2009 American Control Conference
Hyatt Regency Riverfront, St. Louis, MO, USA
June 10-12, 2009

A distributed deterministic annealing algorithm for limited-range sensor coverage

Andrew Kwok and Sonia Martínez

Abstract— **This paper presents a distributed coverage algorithm for a network of mobile agents. Unlike previous work that uses a simple gradient descent algorithm, here we employ an existing deterministic annealing (DA) technique to achieve more optimal convergence values. We replicate the results of the classical DA algorithm while imposing a limited-range constraint to sensors. As the temperature is decreased, phase changes lead to a regrouping of agents, which is decided through a distributed task allocation algorithm. While simple gradient descent algorithms are heavily dependent on initial conditions, annealing techniques are generally less prone to this phenomena. The results of our simulations confirm this fact, as we show in the manuscript.**

I. Introduction

The ability to autonomously deploy over a spatial region, as well as to dynamically adjust to single-point failures, gives mobile networks an advantage over static ones. This prompts the study of designing effective motion coordination algorithms for their unsupervised control [1]. A key area of interest regarding mobile sensor networks is deployment to maximize coverage [2], [3], [4], [5], [6].

However, most current methods for deployment, i.e. [2], [3], rely on gradient techniques to converge to an extremum of a cost function. Consequently, the resulting final value of the cost function may not be the globally optimal one. Many annealing techniques exist to find a better optimal value of a cost function. Of these techniques, there are simulated annealing (SA) algorithms [7], as well as a more recent development, deterministic annealing (DA) [8]. Unfortunately, these are centralized algorithms requiring global knowledge of the total state of the system.

Annealing algorithms differ from standard gradient algorithms through the addition of a temperature state. The goal, as in physical annealing, is to gradually lower this temperature, so that the internal configuration of the system is always at or near the lowest energy state. The SA and DA techniques feature *phase changes* as the temperatures are lowered past certain critical values, and we quantify these transitions for the distributed algorithm version.

A closely related work is that of Sharma et. al. [9]. The resulting algorithm discards information of other agents and resources that are far from a given agent. However, the algorithm still requires knowledge of all agents involved in the optimization to determine the information to discard.

In [10], SA was used to solve the clustering and formation control problems. That work also considered limited-range interactions, however, punctual long-range communication between agents was required. A cell decomposition of the environment had to be done a priori.

In this paper, we extend the DA algorithm of [8]. Here, we take that discrete DA algorithm to make it continuous in both space and time as well as spatially distributed. We strictly enforce that an individual agent can only sense the presence of other agents within a fixed radius. To do so, we introduce a spatial partition of the environment, and use this to develop a distributed local check of phase changes. Additionally, we introduce a task assignment algorithm to reassign vehicles according to phase changes. With the limited-range constraint, we achieve very similar results as in [8], [9]. Additionally, as this sensing radius increases, the algorithm recovers the original DA algorithm.

The paper is organized as follows. In section II, we introduce the limited range coverage problem, as well as provide an overview of the DA algorithm. In section III we derive the gradient direction for a limited-range DA algorithm, and continue in Section IV to provide a sufficient condition to distributively check for phase changes. We merge the two results in Section V by describing an algorithm for a network of autonomous agents to implement that includes a task allocation subroutine. Finally, simulations in Section VI demonstrate the limited-range DA algorithm, followed by some concluding remarks.

II. Notation and the DA algorithm

Let Q be a convex polytope in $\mathbb{R}^d$ including its interior, and let $\|\cdot\|$ denote the Euclidean norm. We will use $\mathbb{R}_{\geq 0}$ to denote the set of positive real numbers. A map $\phi : Q \to \mathbb{R}_{\geq 0}$, or a *distribution density function*, will represent a measure of a priori known information that some event takes place over Q. Equivalently, we consider Q to be the bounded support of the function ϕ. We will also denote the boundary of a set S as ∂S and the interior of a set as $\mathrm{Int}(S)$.

The proposed limited-range distributed DA algorithm is based on formations of agents (with leaders at $p_1, \dots, p_n$) that split during phase changes. The algorithm finishes with formations of N single vehicles at positions $p_1, \dots, p_N$. All agents have a limited sensing radius R_i, which may be heterogeneous.

We now briefly describe the minimization process of the DA scheme as well as compare this with the method in [2]. In [8], the end goal is to minimize a *distortion function*,

$$D = \int_Q \phi(q) \sum_{i=1}^{n} \mathbb{P}(p_i|q) f_i(\|q - p_i\|) dq \,, \qquad (1)$$

A. Kwok and S. Martínez are at the department of Mechanical and Aerospace Engineering, Univ. of California, San Diego, 9500 Gilman Dr, La Jolla CA, 92093 {ankwok, soniamd}@ucsd.edu

978-1-4244-4524-0/09/$25.00 ©2009 AACC

where $f_i\colon \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a general metric (typically $f_i(x) = x^2$) and $\mathbb{P}(p_i|q)$ is the probability of a point q being associated with an agent p_i. However, (1) is not directly minimized. The Shannon entropy function is introduced:

$$H = -\int_Q \phi(q) \sum_{i=1}^{n} \mathbb{P}(p_i|q) \log \mathbb{P}(p_i|q)\,, \tag{2}$$

and the DA algorithm is a discrete-time algorithm that involves the minimization of the Lagrangian $F = D - TH$, where T is the temperature of the system. As temperature decreases, minimizing F becomes more similar to minimizing D. The association probabilities $\mathbb{P}(p_i|q)$ are derived from $\mathbb{P}^*(p_i|q) = \operatorname{argmin}_{\mathbb{P}(p_i|q)} F$. Then, the resulting $\mathbb{P}^*(p_i|q)$ are substituted into F to yield $\hat{F}$, and the optimal agent locations are given by $p_i^* = \operatorname{argmin}_{p_i} \hat{F}$.

As temperature decreases, the system undergoes *phase changes*. A phase change occurs when an equilibrium position p_i^* is no longer attractive in the presence of more than one sufficiently close agent. Rose in [8] provides a necessary and sufficient condition to detect phase changes, and we will provide an analogous check in the limited-range case.

In contrast the objective in [2] was to minimize (1) with trivial association probabilities determined by a Voronoi partition of Q. That is, the probability of $q \in Q$ being associated to p_i is 1 iff q is in its generalized Voronoi region.

As in [2], we choose to analyze the distributed DA coverage problem via general metrics $f_i\colon \mathbb{R}_{\geq 0} \to \mathbb{R}$ such that f_i is Lipschitz and non-decreasing. We assume that each f_i is of the form

$$f_i(x) = g_i(x) 1_{[0,R_i)}\,, \tag{3}$$

such that each g_i is differentiable and non-decreasing over $[0, R_i)$ and $g_i(R_i) = 0$ for continuity.

In what follows we will consider the limited-range heterogeneous analogues of the centroidal sensing metric found in [2]. The sensing function is:

$$f_i^m(x) = \left[x^2 - R_i^2\right] 1_{[0,R_i)}(x)\,, \tag{4}$$

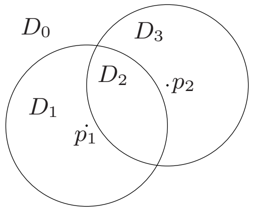
III. Limited-range DA Lagrangian gradient

In order to obtain a continuous-time version of the DA algorithm, we compute the gradient of the Lagrangian F with sensing functions (3) in this section. We first start with a derivation of the association probabilities, and then introduce a partition of Q that takes advantage of the limited-range nature of agent sensors. We refer the reader to the full paper [11] for more details on the computations that follow.

A. Limited-range association probabilities

Similar to the original DA algorithm, we consider each point $q \in Q$ to have some probability of being associated with an agent at p_i. The probabilities, $\mathbb{P}(p_i|q)$ $i \in \{1, \ldots, n\}$, satisfy the following constraint for all $q \in Q$:

$$\sum_{i=1}^{n} \mathbb{P}(p_i|q) = 1\,. \tag{5}$$



Variable	Description
A_1	$D_1 \cup D_2$
B_1	$\{1\}$
B_2	$\{1, 2\}$
C_2	$\{2, 3\}$

Fig. 1. Here we show an example for the notation we have introduced.

Lemma 1: The association probability distribution that minimizes $F = D - TH$ and satisfies (5) is the Gibbs distribution

$$\mathbb{P}(p_i|q) = \frac{\exp\left[-\frac{f_i(\|q - p_i\|)}{T}\right]}{Z(q)}\,, i \in \{1, \ldots, n\}\,, \tag{6}$$

where the normalizing factor is:

$$Z(q) = \sum_{i=1}^{n} \exp\left[-\frac{f_i(\|q - p_i\|)}{T}\right]. \tag{7}$$

We can take the result (6) and substitute it back into F:

$$\hat{F} = -T \int_Q \phi(q) \log Z(q) dq\,, \tag{8}$$

where we use the fact that $\sum_{i=1}^{n} \mathbb{P}(p_i|q) = 1$.

B. Limited-range partition

For further analysis, it is advantageous to partition Q such that $Z(q)$ is differentiable over each region of this partition. We start by assuming that each sensing function f_i has the form from (3). We can define the set

$$A_i = \{q \in Q \parallel 0 \leq \|q - p_i\| < R_i\}\,. \tag{9}$$

Additionally, let β be the set of binary sequences of length n, i.e.: each $b_k \in \beta$, $k \in \{1, \ldots, 2^n\}$ is a finite sequence of zeros and ones.

Proposition 2: Let $\{D_k\}$ be a collection of sets such that for each $b_k \in \beta$,

$$D_k = \bigcap_{i=1}^{M} \{A_i \text{ if } b_{k,i} = 1\,; A_i^C \text{ if } b_{k,i} = 0\}\,. \tag{10}$$

Then, $\{D_k\}$ forms a partition of Q and $Z(q)$ is continuously differentiable in each D_k. •

In the next section, we will use B_k to refer to the indices of the points p_i which form the region D_k. That is,

$$B_k = \{i \in \{1, \ldots, n\} \mid \|q - p_i\| < R_i, \forall\, q \in \operatorname{Int}(D_k)\}\,.$$

The regions D_k also have a convenient relation to each B_i.

Proposition 3: Each ball A_i of radius R_i centered at p_i is *exactly* covered by a subcollection of $\{D_k\}$. We denote the set of indices corresponding to this subcollection as C_i such that $A_i = \bigcup_{k \in C_i} D_k$. •

C. Gradient formulation

The next step in the DA derivation is to optimize the Lagrangian $\hat{F}$ with respect to sensor positions p_i. Each agent in the network will use this result in order to compute its gradient direction.

Proposition 4: Given the Lagrangian (8), and sensing metrics of the form (3), the gradient of (8) is:

$$\frac{\partial \hat{F}}{\partial p_i} = -T \sum_{k\in C_i} \int_{D_k} \phi(q) \frac{1}{Z(q)} \frac{\partial Z}{\partial p_i} dq\,. \tag{11}$$

Remark 5: We can compute the derivative $\frac{\partial Z}{\partial p_i}$ using the sensing function (4). We begin by computing $\frac{\partial Z}{\partial p_i}$.

$$\frac{\partial Z}{\partial p_i} = \frac{2}{T}(q-p_i)^T \exp\left[-\frac{\|q-p_i\|^2 - R^2}{T}\right],$$

whenever $\|q-p_i\| < R_i$, otherwise $\frac{\partial Z}{\partial p_i} = 0$. We then obtain the gradient (11) to be:

$$\frac{\partial \hat{F}}{\partial p_i} = -2 \sum_{k\in C_i} \int_{D_k} \phi(q)(q-p_i)^T \mathbb{P}(p_i|q) dq\,. \tag{12}$$

This is similar to the gradient expression for the mixed coverage case in [2], with the addition of the association probabilities $\mathbb{P}(p_i|q)$ as an extra weighting factor. •

IV. Limited-range DA phase changes

As temperature decreases, the equilibrium points of $\hat{F}$ under the evolution of (11) become unstable. When this happens a phase change occurs and we say that we have reached a critical temperature. We present a sufficient condition for agents to individually check if they have reached a critical temperature value under both area-maximizing and mixed centroidal-area coverage.

Using a similar argument as in [8], we enlarge the group of leaders $\{p_1,\dots,p_n\}$ with a set of virtual agents $\{p_{n+1},\dots,p_l\}$ so that for all $j\in\{n+1,\dots,l\}$, $p_j = p_i$ for some $i\in\{1,\dots,n\}$. Then we introduce perturbations $\Psi = (\psi_1,\dots,\psi_l) \in \mathbb{R}^{2l}$. Given a scaling factor ϵ, consider the perturbed agent locations, $x_i = p_i + \epsilon\psi_i$, for $i\in\{1,\dots,l\}$. Critical points of $\hat{F}$ correspond to configurations where $\left.\frac{d\hat{F}(x_1,\dots,x_l)}{d\epsilon}\right|_{\epsilon=0} = 0$. However, those configurations fail to be a minimum when the second derivative $\left.\frac{d^2\hat{F}}{d\epsilon^2}\right|_{\epsilon=0} \le 0$. We now find the second derivative. Consider the partition $\{D_k\}$ associated with the $\{x_i\}$, $i\in\{1,\dots,l\}$.

Consider the second derivative of $\hat{F}$ with respect to ϵ:

$$\begin{aligned}\frac{d^2\hat{F}}{d\epsilon^2} = -T\sum_k \Bigg\{ &\int_{D_k} \phi(q)\left[-\frac{1}{Z^2(q)}\left(\frac{\partial Z}{\partial \epsilon}\right)^2 + \frac{1}{Z(q)}\frac{\partial^2 Z}{\partial \epsilon^2}\right] dq \\ &+ \int_{\partial D_k} \phi(\gamma_k)\frac{1}{Z(\gamma_k)}\frac{\partial Z}{\partial \epsilon} n^T(\gamma_k)\frac{\partial \gamma_k}{\partial \epsilon} d\gamma_k \Bigg\}.\end{aligned} \tag{13}$$

Let $y_i = q - x_i$ to reduce the amount of notation. In a particular region D_k, the derivative of Z with respect to ϵ simplifies to

$$\frac{dZ}{d\epsilon} = \sum_{j\in B_k} -\frac{1}{T}\frac{\partial f_j}{\partial \epsilon}\exp\left[-\frac{f_j(\|y_j\|)}{T}\right]. \tag{14}$$

Continuing, the second derivative is:

$$\frac{d^2 Z}{d\epsilon^2} = \sum_{j\in B_k}\left\{\left[\left(\frac{1}{T}\frac{\partial f_j}{\partial \epsilon}\right)^2 - \frac{1}{T}\frac{\partial^2 f_j}{\partial \epsilon^2}\right]\exp\left[-\frac{f_j(\|y_j\|)}{T}\right]\right\}. \tag{15}$$

Since $y_i = q - p_i - \epsilon\psi_i$, using the chain rule, $\frac{\partial f_i}{\partial \epsilon} = -\psi_i^T\frac{\partial f_i}{\partial y_i}$ and $\frac{\partial^2 f_i}{\partial \epsilon^2} = \psi_i^T\frac{\partial^2 f_i}{\partial y_i^2}\psi_i$.

We substitute the results (14) and (15) into (13), and note that $\frac{1}{Z}\exp\left[-\frac{f_i}{T}\right] = \mathbb{P}(x_i|q)$ to get:

$$\begin{aligned}\frac{d^2\hat{F}}{d\epsilon^2} = -T\sum_k \Bigg\{ &\int_{D_k}\phi(q)\Bigg(-\frac{1}{T^2}\left(\sum_{j\in B_k}\frac{\partial f_j}{\partial \epsilon}\mathbb{P}(x_j|q)\right)^2 \\ &+ \sum_{j\in B_k}\left[\left(\frac{1}{T}\frac{\partial f_j}{\partial \epsilon}\right)^2 - \frac{1}{T}\frac{\partial^2 f_j}{\partial \epsilon^2}\right]\mathbb{P}(x_j|q)\Bigg) dq \\ &- \frac{1}{T}\int_{\partial D_k}\phi(\gamma_k)\sum_{j\in B_k}\left(\frac{\partial f_j}{\partial \epsilon}\mathbb{P}(p_j|\gamma_k)\right) n^T(\gamma_k)\frac{\partial \gamma_k}{\partial \epsilon} d\gamma_k\Bigg\}.\end{aligned} \tag{16}$$

The check for critical temperature is to numerically compute $\left.\frac{d^2\hat{F}}{d\epsilon^2}\right|_{\epsilon=0}$ at an equilibrium configuration. The equilibrium configurations occurs when $\frac{\partial \hat{F}}{\partial p_i} = 0$ for all i, or equivalently, when $\left.\frac{d\hat{F}}{d\epsilon}\right|_{\epsilon=0} = 0$. If the second derivative is negative, then the equilibrium configuration is unstable, and that signifies that we are below a critical temperature value. To simplify the critical temperature check and make it spatially distributed, we consider the following perturbation. Let $S_i \subseteq \{1,\dots,m\}$ be such that $j\in S_i$ implies $p_j = p_i$. We define Ψ_i to be

$$\Psi_i = \left\{(\psi_1,\dots,\psi_m) \mid \psi_j = 0, \forall\, j\notin S_i; \sum_{j\in S_i}\psi_j = 0\right\}. \tag{17}$$

If the critical temperature has not yet been reached, then these coincident agents (i.e., leaders and virtual agents) will remain together. Otherwise, the coincident agents are at an unstable equilibrium point, and any perturbation will force them apart. Using this particular perturbation, we will obtain a sufficient condition for critical temperature. We will now take the above results and consider the metric function (4). This metric is most similar to that found in [8] and [9].

Proposition 6: Critical temperature for the centroidal-area DA algorithm has been reached if for $i\in\{1,\dots,n\}$, any of the following matrices $\mathcal{F}_i$ are negative definite:

$$\mathcal{F}_i = \sum_{k\in C_i}\int_{D_k}\phi(q)\mathbb{P}(p_i|q)\left[I - \frac{2}{T}(q-p_i)(q-p_i)^T\right] dq, \tag{18}$$

and $\dot{p}_i = 0$ for all $i\in\{1,\dots,n\}$.

Proof: The derivatives $\frac{\partial f_i^m}{\partial y_i}$ and $\frac{\partial^2 f_i^m}{\partial y_i^2}$, when $\|y_i\| \le R_i$, are:

$$\frac{\partial f_i^m}{\partial y_i} = 2y_i^T\,, \qquad \frac{\partial^2 f_i^m}{\partial y_i^2} = 2I\,.$$

Since $y_i = q - p_i - \epsilon\psi_i$, when $\epsilon = 0$, $y_j = q - p_i$ for all $j\in S_i$. Similarly, the association probabilities $\mathbb{P}(x_j|q) = \mathbb{P}(p_i|q)$ for all $j\in S_i$. Therefore, with the perturbations (17) and the mixed metric (4), the second derivative (16) evaluated at $\epsilon = 0$ can be simplified as follows:

$$\left.\frac{d^2\hat{F}}{d\epsilon^2}\right|_{\epsilon=0} = 2\sum_{j\in S_i}\sum_{k\in C_i}\int_{D_k}\phi(q)\mathbb{P}(p_i|q) \cdot \left[\psi_j^T I\psi_j - \frac{2}{T}\left(\psi_j^T(q-p_i)\right)^2\right] dq\,.$$

Factoring ψ_j from the left and right sides and using the substitution (18), the second derivative evaluated at $\epsilon = 0$ is:

$$\left.\frac{d^2\hat{F}}{d\epsilon^2}\right|_{\epsilon=0} = 2\sum_{j\in S_i}\psi_j^T\mathcal{F}_i\psi_j\,.$$

Thus, in order for an equilibrium configuration to be stable, the matrix quantity in (18) must be positive definite. ■

V. Distributed implementation

We have so far demonstrated how a network of agents can descend the gradient and check for phase changes in a distributed DA algorithm. However, we still must provide a distributed method for implementing these phase changes.

The DA algorithm begins with one active agent, and the other agents moving in formation with it. A formation will split in two if its critical temperature is reached. The agents following in formation are divided evenly between the current formation leader and a new formation leader. After the first phase change, it is possible that future phase changes occur at an agent who is by itself. Therefore, this agent must communicate its desire for an additional companion, and the network of agents must distributively assign an inactive agent to this task. We propose a task-assignment algorithm to accomplish this.

We provide a possible scheme under the following assumptions: (1) Agents have knowledge of the total number of formations n and the total number of agents N, (2) The communication graph between all active agents is connected, (3) Each active agent knows the number of inactive agents traveling with it, and (4) All agents have knowledge of the initial temperature, and the cooling factor α.

Connectivity of the communication graph is important because both the temperature and the total number of active agents must be constant across all agents. We assume that if the graph is connected, the agents can agree on the current temperature, and determine through a flooding algorithm (see [12]) the number of active agents n at any point in time. Additionally, agents must wait for the flooding algorithms to terminate; the worst case is proportional to the diameter of the communication graph.

Algorithm 1: Distributed DA algorithm for each agent

```
$T \leftarrow$ initial temperature
while $T > T_{min}$ or $n < N$ do
    while floodMax($\|\dot{p}_i\|$) $> \epsilon$) do
        $\dot{p}_i \leftarrow -$computeGradient()
    end
    if checkSplit() == true then flood("$T_c$ reached")
    if received "$T_c$ reached" then
        doTaskAssign() $N - n$ times
    end
    $T \leftarrow \alpha T$
end
doNormalCoverage()
```

Algorithm 2: Task assignment algorithm for each agent

```
$a_i \leftarrow$ number of agents in formation
if checkSplit() == true then
    if $a_i == 0$ then
        flood("need companion at $p_i$")
        $M \leftarrow$ positions $p_j$ of replies for help
        if $m_i == null, \forall m_i \in M$ then
            return
        else
            $J \leftarrow$ sortAscending($\{\|p_i - p_j\|\}, j \in M$)
            $j^* \leftarrow$ removeFirst($J$)
            sendMsg("request companion", $j^*$)
            flood("increment $n$ by 1")
        end
    else
        split formation evenly
        flood("increment $n$ by 1")
    end
else                    // no splitting at $p_i$
    $M \leftarrow$ received companion requests $p_j$
    $J \leftarrow$ sortAscending($\{\|p_i - p_j\|\}, j \in M$)
    if $a_i == 0$ then
        sendMsg(null, $\forall j \in J$)
    else
        while length(J) $> 0$ and $a_i > 0$ do
            $j^* \leftarrow$ removeFirst($J$)
            sendMsg("help available from $p_i$", $j^*$)
            $a_i \leftarrow a_i - 1$
        end
    end
end
```

The scheme also uses primitives for flooding or agreement over the network to acquire global information. We define `flood(`msg`)` to be an algorithm that floods a message over the entire network, such that after its completion, each active agent will have knowledge of msg (possibly the `null` message). Messages to a particular agent i can be sent with `sendMsg(`msg`, `i`)`. We also define `floodMax(`x_i`)` as a flooding method to determine $\max_{i\in\{1,\ldots,n\}} x_i$ over the entire network as in [12]. We let `computeGradient()` be the function that computes (11), and we let `checkSplit()` be the function that determines if a critical temperature has been reached as in (18). Finally, we introduce `doNormalCoverage()` to mean to perform limited-range coverage as from [2].

The distributed DA algorithm can informally be described in the pseudocode of Algorithm 1. Starting with a single formation, and a high initial temperature, formations descend the gradient (11). When all agents agree they are stationary, they individually check for phase changes and, if necessary, implement Algorithm 2 $N - n$ times to guarantee the assignment of all companion requests. The temperature is lowered, regardless of whether or not there was a phase change, and the gradient descent is continued. This process repeats until the system temperature is below a minimum temperature threshold T_{min} or if $n = N$. Once this happens, the agents perform the normal coverage algorithm described in [2], as this is equivalent to having $T = 0$.

The pseudocode of Algorithm 2 outlines the task assignment algorithm for agents that are alone but need to split. Roughly speaking, there are three rounds of communication: an agent broadcasts its need for a companion, other agents reply if they can help, and finally a handshake is formed with the agent transfer. In this algorithm, n is incremented for every new formation, and this command is flooded over the network. This algorithm has a finite termination time upper bounded by $3n + n(N - n)$ messages passed.

VI. Simulations

We present a simulation of the limited-range DA algorithm using the mixed centroidal-area sensing function (4). The total number of agents is $N = 6$ and the square region Q has length 10 per side. We will demonstrate the performance of the DA algorithm versus a normal Lloyd-type gradient descent found in [2] for a sensing radius of $R = 3$. The density function ϕ is a sum of 6 Gaussians with different variances and magnitudes as shown in Figure 2.

Due to the smaller sensing radius, initial conditions begin to influence the outcome of the DA algorithm. For this particular choice of ϕ, we have the two possible outcomes shown in Figure 2 for the limited-range DA-algorithm. The better outcome of the DA attains a final cost of -151.5, while the worse outcome reaches a final cost of -110.5.

Next, 50 Lloyd-like gradient descent simulations were run. Each simulation was initialized with a cluster of 6 agents uniformly distributed over a 1×1 square. This square is then randomly placed over the region Q to simulate deployment from a random initial position. Over the 50 random Lloyd-like gradient descent simulations, only 2 reach the configuration shown in Figure 2 (c). The worst case of all 50 trials was a cost of -103.5 shown in Figure 2 (h), and the average cost was -135.3.

Further analysis of this scenario, however, demonstrates that the limited-range DA algorithm still has an advantage over a normal gradient descent algorithm. Figure 2(g) shows the set of initial conditions for which the limited-range DA algorithm converges to the best solution. Note that approximately half of the possible initial condition locations leads to the optimal solution while only 4% of the Lloyd-like gradient descent simulations achieved the same final cost.

The limited-range DA algorithm may have decreased performance versus a normal gradient-descent algorithm. If sensing range is not large enough, as was observed in the previous example, the DA algorithm may fall into a local minimum. Consider the distribution shown in Figure 3(a), where there are two equal Gaussians symmetrically placed at opposite corners of Q. Almost every simulation of the limited-range DA algorithm results in a final configurations like 3(a), or its mirror image. This occurs because the DA algorithm begins with only one agent, and this agent moves towards the nearest Gaussian that it senses and stays there. Then, future phase changes result in only adding more agents around the same Gaussian.

On the other hand, over 50 trials of the Lloyd-like gradient descent with similar initial conditions as before, we see an improved statistic. Only 18 of the 50 simulations fell into the worst-case minima of Figure 3(a). However, none of the simulations were able to converge to the best configuration, which is having 5 agents located around each Gaussian.

A possible way to address this problem of the limited-range DA algorithm is to consider a heating and cooling cycle. Agents can deploy over Q using an area-maximizing technique as in [2]. Thus, agents will tend to move away from each other and cover all of Q, as shown in Figure 3(a). Then, the limited-range DA algorithm is run with a high temperature. This forces agents to collect together about denser parts of Q, shown in Figure 3(b)–(c). Finally, the usual limited-range DA coverage is run, causing agents to split evenly over the important areas of Q, as in Figure 3(d)–(f). Note, however, that this requires a connected communication graph, so agents at the top-left corner of Q must be able to communicate with agents at the bottom-right corner.

VII. Conclusions

We have introduced a limited-range and distributed implementation of the DA algorithm developed by Rose, and applied it to the coverage problem. We developed limited-range results that extend those in [8] and [9]. When the sensing radius is as large as the diameter of Q, this algorithm becomes the normal DA algorithm of Rose. While the limited-range DA algorithm is able to outperform a Lloyd-like gradient descent algorithm in many cases, the algorithm has its limitations as sensing range decreases. Consideration of a heating and cooling cycle produces improved results, but it is still an ad hoc solution to the underlying problem.

VIII. Acknowledgments

The authors would like to thank Jorge Cortes and Francesco Bullo for initial discussions on the use of DA in coverage control algorithms.

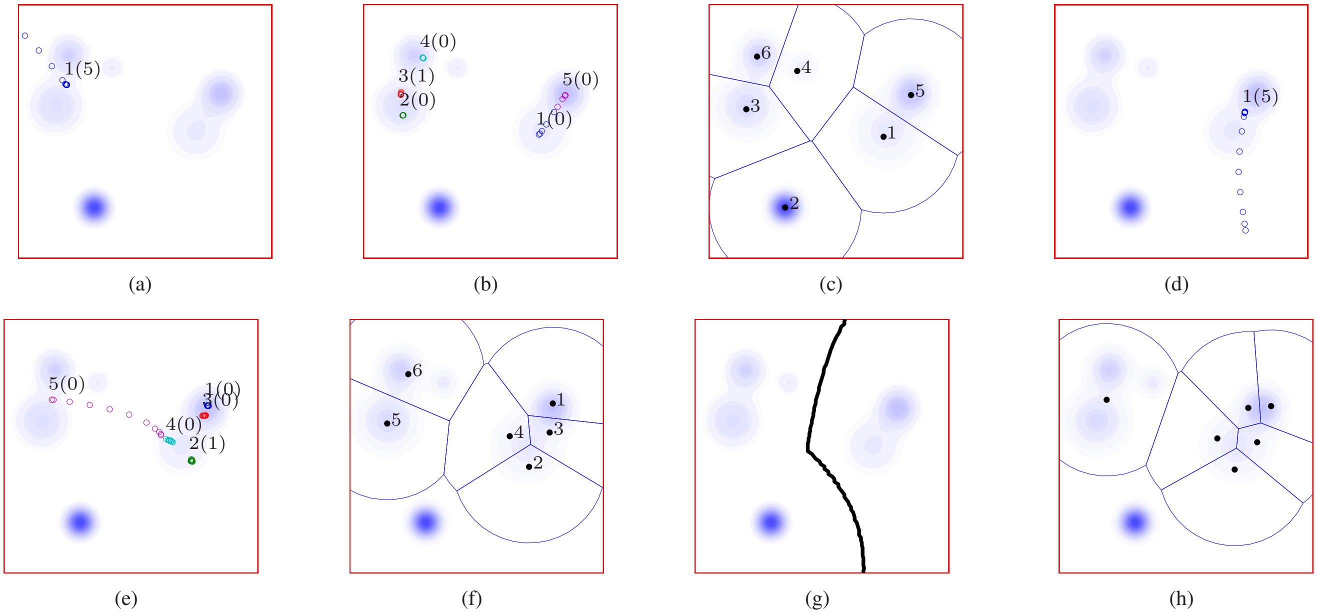
References

[1] R. M. Murray, Ed., *Control in an Information Rich World: Report of the Panel on Future Directions in Control, Dynamics and Systems*. Philadelphia, PA: SIAM, 2003.

[2] J. Cortés, S. Martínez, and F. Bullo, “Spatially-distributed coverage optimization and control with limited-range interactions,” *ESAIM. Control, Optimisation & Calculus of Variations*, vol. 11, pp. 691–719, 2005.

[3] A. Howard, M. J. Matarić, and G. S. Sukhatme, “Mobile sensor network deployment using potential fields: A distributed scalable solution to the area coverage problem,” in *International Conference on Distributed Autonomous Robotic Systems (DARS02)*, Fukuoka, Japan, Jun. 2002, pp. 299–308.

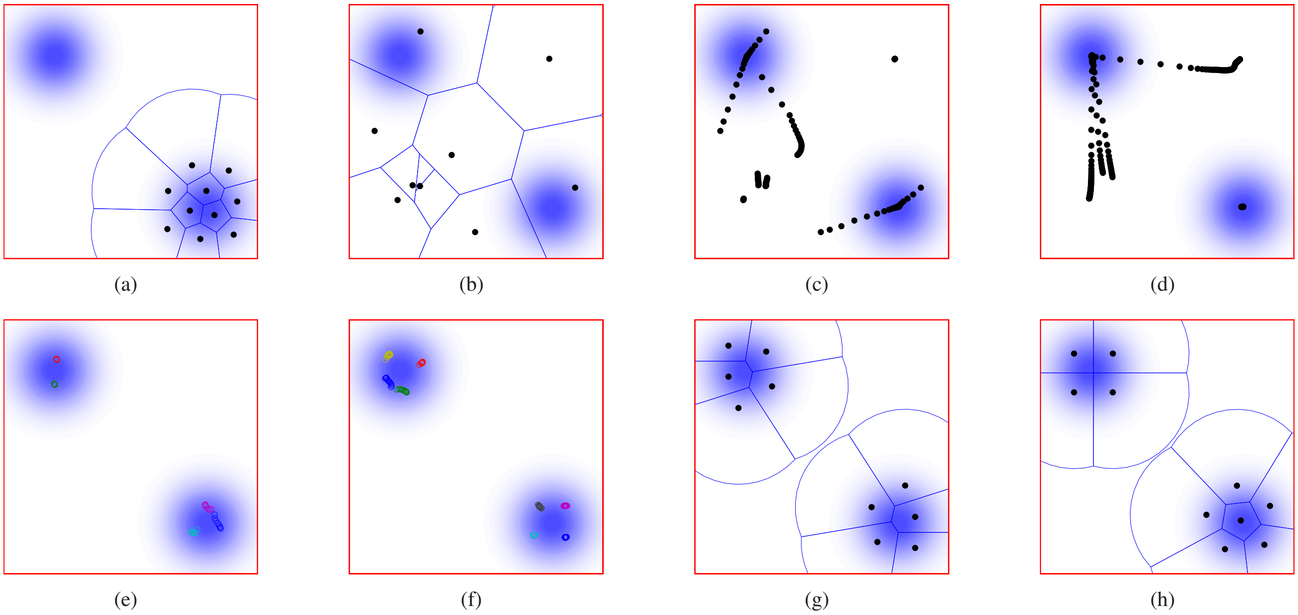
[4] C. Belta and V. Kumar, “Abstraction and control for groups of robots,” *IEEE Transactions on Robotics*, vol. 20, no. 5, pp. 865–875, 2004.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 2. Two runs of the limited-range DA algorithm with $R = 3$. In (a)–(c), the temperature begins at $T = 20$ and decreases: $T = 2.2$ (a), $T = 0.8$ (b), with a final configuration in (c). Similarly in (d)–(f), the temperature begins at $T = 20$ and decreases: $T = 2.0$ (d), $T = 0.5$ (e), with a final configuration in (f). In Figure (g), initial positions of the limited-range DA to the left of the thick black line converge to the configuration shown in (c). Figure (h) shows a worst-case result for the Lloyd-like gradient descent.



(a) (b) (c) (d)

(e) (f) (g) (h)

Fig. 3. A demonstration of a heating and cooling cycle with $R = 3$. A single run of the DA algorithm typically results in configurations like (a). Instead, consider first performing area-maximizing coverage (b), then agents run the limited-range DA algorithm for high temperature, $T = 20$, (c)–(d). Finally, agents perform the usual limited-range DA algorithm in (e)–(g). The best result from a Lloyd-like gradient descent algorithm is shown in (h).

[5] W. Li and C. G. Cassandras, "Distributed cooperative coverage control of sensor networks," in *IEEE Conf. on Decision and Control*, December 2005, pp. 2542–2547.

[6] M. Schwager, J. McLurkin, and D. Rus, "Distributed coverage control with sensory feedback for networked robots," in *Proceedings of Robotics: Science and Systems*, August 2006.

[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.

[8] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proceedings of the IEEE*, vol. 80, no. 11, pp. 2210–2239, 1998.

[9] P. Sharma, S. Salapaka, and C. Beck, "A scalable deterministic annealing algorithm for resource allocation problems," in *American Control Conference*, June 2006, pp. 3092–3097.

[10] W. Xi, X. Tan, and J. S. Baras, "Gibbs sampler-based coordination of autonomous swarms," *Automatica*, vol. 42, no. 7, pp. 1107–1119, July 2006.

[11] S. Martínez, "`http://faemino.ucsd.edu/~soniamartinez/papers/index.html`."

[12] N. A. Lynch, *Distributed Algorithms*. San Mateo, CA: Morgan Kaufmann Publishers, 1997.