control for Gossiping Robots," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 364–378, 2012.

[3] L. C. A. Pimenta, V. Kumar, R. C. Mesquita, and G. A. S. Pereira, "Sensing and coverage for a network of heterogeneous robots," in *2008 47th IEEE Conference on Decision and Control*. IEEE, 2008, pp. 3947–3952.

[4] M. Schwager, D. Rus, and J.-J. Slotine, "Decentralized, Adaptive Coverage Control for Networked Robots," *The International Journal of Robotics Research*, vol. 28, no. 3, pp. 357–375, 2009.

[5] C. Song, G. Feng, Y. Fan, and Y. Wang, "Decentralized adaptive awareness coverage control for multi-agent networks," *Automatica*, vol. 47, no. 12, pp. 2749–2756, 2011.

[6] C. Song, L. Liu, G. Feng, Y. Wang, and Q. Gao, "Persistent awareness coverage control for mobile sensor networks," *Automatica*, vol. 49, no. 6, pp. 1867–1873, 2013.

[7] T. Campbell and J. P. How, "Approximate Decentralized Bayesian Inference," *30th Conference on Uncertainty in Artificial Intelligence*, 2014.

[8] J. Predd, S. Kulkarni, and H. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp. 56–69, 2006.

[9] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.* IEEE, 2005, pp. 63–70.

[10] P. M. Djuric, "Distributed Bayesian learning in multiagent systems: Improving our understanding of its capabilities and limitations," *IEEE Signal Processing Magazine*, vol. 29, no. 2, pp. 65–76, 2012.

[11] S. Bandyopadhyay and S.-J. Chung, "Distributed estimation using Bayesian consensus filtering," in *2014 American Control Conference.* IEEE, 2014, pp. 634–641.

[12] C. S. R. Fraser, L. F. Bertuccelli, H. L. Choi, and J. P. How, "A hyperparameter consensus method for agreement under uncertainty," *Automatica*, vol. 48, no. 2, pp. 374–380, 2012.

[13] K. E. Hoff, J. Keyser, M. Lin, D. Manocha, and T. Culver, "Fast computation of generalized Voronoi diagrams using graphics hardware," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques - SIGGRAPH '99.* New York, New York, USA: ACM Press, 1999, pp. 277–286.

[14] M. Velić, D. May, and L. Moresi, "A Fast Robust Algorithm for Computing Discrete Voronoi Diagrams," *Journal of Mathematical Modelling and Algorithms*, vol. 8, no. 3, pp. 343–355, 2008.

[15] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[16] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[17] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *Proc. of the European Control Conference*, Zürich, Switzerland, July 17–19 2013, pp. 502–510, http://control.ee.ethz.ch/ mpt.