A Stochastic Algorithm for Self-Organization of Autonomous Swarms

Wei Xi, Xiaobo Tan, and John S. Baras

Abstract—In earlier work of the authors simulation results indicated the possibility of achieving self-organization of autonomous vehicles through Gibbs sampler-based simulated annealing. However, the dynamic graph structure associated with the network evolution presents challenges in convergence analysis. In this paper a novel algorithm is presented and shown to yield desired global configurations with primarily local interactions. Its convergence speed is provided in terms of the Gibbs potential function. The analytical results are further verified through simulation.

I. INTRODUCTION

With the rapid advances in sensing, communication, computation, and actuation capabilities, autonomous unmanned vehicles (AUVs) are expected to cooperatively perform dangerous or explorative tasks in various hazardous, unknown or remote environments [1]. Distributed methods for control and coordination of vehicles are especially appealing due to large scales of vehicle networks and bandwidth constraints on communication [2], [3], [4], [5]. A popular approach is based on artificial potential functions, which encode desired vehicle behaviors such as inter-vehicle interactions, obstacle avoidance, and target approaching [6], [7], [8], [9], [10]. Vehicles then follow the negative gradients of potentials mimicking the emergent behaviors (e.g. foraging) demonstrated by swarms of bacteria, insects, and animals [11].

The potential function-based approach has been explored for path planning and control of robotic manipulators and mobile robots over the past two decades [12], [13], [14]. Despite its simple, local, and elegant nature, this approach suffers from the problem that the system dynamics could be trapped at the local minima of potential functions [15]. Researchers attempted to address this problem by designing potential functions that have no other local minima [16], [17], or escaping from local minima using ad hoc techniques, e.g., random walk [18], virtual obstacles [19], and virtual local targets [20].

An alternative approach to dealing with the local minima problem was explored using the concept of Markov Random Fields (MRFs) by Baras and Tan [21]. Traditionally used in statistical mechanics and in image processing [22], MRFs were proposed to model swarms of vehicles. Similar to the artificial potential approach, global objectives and constraints (e.g., obstacles) are reflected through the design of potential functions. The movement of vehicles is then decided using simulated annealing based on the Gibbs sampler. Simulations indicated that, with this approach, it was possible to achieve global goals despite the presence of local minima of potentials. However, the dynamic graph structure underlying network evolution presents challenges in convergence analysis. As a first step, two relatively simple cases were analyzed [23]: single-vehicle path planning with limited (i.e., local) sensing and moving ranges, and multi-vehicle path planning with full sensing and moving ranges.

In this paper we present an MRF-based algorithm for self-organization of multiple vehicles. The algorithm requires only limited sensing, communication, and moving ranges for vehicles, and a mechanism for minimal full-range information transfer. Such a mechanism could be provided, e.g., by a dedicated base station, or by individual vehicles with short-time, long-range communication capabilities. It is shown that the algorithm, with primarily local interactions, leads to globally optimal vehicle configurations represented by the global minima of Gibbs potentials. Furthermore, the convergence speed is characterized, providing insight into the design of potential functions. Simulation results are provided to illustrate and verify the analysis.

The remainder of the paper is organized as follows. The concept of MRFs and the problem setup are reviewed in Section II. The distributed algorithm is presented in Section III, and is analyzed in Section IV. Simulation results are presented in Section V. Section VI concludes the paper.

II. PROBLEM SETUP

A. MRFs and Gibbs Sampler

Let \( S \) be a finite set of cardinality \( \sigma \), with elements indexed by \( s \) and called sites. For \( s \in S \), let \( \Lambda_s \) be a finite set called the phase space for site \( s \). A random field on \( S \) is a collection \( X = \{X_s\}_{s \in S} \) of random variables \( X_s \) taking values in \( \Lambda_s \). A configuration of the system is \( x = \{x_s, s \in S\} \) where \( x_s \in \Lambda_s, \forall s \). The product space \( \Lambda_1 \times \cdots \times \Lambda_\sigma \) is called the configuration space. A neighborhood system on \( S \) is a family \( N = \{N_s\}_{s \in S} \), where \( \forall s, r \in S, N_r \subset S, s \notin N_s, \) and \( r \in N_s \) if and only if \( s \in N_r \). \( N_s \) is called the neighborhood of site \( s \). The random field \( X \) is called a Markov random field (MRF) with respect to
the neighborhood system \( \mathcal{N} \) if, \( \forall s \in S, P(X_s = x_s|X_r = x_r, r \neq s) = P(X_s = x_s|X_r = x_r, r \in \mathcal{N}_s) \).

A random field \( X \) is a Gibbs random field if and only if it has the Gibbs distribution:

\[
P(X = x) = \frac{e^{U(x)} Z}{\sum_x e^{U(x)}}, \quad \forall x,
\]

where \( T \) is the temperature variable (widely used in simulated annealing algorithms), \( U(x) \) is the potential (or energy) of the configuration \( x \), and \( Z \) is the normalizing constant, called the partition function: \( Z = \sum_x e^{U(x)} \).

One then considers the following useful class of potential functions \( U(x) = \sum_{s \in \Lambda} \Phi_s(x) \), which is a sum of individual contributions \( \Phi_s \) evaluated at each site. The Hammersley-Clifford theorem [24] establishes the equivalence of a Gibbs random field and a MRF on a finite graph.

The Gibbs sampler belongs to the class of Markov Chain Monte Carlo (MCMC) methods, which sample Markov chains leading to stationary distributions. The algorithm updates the configuration by visiting sites sequentially or randomly with certain proposal distribution [22], and sampling from the local specifications of a Gibbs field. A sweep refers to one round of sequential visits to all sites, or \( \sigma \) random visits under the proposal distribution.

The convergence of the Gibbs sampler was studied by D. Geman and S. Geman in the context of image processing [25]. There it was shown that as the number of sweeps goes to infinity, the distribution of \( X(n) \) converges to the Gibbs distribution \( \Pi \). Furthermore, with an appropriate cooling schedule, simulated annealing using the Gibbs sampler yields a uniform distribution on the set of minimizers of \( U(x) \). Thus the global objectives could be achieved through appropriate design of the Gibbs potential function.

**B. Problem Setup for Self-Organization of Multiple Vehicles**

Consider a 2D mission space (the extension to 3D space is straightforward), which is discretized into a lattice of cells. For ease of presentation, each cell is assumed to be square with unit dimensions. One could of course define cells of other geometries (e.g., hexagons) and of other dimensions (related to the coarseness of the grid) depending on the problems at hand. Label each cell with its coordinates \((i,j)\), where \(1 \leq i \leq N_1, 1 \leq j \leq N_2\), for \(N_1, N_2 > 0\). There is a set of vehicles (or mobile nodes) \( S \) indexed by \( s = 1, \ldots, \sigma \) on the mission space. To be precise, each vehicle (node) \( s \) is assumed to be a point mass located at the center of some cell \((i_s,j_s)\), and the position of vehicle \( s \) is taken to be \( p_s = (i_s,j_s) \). At most one vehicle is allowed to stay in each cell at any time instant.

The distance between two cells, \((i_a,j_a)\) and \((i_b,j_b)\), is defined to be \( R \triangleq \sqrt{(i_a - i_b)^2 + (j_a - j_b)^2} \). There might be multiple obstacles in the space, where an obstacle is defined to be a set of adjacent cells that are inaccessible to vehicles. For instance, a “circular” obstacle centered at \((i^o,j^o)\) with radius \( R^o \) can be defined as \( O \triangleq \{(i,j) : \sqrt{(i - i^o)^2 + (j - j^o)^2} \leq R^o\} \). The accessible area is the set of cells in the mission space that are not occupied by obstacles. An accessible-area graph can then be induced by letting each cell in the accessible area be a vertex and connecting neighboring cells with edges. The mission space is connected if the associated accessible-area graph is connected, which will be assumed in this paper. There can be at most one target area in the space. A target area is a set of adjacent cells that represent desirable destinations of mobile nodes.

In this paper all nodes are assumed to be identical. Each node has a sensing range \( R_s \): it can sense whether a cell within distance \( R_s \) is occupied by some node or obstacle. Communication between two nodes that are within a distance of \( R_s \) is regarded as local. The moving decision of each node \( s \) depends on other nodes located within distance \( R_i \) (\( R_i \leq R_s \)), called the interaction range. These nodes form the set \( \mathcal{N}_s \) of neighbors of node \( s \). A node can travel at most \( R_m \) (\( R_m \leq R_s \)), called moving range, within one move.

![Illustration of the sensing range \( R_s \), the interaction range \( R_i \), and the moving range \( R_m \). Note since the mission space is a discretized grid, a cell is taken to be within a disk if its center is.](image)

The neighborhood system defined earlier naturally leads to a dynamic graph, where each vehicle stands for a vertex of the graph and the neighborhood relation prescribes the edges between vehicles. An MRF can then be defined on the graph, where each vehicle \( s \) is a site and the associated phase space \( \Lambda_s \) is the set of all cells located within the moving range \( R_m \) from location \( p_s \) and not occupied by obstacles or other vehicles. The configuration space of the MRF is denoted as \( \mathcal{X} \).

The Gibbs potential \( U(x) = \sum_s \Phi_s(x) \), where \( \Phi_s(x) \) is considered to be a summation of all clique potentials \( \Psi_{s,r}(x) \), and depends only on \( x_s \) and \( \{x_r, r \in \mathcal{N}_s\} \). The clique potentials \( \Psi_{s,r}(x) \) are used to describe local interactions depending on applications. Specifically,

\[
\Phi_s(x) = \Psi_{s}(x_s) + \sum_{r \in \mathcal{N}_s} \Psi_{s,r}(x_s,x_r). \tag{1}
\]

There are important differences between a classical MRF introduced in Subsection II-A and the MRF defined for the vehicle networks. In a classical MRF, both the phase space \( \Lambda_s \) and the neighborhood \( \mathcal{N}_s \) are time-invariant;
however, for a vehicle network, both $\Lambda_s$ and $\mathcal{N}_s$ depend on the dynamic graph and therefore vary with time. These differences prevent the classical MRF theory from being adopted directly to analyze the convergence behavior of the path planning algorithm.

III. A DISTRIBUTED ALGORITHM

The algorithm to be presented next uses a randomized sequence for updating the nodes, and a key idea involved is the configuration-temperature-dependent proposal distribution $G^T_T(s)$. In particular, given a configuration $x$ and a temperature $T$,

$$
G^T_T(s) = \frac{\sum_{z \in \mathcal{N}_m^e(s)} e^{-\frac{U(x)}{T}}}{\sum_{s' \in S} \sum_{z \in \mathcal{N}_m^e(s')} e^{-\frac{U(x)}{T}}}.
$$

In (2) $\mathcal{N}_m^e(s)$ denotes the set of $e$-neighbors of configuration $x$ within one move:

$$
\mathcal{N}_m^e(s) = \{ z : z_{S_i \setminus s} = x_{S_i \setminus s}, \| z_s - x_s \| \leq R_m \},
$$

where $S_i$'s denotes the set of all nodes except $s$.

Since for $z \in \mathcal{N}_m^e(s)$, $U(z) - U(x) = \Phi_s(z) - \Phi_s(x)$, (2) can be rewritten as:

$$
G^T_T(s) = \frac{\sum_{z \in \mathcal{N}_m^e(s)} e^{-\frac{\Phi_s(z)}{T}}} {\sum_{s'} \sum_{z \in \mathcal{N}_m^e(s')} e^{-\frac{\Phi_s(z)}{T}}} = \frac{\sum_{z \in \mathcal{N}_m^e(s)} e^{-\frac{\Phi_s(z)}{T} - \frac{\Phi_s(x)}{T}}} {\sum_{s'} \sum_{z \in \mathcal{N}_m^e(s')} e^{-\frac{\Phi_s(z)}{T} - \frac{\Phi_s(x)}{T}}}
$$

Note that, from (1), each node $s$ would be able to evaluate $D^T_T(s) = \sum_{z \in \mathcal{N}_m^e(s)} e^{-\frac{\Phi_s(z)}{T} - \frac{\Phi_s(x)}{T}}$ if $R_s \geq R_i + R_m$.

In sampling, node $s$ is first randomly selected with probability $G^T_T(s)$, and then $x_s$ is updated according to its local characteristics while $x_{S_i \setminus s}$ is kept fixed:

$$
P(x_s = l) = \frac{e^{-\frac{\Phi_s(x_{S_i \setminus s} = l)}{T}}}{\sum_{l' \in C_m^e} e^{-\frac{\Phi_s(x_{S_i \setminus s} = l')}}},
$$

where $C_m^e$ is the set of candidate locations node $s$ can take, i.e., $l \in C_m^e$ is not occupied by any obstacle or other nodes, and $\| x_s - l \| \leq R_m$. One can verify that there exists a smallest integer $\tau$, such that after $\tau$ steps of sampling, any configuration $x$ has a positive probability of becoming any other configuration $y$.

The self-organization algorithm works as follows. Pick an appropriate cooling schedule $T(n)$ with $T(n) \to 0$ as $n \to \infty$. Pick a sufficiently large $N_{\text{max}}$. For each temperature $T(n)$, run $\tau$ steps of sampling as described above (this will be called one annealing step). To be specific:

- **Step 1. Initialization.** Start with an arbitrary configuration $x(0)$ and let $n = 1$, $k = 1$. Pick an arbitrary node $s(0)$. Have all nodes to evaluate and send $D^T_T(s(0))$ to $s(0)$. Node $s(0)$ calculates the proposal distribution $G^{T(0)}_{T(1)}(s(0))$ according to (2), namely,

$$
G^{T(0)}_{T(1)}(s(0)) = \frac{D^T_T(s(0))}{\sum_{s'} D^T_T(s')}
$$

Node $s(0)$ then selects a node $s^{(1)}(1)$ for updating by sampling the distribution $G^{T(0)}_{T(1)}(s(0))$, and it sends the vector $\{D^T_T(s^{(1)}(1), s) \in S\}$ to $s^{(1)}(1)$;

- **Step 2. Updating the selected node.** Node $s^{(k)}(n)$ updates its location by sampling its local characteristics (see (3)). Denote the new configuration as $x^{(k)}(n)$;

- **Step 3. Selecting the next node.** Note that the neighborhood $\mathcal{N}_s$ of a node $s$ changes only if node $s^{(k)}(n)$ was in $\mathcal{N}_s$ before its updating or is currently in $\mathcal{N}_s$. For either case, the distance between such $s$ (denoting the set of such nodes as $\mathcal{N}^{k}(n)$) and $s^{(k)}(n)$ is now no greater than $R_i + R_m \leq R_s$ and they can communicate locally. The node $s^{(k)}(n)$ thus collects and updates $D^{(k)}_{T(n)}(s)$ for nodes in $\mathcal{N}^{k}(n)$.

Let $k = k + 1$. If $k = \tau$, let $k = 0$ and $n = n + 1$. The current node evaluates and samples the new proposal distribution, selects the next node to be updated, and communicates the updated $\{D^{(k)}_{T(n)}(s)\}$ to the next node (the superscript of $D$ is omitted when it is clear from the context);

- **Step 4. If $n < N_{\text{max}}$, go to Step 2; otherwise quit.**

Remark 3.1: Long-range (over a distance greater than $R_s$) communication is only required for initialization and for transferring $\{D^T_T(s)\}$ to the newly selected node at each step. Since $\{D^T_T(s)\}$ is just a $\sigma$-dimensional vector, information exchange in the algorithm is primarily at the local level. The (minimal) global communication can be achieved through, e.g., fixed base stations, or individual vehicles with short-time, long-range transmission capabilities.

IV. CONVERGENCE ANALYSIS

Let $P_T$ denote the Markov kernel defined by the random update scheme (2) and (3), i.e.,

$$
P_T(x, y) \triangleq \Pr(X(n + 1) = y | X(n) = x)
$$

$$
= \sum_{s \in S} G^T_T(s) \cdot 1(y \in \mathcal{N}_m^e(s)) e^{-\frac{U(y)}{T}}
$$

$$
= \sum_{s \in S} \sum_{z \in \mathcal{N}_m^e(s)} e^{-\frac{\Phi_s(z)}{T} - \frac{\Phi_s(x)}{T}} \cdot 1(y \in \mathcal{N}_m^e(s)) e^{-\frac{U(y)}{T}}
$$

$$
= \sum_{s \in S} \sum_{s' \in S} \sum_{z \in \mathcal{N}_m^e(s')} e^{-\frac{\Phi_s(z)}{T} - \frac{\Phi_s(x)}{T}} \cdot 1(y \in \mathcal{N}_m^e(s)) e^{-\frac{U(y)}{T}}.
$$

Let $\tau$ be the integer as selected in Section III, and let $Q_T = P_T^{\tau}$.

Theorem 4.1: The Markov kernel $Q_T$ has a unique stationary distribution $\Pi_T$ with

$$
\Pi_T(x) = \frac{e^{-\frac{U(x)}{T}} \sum_{s \in S} \sum_{z \in \mathcal{N}_m^e(s)} e^{-\frac{U(z)}{T}}}{Z_T},
$$

where $Z_T = \sum_y e^{-\frac{U(y)}{T}} \sum_{s \in S} \sum_{z \in \mathcal{N}_m^e(s)} e^{-\frac{U(z)}{T}}$ is the partition function.

1In the notation $x^{k}(n)$ or $s^{(k)}(n)$, $n$ indexes the annealing temperature, while $k$ (from 1 to $\tau$) indexes the sampling step within a fixed temperature.
Proof. First one can show that $\Pi_T$ is a stationary distribution of $P_T$. From (4) and (5),
\[
\sum_y \Pi(y) P_T(y, x) = \sum_y e^{-\frac{U(y)}{n}} \sum_{s' \in S} \sum_{s \in N(x)^c_{n_m}} e^{-\frac{U(s)}{n}} Z_T,
\]
\[
\sum_{s \in S} s' \in S \sum_{s \in N_{n_m}(s')} e^{-\frac{U(s)}{n}} Z_T = \sum_{s \in S} s' \in S \sum_{s \in N(x)^c_{n_m}} e^{-\frac{U(s)}{n}} Z_T,
\]
\[
= e^{-\frac{U(x)}{n}} \sum_y e^{-\frac{U(y)}{n}} \sum_{s \in N_{n_m}(s)} \sum_{s \in S} 1(x \in N_{n_m}(s)) = e^{-\frac{U(x)}{n}} \sum_y e^{-\frac{U(y)}{n}} \sum_{s \in S} 1(x \in N_{n_m}(s))
\]
\[
= e^{-\frac{U(x)}{n}} \sum_{s \in S} e^{-\frac{U(x)}{n}} \sum_{s \in N_{n_m}(s)} e^{-\frac{U(s)}{n}} Z_T = \Pi_T(x).
\]
Since $Q_T = P_T$, $\Pi_T$ is also a stationary distribution for $Q_T$. Due to the choice of $\tau$, $Q_T(x, y) > 0, \forall x, y$. Thus from the Perron-Frobenius theorem, $Q_T$ has a unique stationary distribution, which is $\Pi_T$. □

Let $\Delta$ be the maximal local oscillation of the potential $U$:
\[
\Delta = \max_x \max_{y \in N_{n_m}^c} |U(x) - U(y)|,
\]
where $N_{n_m}^c = \bigcup_{s \in S} N_{n_m}(s)$.

**Theorem 4.2.** Let $\{T(n)\}$ be a cooling schedule decreasing to 0 such that eventually,
\[
T(n) \geq \frac{\tau \Delta}{\ln n}.
\]

Let $Q_n = P_{T(n)}$, and let $M$ be the set of global minima of $U(\cdot)$. Then for any initial distribution $\nu$,
\[
\lim_{n \to \infty} \nu^n Q_1 \cdots Q_n \to \nu_{\infty},
\]
where $\nu_{\infty}$ is the distribution (5) evaluated at $T = 0$. In particular,
\[
\sum_{x \in M} \nu_{\infty}(x) = 1.
\]

**Proof.** Let $\alpha_x = \min_{y \in N_{n_m}^c} U(y)$. For $y \in N_{n_m}^c$, from (4),
\[
P_T(x, y) = \frac{e^{-\frac{U(y)}{n}} - \alpha_x}{\sum_{s' \in S} \sum_{s \in N_{n_m}(s')} e^{-\frac{U(s)}{n}}} \geq e^{-\frac{\lambda}{\sigma|\mathcal{X}|}},
\]
where $|\mathcal{X}|$ denotes the cardinality of the configuration space $\mathcal{X}$. For $Q_T = P_T > 0$,
\[
\min_{x, y} Q_T(x, y) \geq (\min_{x', y' \in N_{n_m}^c} P_T(x', y')) \geq e^{-\frac{\lambda}{\sigma|\mathcal{X}|}}.
\]

Let $C(Q_T)$ denote the contraction coefficient [22] of $Q_T$, i.e.,
\[
C(Q_T) \triangleq \frac{1}{2} \max_{x, y} \|Q_T(x, \cdot) - Q_T(y, \cdot)\|_1.
\]

Using Lemma 4.2.3 of [22], one has
\[
C(Q_T) \leq 1 - |\mathcal{X}| \min_{x, y} Q_T(x, y) \leq 1 - \lambda e^{-\frac{\lambda}{\sigma|\mathcal{X}|}},
\]
where $\lambda = \frac{|\mathcal{X}|}{\sigma|\mathcal{X}|} < 1$. This implies $C(Q_n) < 1 - \lambda e^{-\frac{\lambda}{\sigma|\mathcal{X}|}}$. The claim (6) can then be proved following the proof of Theorem 3.2 in [23]. As $T(n) \to 0$, $\Pi_{T(n)}(x) \to 0$, for all $x \notin M$, as one can verify from (5). Eq. (7) thus follows. □

From Theorem 4.2, the distributed algorithm can achieve global objectives provided that the global minimizers of $U(\cdot)$ correspond to the desired configurations.

Let $\tilde{m} = \min_{x \in M} U(x) - m$, i.e., the minimal potential difference between other configurations and the global minimizer. The following result characterizes the convergence speed of the distributed algorithm:

**Proposition 4.1.** Consider the distributed self-organization algorithm with a cooling schedule $T(n) = \frac{\tau \Delta}{\ln n}$. Then the following estimate holds for any initial distribution $\nu$:
\[
\|\nu Q_1 \cdots Q_n - \Pi_{\infty}\| = O(n^{-\frac{2\lambda}{2m + \lambda \tau}}) = O(n^{-g}),
\]
where $\lambda, \tau$, and $\Delta$ are as defined in Theorem 4.2, and $g = \frac{2\lambda}{2m + \lambda \tau}$ is called the indicator of convergence speed in this paper.

The proposition is very similar to the result for the single-vehicle case [23], and its proof is omitted here in the interest of space. The dependence of the indicator $g$ on the potential function $U$ could be exploited to speed up the convergence.

**V. SIMULATION RESULTS**

Simulations were conducted to verify the analysis in the previous section. The emphasis was on scenarios involving inter-vehicle interactions (e.g., formation control). Two examples are presented, one on clustering and the other on formation control. Other objectives or constraints, such as target-approaching and obstacle avoidance, can be easily incorporated, as was done in the single-vehicle case [23].

**A. Clustering**

The goal is to cluster all the nodes without specifying a specific target area. This is more challenging than the case of having an explicit target, as the latter provides persistent attraction from a fixed location. The potential function used was:
\[
U(x) = \sum_{r \neq x, \|x_r - x_s\| \leq R_s} - \frac{c}{\|x_r - x_s\|},
\]
where $c$ is some constant. Clearly, the more neighbors each node has and the closer they are, the lower the potential $U$. Simulation was performed for 50 nodes on a 30 by 30 grid, and the following parameters were used: $R_l = 4\sqrt{2} + \epsilon$ ($\epsilon > 0$ and very small), $R_m = 2\sqrt{2} + \epsilon$, $R_s = R_l + R_m$ (this was also the case for all other simulation runs), $c = 2$, annealing schedule $T(n) = \frac{1}{0.08 \ln n}$, and $\tau = 50$.

Fig. 2 shows the snapshots of the network evolution. The algorithm’s ability to overcome local minima is evident from the figure: the nodes initially evolved into two separated (farther than $R_s$) sub-clusters, and yet they merged into one cluster after 500 annealing steps.
B. Formation

The goal is to have the nodes to form (square) lattice structures with a desired inter-vehicle distance $R_{des}$. The potential function used was:

$$U(x) = \sum_{r \neq s, \|x_r - x_s\| \leq R_i} c_1(\|x_r - x_s\| - R_{des})^\alpha - c_2,$$

where $c_1 > 0$, $c_2 > 0$, and $\alpha > 0$. A proper choice of $c_2$ encourages nodes to have more neighbors. The power $\alpha$ shapes the potential function. In particular, for $\|x_r - x_s\| - R_{des} < 1$, smaller $\alpha$ leads to larger potential difference from the global minimum.

First, simulations were conducted for 9 nodes on an 8 by 8 grid. Parameters used were: $R_i = 4\sqrt{2} - \epsilon$, $R_m = 2\sqrt{2} + \epsilon$, $R_{des} = 2$, $c_1 = 10$, $c_2 = 1.05$, $\alpha = 0.02$, $T(n) = \frac{1}{0.01 \ln n}$, and $\tau = 20$. The desired configuration (global minimizer of $U$) is shown in Fig. 3 (modulo vehicle permutation and formation translation on the grid). Simulated annealing was performed for $10^4$ steps. Empirical distributions with respect to configuration potentials were calculated based on the average of every 2,500 steps (Fig. 4). The trend of convergence to lowest potential is clear from Fig. 4. One can further calculate the error $\|\nu_n - \Pi_\infty\|_1$, where $\nu_n$ is the empirical distribution of configurations (again modulo vehicle permutation and network translation), and

$$\Pi_\infty(x) = \begin{cases} 1 & \text{if } x \text{ is desired} \\ 0 & \text{otherwise} \end{cases}.$$ 

Therefore,

$$\|\nu_n - \Pi_\infty\|_1 = 1 - \nu_n(x^*) + |0 - (1 - \nu_n(x^*))| = 2(1 - \nu_n(x^*)),$$

where $x^*$ denotes the desired formation. The evolution of $\|\nu_n - \Pi_\infty\|_1$ is shown in Fig. 5, where $\nu_n(x^*)$ is calculated as the relative frequency of sampling $x^*$ in 1000 annealing steps.

VI. CONCLUSIONS AND DISCUSSIONS

In this paper a distributed stochastic algorithm was presented for self-organization of multiple vehicles. The algorithm was based on Gibbs-sampler with a random visiting scheme. The specific choice of the proposal distribution results in Gibbs-type distributions for vehicle configurations, leading to the convergence of the algorithm. Simulation
results were shown to demonstrate the performance of the algorithm and verify the analysis.

The random visiting scheme entails (possibly) long-range communications for notifying newly selected nodes although such information exchange is minimal. A deterministic sequential visiting scheme would eliminate this requirement, however, the convergence behavior would be unclear since the stationary distribution for each $T$ is no longer of the Gibbs-type.

The self-organization scheme is meant to be a high-level planning algorithm. It should be combined with low-level planning and path tracking control modules in implementation. For example, the inter-vehicle collision could be avoided by adding gyroscopic force in low-level control modules. In this paper the mission space was discretized into a lattice with square cells. One could use cells of other geometries without changing the algorithm (except the numbering scheme for cells) to implement, e.g., triangular or hexagonal formations. One could also increase the number of cells to improve the resolution of local maps. For future work, it would be interesting to investigate the robustness of the algorithm when uncertainties in sensing exist.

REFERENCES


