2015 American Control Conference
Palmer House Hilton
July 1-3, 2015. Chicago, IL, USA

Team-triggered coordination of robotic networks for optimal deployment

Cameron Nowzari Jorge Cortés George J. Pappas

Abstract— **This paper introduces a novel team-triggered algorithmic solution for a distributed optimal deployment problem involving a group of mobile sensors. Distributed self-triggered algorithms relieve the requirement of synchronous periodic communication among agents by providing opportunistic criteria for when communication should occur. However, these criteria are often conservative since worst-case scenarios must always be considered to ensure the monotonic evolution of a relevant objective function. Here we introduce a team-triggered algorithm that builds on the idea of 'promises' among agents, allowing them to operate with better information about their neighbors when they are not communicating, over a dynamically changing graph. We analyze the correctness of the proposed strategy and establish the same convergence guarantees as a coordination algorithm that assumes perfect information at all times. The technical approach relies on tools from set-valued stability analysis, computational geometry, and event-based systems. Simulations illustrate our results.**

I. Introduction

This paper considers a robotic sensor network performing a distributed deployment task over a region of interest. Similar works often assume agents have continuous or period communication with one another at all times to perform their desired task. This can often be undesirable, especially as the size of the network grows large, since it is a waste of communication bandwidth that might be shared across other systems or networks. More recently, event- and self-triggered coordination strategies have been studied to relax this requirement by giving agents more autonomy, allowing them to decide among themselves when communication should occur. Our objective is to design a team-triggered coordination strategy for the deployment problem that combines ideas from both event- and self-triggered into a unified approach that enjoys benefits from both strategies.

Literature review: This work builds on coverage control problems for sensor networks developed in [1], where distributed algorithms based on centroidal Voronoi partitions are presented. Other works on deployment problems include [2], [3]. A common assumption in the above works is that agents have access to constant communication with their neighbors at all times. Our main goal is to relax this assumption by providing agents with sufficient levels of autonomy. A related line of work that addresses this issue is the study of event- and self-triggered controllers, particularly in distributed setups. In these works, agents are given criteria to determine when their control signals should be updated rather than doing this continuously. These ideas have been applied to various tasks including consensus via event-triggered [4], [5], [6] or self-triggered control [4], [7], rendezvous [8], model predictive control [9], and model-based event-triggered control [10], [11]. We are particularly interested in works that design distributed triggering strategies not only for the controller, but for when communication is required, e.g. [10], [12], [13]. In [10], agents are responsible for monitoring not only their own estimates, but estimates that other agents have to ensure they stay within some performance bounds. In [12], [13], agents autonomously decide when it is necessary to broadcast new information to their neighbors. In [14], the authors propose a self-triggered strategy to relax the constant communication requirement in [1]. This is done by bounding the distance agents can move in a given time frame and utilizing outdated information to determine when fresh information is required. The drawback of this strategy is that, in order to ensure the monotonic evolution of a relevant objective function, agents must consider worst-case conditions at all times to ensure they are receiving updates frequently enough to complete the given task. In this work we aim for similar goals by employing the team-triggered coordination approach introduced in [15].

Statement of contributions: This paper builds on a deployment algorithm where agents utilize a self-triggered coordination strategy to decide when updated information from their neighbors is required. Our main contribution is the development of a modified version of a team-triggered algorithm for the deployment problem. Unlike prior work in team-triggering that has only considered static communication topologies, we consider here a dynamic graph that depends on agent positions and when communication is required. A dynamic communication graph requires a nontrivial treatment in the context of team-triggering because agents are generally unaware of whether the topology has changed at any given time. We are able to characterize communication requirements using the geometric properties that determine the communication graph. Additionally, we utilize a controller that operates on set-valued information rather than points to make the most out of the information available to the agents. We analyze the correctness of the proposed algorithm and establish the same convergence guarantees as a coordination algorithm that assumes perfect information is available at all times. The technical approach combines elements from set-valued stability analysis, computational geometry, and event-based systems.

II. Preliminaries

We let $\mathbb{R}_{\geq 0}$ and $\mathbb{Z}_{\geq 0}$ be the sets of nonnegative real and integer numbers, respectively, and $\|\cdot\|$ the Euclidean distance.

The authors are with the Dept. of Electrical and Systems Engineering, University of Pennsylvania, and the Dept. of Mechanical and Aerospace Engineering, University of California, San Diego, {cnowzari,pappasg}@seas.upenn.edu, cortes@ucsd.edu. Work supported by NSF award CNS-1329619 and ONR-MURI HUNT award N00014-08-1-0696.

We denote by $[p,q] \subset \mathbb{R}^d$ the closed line segment with extreme points p and $q \in \mathbb{R}^d$. Let $\phi : \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ be a bounded measurable function that we term *density*. For $S \subset \mathbb{R}^d$, the *mass* and *center of mass* of S with respect to ϕ are

$$M_S = \int_S \phi(q)dq, \qquad C_S = \frac{1}{M_S}\int_S q\phi(q)dq.$$

The *circumcenter* of $S \subset \mathbb{R}^d$ is the center of the closed ball of minimum radius that contains S. The *circumradius* $\mathrm{cr}(S)$ is the radius of this ball. We denote by $\overline{B}(p,r)$ the closed ball centered at $p \in S$ with radius r. Given $v \in \mathbb{R}^d \setminus \{0\}$, let $\mathrm{unit}(v)$ be the unit vector in the direction of v.

A. Voronoi partitions

We refer to [16] for a comprehensive treatment on Voronoi partitions and briefly present some relevant concepts here. Let S be a convex polygon in $\mathbb{R}^2$ and $P = (p_1, \dots, p_n)$ be the location of n sensors. A *partition* of S is a collection of n polygons $\mathcal{W} = \{W_1, \dots, W_n\}$ with disjoint interiors whose union is S. The *Voronoi partition* $\mathcal{V}(P) = \{V_1, \dots, V_n\}$ of S generated by the points $P = (p_1, \dots, p_n)$ is

$$V_i = \{q \in S \mid \|q - p_i\| \leq \|q - p_j\|, \forall j \neq i\}.$$

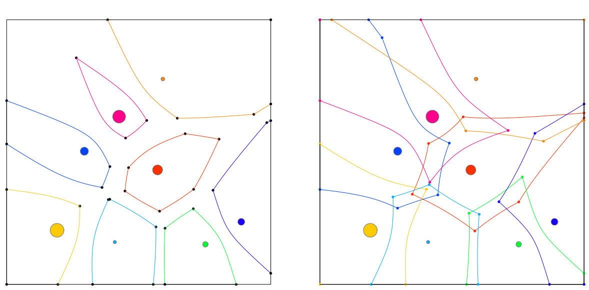
When the Voronoi regions V_i and V_j are adjacent (i.e., they share an edge), p_i is called a *(Voronoi) neighbor* of p_j (and vice versa). We denote the neighbors of agent i by $\mathcal{N}_i$. $P = (p_1, \dots, p_n)$ is a *centroidal Voronoi configuration* if it satisfies that $p_i = C_{V_i}$, for all $i \in \{1, \dots, N\}$.

B. Space partitions with uncertain information

Following [17], [18], [14], consider regions $D_1, \dots, D_N \subset S$, each containing a site $p_i \in D_i$. The *guaranteed Voronoi diagram* of S generated by $D = (D_1, \dots, D_N)$ is the collection $\mathrm{g}\mathcal{V}(D_1, \dots, D_N) = \{\mathrm{g}V_1, \dots, \mathrm{g}V_N\}$,

$$\mathrm{g}V_i = \{q \in S \mid \max_{x \in D_i} \|q - x\| \leq \min_{y \in D_j} \|q - y\| \text{ for all } j \neq i\}.$$

We define the ith component of $\mathrm{g}\mathcal{V}(D_1, \dots, D_N)$ as $\mathrm{g}V_i(D)$. Note that $\mathrm{g}V_i$ contains the points of S that are guaranteed to be closer to p_i than to any other of the nodes p_j, $j \neq i$. The guaranteed Voronoi diagram is not a partition of S, see Figure 1(a). If every region D_i is a point, $D_i = \{p_i\}$, then $\mathrm{g}\mathcal{V}(D_1, \dots, D_N) = \mathcal{V}(p_1, \dots, p_N)$. For any collection of points $p_i \in D_i$, the guaranteed Voronoi diagram is contained in the Voronoi partition, i.e., $\mathrm{g}V_i \subset V_i$, $i \in \{1, \dots, N\}$. Similarly, the *dual guaranteed Voronoi diagram* of S generated by $D_1, \dots, D_N$ is the collection of sets $\mathrm{dg}\mathcal{V}(D_1, \dots, D_N) = \{\mathrm{dg}V_1, \dots, \mathrm{dg}V_N\}$ defined by



(a) (b)

Fig. 1. Guaranteed (a) and dual guaranteed (b) Voronoi diagrams.

$$\mathrm{dg}V_i = \{q \in S \mid \min_{x \in D_i} \|q - x\| \leq \max_{y \in D_j} \|q - y\| \text{ for all } j \neq i\}.$$

For any collection of points $p_i \in D_i$, the dual guaranteed Voronoi diagram is guaranteed to contain the Voronoi partition, i.e., $V_i \subset \mathrm{dg}V_i$, $i \in \{1, \dots, N\}$.

III. Problem statement

Consider a group of agents moving in a convex polygon $S \subset \mathbb{R}^2$ with positions $P = (p_1, \dots, p_N)$. We consider single integrator dynamics

$$\dot{p}_i = u_i \tag{1}$$

where $\|u_i\| \leq v_{\max}$ for all $i \in \{1, \dots, N\}$ for some $v_{\max} > 0$, i.e., $u_i \in \overline{B}(0, v_{\max})$. For simplicity, we assume all agents are able to take actions such as computing control signals synchronously at a fixed period $\Delta t > 0$. All results provided in the paper still hold without this assumption.

Following [1], the objective is to achieve optimal deployment with respect to the aggregate distortion $\mathcal{H}$. The performance at q of agent p_i degrades with $\|q - p_i\|^2$. Assume a density $\phi : S \to \mathbb{R}$ is available, with $\phi(q)$ reflecting the likelihood of an event happening at q. Letting $P \in S^N$ denote the set of agent positions, consider the minimization of

$$\mathcal{H}(P) = E_\phi \left[\min_{i \in \{1, \dots, N\}} \|q - p_i\|^2 \right]. \tag{2}$$

This function is useful when an agent closest to an event is responsible for addressing it. Examples include servicing tasks, spatial sampling of random fields, resource allocation, and event detection, see [19], [20] and references therein. Note that if we define, with a slight abuse of notation,

$$\mathcal{H}(P, \mathcal{W}) = \sum_{i=1}^{N} \int_{W_i} \|q - p_i\|^2 \phi(q)dq, \tag{3}$$

where $\mathcal{W}$ is a partition of S, and the ith agent is responsible for the region W_i, then $\mathcal{H}(P, \mathcal{V}(P))$ corresponds to the aggregate distortion function defined in (2). Hence, the function $\mathcal{H}$ is then to be minimized with respect to the locations P and the regions $\mathcal{W}$. Interestingly, one can show [19], [1] that, given $P \in S^N$ and a partition $\mathcal{W}$ of S,

$$\mathcal{H}(P, \mathcal{V}(P)) \leq \mathcal{H}(P, \mathcal{W}), \tag{4}$$

i.e., the optimal partition is the Voronoi partition. Moreover, for $P' \in S^N$ with $\|p'_i - C_{W_i}\| \leq \|p_i - C_{W_i}\|$, $i \in \{1, \dots, N\}$,

$$\mathcal{H}(P', \mathcal{W}) \leq \mathcal{H}(P, \mathcal{W}), \tag{5}$$

i.e., the optimal sensor positions are the centroids. The algorithmic solutions to optimize in a distributed way the objective function $\mathcal{H}$ rely strongly on this observation. Our objective here is to synthesize an efficient coordination strategy to optimize this function that employs opportunistic state-triggered communication to minimize energy expenditure and yet enjoys good performance guarantees.

IV. Periodic and self-triggered algorithms

This section briefly reviews the algorithmic solutions to the problem stated in Section III based on periodic and self-triggered communication, respectively.

A. Periodic algorithmic solution

The periodic-communication distributed coordination strategy proposed in [1] is based on the properties of the Voronoi partition, and more specifically, on the optimality characterizations provided by (4) and (5). Basically, each agent periodically and synchronously communicates with its neighbors its state information, computes the centroid of its own Voronoi cell, and moves towards it. The executions of the resulting algorithm are asymptotically guaranteed [1] to converge to the set of centroidal Voronoi configurations.

B. Self-triggered algorithmic solution

Here we review important elements of the self-triggered deployment algorithm proposed in [14]. We begin by introducing the data structure agents maintain about one other given updated position information.

1) Agent data structure: Let t_ℓ^i be a time at which agent i has just received position information $p_j^i = p_j(t_\ell^i)$ from another agent j. Then, at time $t \geq t_\ell^i$ agent i knows that agent j has not moved farther than $r_j^i = v_{\max}(t - t_\ell^i)$ from p_j^i. This information that agent i maintains about agent j is

$$\mathbf{X}_j^i(t) = \overline{B}(p_j^i, r_j^i) \cap S. \tag{6}$$

We refer to this as a *guaranteed set* because, given the dynamics (1), this set has the property that $p_j(t) \in \mathbf{X}_j^i(t)$ for all $t \geq t_\ell^i$. The data is stored in

$$\mathcal{D}^i(t) = (\mathbf{X}_1^i(t), \ldots, \mathbf{X}_N^i(t)) \subset S^N. \tag{7}$$

Note that agent i may not necessarily have information about all agents $j \in \{1, \ldots, N\}$. In the case that agent i does not have any information about some agent j, let $\mathbf{X}_j^i(t) = \emptyset$. We refer to $\mathcal{D} = (\mathcal{D}^1, \ldots, \mathcal{D}^N) \subset S^{N^2}$ as the entire memory of the network.

Given the above data structure, we are now able to present the two components of the self-triggered algorithm: a motion control part that determines the best way to move given the available information and an update decision part that determines when new information is needed.

2) Motion control: If an agent had perfect knowledge of other agents' positions, then to optimize $\mathcal{H}$, it could compute its own Voronoi cell and move towards its centroid, as in [1]. Since this is not the case we are considering, we instead use an alternative motion control. The following result gives a condition under which an agent can get closer to the centroid of its own Voronoi cell with uncertain information.

Proposition IV.1 (Motion control [14]) *Given the position information p_i of agent i and the data $\mathcal{D}^i$ available to it, let*

$$\mathrm{bnd}_i \equiv \mathrm{bnd}(gV_i, dgV_i) = 2\,\mathrm{cr}(dgV_i)\Big(1 - \frac{M_{gV_i}}{M_{dgV_i}}\Big). \tag{8}$$

If for $p' \in (p_i, C_{gV_i}]$,

$$\|p' - C_{gV_i}\| \geq \mathrm{bnd}_i, \tag{9}$$

then $\|p' - C_{V_i}\| < \|p_i - C_{V_i}\|$.

Exploiting Proposition IV.1, we are able to come up with a motion control law given uncertain information. Intuitively, agent i uses its currently stored information about other agents' locations to calculate its own guaranteed and dual guaranteed Voronoi cells. It then moves towards the centroid of its guaranteed Voronoi cell until it is within distance bnd_i of it. Note that this law assumes that each agent has access to the value of the density ϕ over its guaranteed Voronoi cell. This yields the motion control law computed at time t_ℓ

$$u_i^*(t_\ell) = v_i\,\mathrm{unit}(C_{\mathrm{g}V_i} - p_i), \tag{10}$$

where

$$v_i = \begin{cases} v_{\max}, & \text{if } \|p_i - C_{\mathrm{g}V_i}\| \geq \mathrm{bnd}_i + v_{\max}\Delta t, \\ 0, & \text{if } \|p_i - C_{\mathrm{g}V_i}\| \leq \mathrm{bnd}_i, \\ \frac{\|C_{\mathrm{g}V_i} - p_i\| - \mathrm{bnd}_i}{\Delta t}, & \text{otherwise.} \end{cases}$$

The Motion Control Law is formalized in Algorithm 1.

Algorithm 1: Motion Control Law

Agent $i \in \{1, \ldots, N\}$ performs:
1: set $D = \mathcal{D}^i$
2: compute $L = \mathrm{g}V_i(D)$ and $U = \mathrm{dg}V_i(D)$
3: compute $q = C_L$ and $r = \mathrm{bnd}(L, U)$
4: compute u_i^* as defined in (10)

Given the result of Proposition IV.1, agent i can simply move towards its computable $C_{\mathrm{g}V_i}$ and guarantee to be decreasing the value of the optimization function $\mathcal{H}$ as long as (9) is satisfied. As time elapses without new information, the bound bnd_i begins to grow until (9) is no longer satisfied. This gives a natural triggering condition for when agent i needs updated information from its neighbors. We discuss this next.

3) Self-triggered update policy: To specify this component, we build on the discussion of the previous section, specifically on making sure that condition (9) is feasible.

The update policy is described informally as follows. Each agent uses its stored information about other agents' locations to calculate its own guaranteed Voronoi and dual guaranteed Voronoi cells, and the bound (8). Then, it decides that up-to-date location information is required if its computed bound is larger than the distance to the centroid of its guaranteed Voronoi cell and ε, where $\varepsilon > 0$ is an a priori chosen design parameter to limit updates when agents are near convergence.

Formally, the self-triggered updating mechanism followed by each agent is described in Algorithm 2.

The self-triggered deployment algorithm is then the combination of Algorithms 1 and 2 above. The synthesized algorithm has been shown to asymptotically converge to the set of centroidal Voronoi configurations [14].

Algorithm 2: Self-Triggered Update Policy

```
Agent i ∈ {1, ..., N} performs:
set D = D^i
compute L = gV_i(D) and U = dgV_i(D)
compute q = C_L and r = bnd(L, U)
(Requesting new information)
if r ≥ max{||q − p_i||, ε} then
    request updated information
end if
```

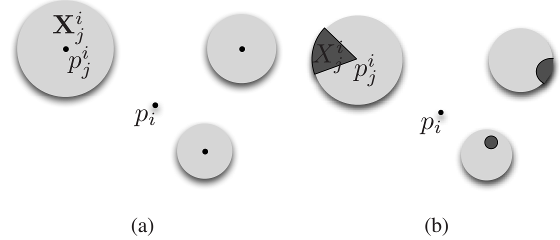


Fig. 2. Graphical representation of the information available to an agent i with (a) no promises and (b) varying promises from different agents. The dark regions correspond to the promise sets X_j^i given the promises and the light regions correspond to the guaranteed sets $\mathbf{X}_j^i$.

V. Team-triggered algorithmic solution

A drawback of the distributed self-triggered communication protocol presented in Section IV is that in general it is very conservative. More specifically, an agent requests information when it can no longer *guarantee* that the value of the objective function will decrease. The problem with this is that worst-case considerations about neighbors actions must be considered at all times, even though neighbors may not be acting in this way in general.

In this section we apply a modification of the team-triggered idea from [15] to address this issue. The backbone of the team-triggered strategy is the use of 'promises' among agents that provide a better quality of information than strictly position information as is used in the self-triggered algorithm. We discuss this first.

A. Promises

A *promise* that an agent j makes to agent i at a given time t_ℓ^i is a subset of the allowable control space $\mathcal{U}_j^i \subset \overline{B}(0, v_{\max})$. This set conveys the promise that agent j will only use controls $u_j(t) \in \mathcal{U}_j^i$ for $t \geq t_\ell^i$. From this promise, agent i is able to generate a promise set $X_j^i(t)$ such that $p_j(t) \in X_j^i(t)$ is guaranteed for all $t \geq t_\ell^i$ provided that the promise is kept.

Promises provide agents with a higher quality of information than simply position information. Given position information $p_j(t_\ell^i)$ and a promise $\mathcal{U}_j^i$ that agent j makes to agent i at time t_ℓ^i, agent i can compute a *promise set*

$$X_j^i(t) = \{x_j(t) \in S \mid \exists u_j : [t_\ell^i, t] \to \mathcal{U}_j^i \quad (11)$$
$$\text{such that } x_j(t) = p_j(t_\ell^i) + \int_{t_\ell^i}^{t} u_j(\tau) d\tau\}.$$

This conveys to agent i that $p_j(t) \in X_j^i(t)$ for $t \geq t_\ell^i$. The shape of these sets vary depending on how promises are made among the agents. Figure 2 shows an example of what these promise sets might look like for a given agent i.

For the remainder of the paper, we only consider *ball-radius promises* as defined below for simplicity. However, we note that all subsequent results hold for any promises $\mathcal{U}_j^i$.

Definition V.1 (Ball-radius promises) The ball-radius promise that agent j makes to agent i at time t_ℓ^i is a ball

$$\mathcal{U}_j^i = \overline{B}(u_j^i, \delta_j) \cap \overline{B}(0, v_{\max}), \quad (12)$$

where $u_j^i = u_j(t_\ell^i)$ is the control signal used by agent j at time t_ℓ^i and $\delta_j > 0$. This promise is a ball of radius δ_j in the control space centered at the control signal u_j^i. In general, δ_j does not need to be a constant as discussed in [15], but we consider constant $\delta_j = \delta > 0$ for all agents here. •

B. Agent data structure

Equipped with the promise information exchanged among agents, we are able to provide agents with better information about one another. Given the promise U_j^i that agent j send to agent i as defined in Definition V.1 (provided by $u_j^i = u_j(t_\ell^i)$ and δ), agent i can now construct the promise set

$$X_j^i(t) = \overline{B}(p_j^i + (t - t_\ell^i)u_j^i, (t - t_\ell^i)\delta) \cap \mathbf{X}_j^i(t). \quad (13)$$

This promise set has the property that $X_j^i(t) \subset \mathbf{X}_j^i(t)$ providing a higher quality of information to agent i.

With a slight abuse of notation, we redefine the data that agents maintain using information from promises,

$$\mathcal{D}^i(t) = (X_1^i(t), \ldots, X_N^i(t)) \subset S^N. \quad (14)$$

C. Team-triggered algorithm design

Given the new data structure, we are able to reuse the motion control law and self-triggered update policy presented in Section IV. The only difference now is that agents use promises (13) rather than guaranteed sets (6) which allows agents to operate with better information resulting in better control decisions and less conservative conditions on when new information is required.

The only issue left to resolve is to ensure that agents are operating with correct information at all times. In other words, we need a mechanism such that if an agent breaks a promise to another agent at any time, it must immediately send updated information to that agent.

According to Algorithm 2, agent i checks if condition (9) is feasible or $\mathrm{bnd}_i \leq \varepsilon$, and therefore it is advantageous to execute the Motion Control Law. However, this decision only makes sense assuming that agents keep their promises at all times. If a promise to agent i is not kept, Proposition IV.1 does not hold because it is not guaranteed that $\mathrm{g}V_i \subset V_i \subset \mathrm{dg}V_i$.

We address this by supplementing the self-triggered deployment algorithm with the Event-Triggered Update Policy presented in Algorithm 3.

Algorithm 3 is responsible for ensuring that agents are operate with accurate information at all times. If at any time an agent breaks a promise to a neighbor it must correct this by

Algorithm 3: Event-Triggered Update Policy

Agent $i \in \{1,\ldots,N\}$ performs:
1: **if** there exists j such that $p_i \notin X_i^j$ **then**
2: send current position information p_i to agent j
3: send current control signal u_i to agent j
4: **end if**

immediately sending new position information and control signal. Lastly, if an agent i receives unsolicited information, meaning a neighboring agent has broken a promise and sent it a new one, it must also request updated information from all its neighbors. This is important due to the dynamic nature of the communication topology as we discuss next.

1) The synthesized algorithm: Here, we present the team-triggered algorithm to achieve optimal deployment with outdated information. The algorithm is the result of combining the Motion Control Law, the Self-Triggered Update Policy, and the Event-Triggered Update Policy with a procedure to acquire updated information about other agents when this requirement is triggered by Algorithm 2 (Requesting new information). In the self-triggered algorithm proposed in [14], it is sufficient to only receive information from an agent's Voronoi neighbors when an updated is required. However, due to the new information structure provided by the promises, we need a modification to ensure that $\mathrm{g}V_i \subset V_i$ for all agents at all times. Let the communication radius $R_i^{\text{self}} = 2\max_{j\in\mathcal{N}_i}\|p_i - p_j\|$. When agent i decides new information is needed for the self-triggered algorithm, it requests information from all agents within R_i^{self} of it. A method for computing R_i^{self} is provided in [14]. Instead, we utilize the communication radius $R_i = R_i^{\text{self}} + \beta$ where $\beta > 0$ is an a priori chosen design parameter. To capture the fact that the topology is changing, we define for each agent i, a subset of agents $\mathcal{A}^i \subset \{1,\ldots,N\}$ whose information will be used rather than all the agents of the network. Each time an agent i requests updated information, $\mathcal{A}^i$ is set to to the list of all agents within R_i of p_i. We define $\pi_{\mathcal{A}^i}$ as the map that extracts the information about the agents contained in $\mathcal{A}^i$ from $\mathcal{D}^i$. The following result then specifies a minimum required communication rate to ensure information is shared often enough. Its proof is omitted due to space constraints.

Lemma V.2 (Minimum required communication) *If agents request updated information at least every $\frac{\beta}{6v_{max}}$ sec, then $\mathrm{g}V_i(\pi_{\mathcal{A}^i}) \subset V_i$ holds for all $i \in \{1,\ldots,N\}$ at all times.*

Proof: Let t_0 be the time at which agent i receives updated information and thus $\mathcal{A}^i = \{j \in \{1,\ldots,N\} \mid \|p_i(t_0) - p_j(t_0)\| \le R_i^{\text{self}}(t_0) + \beta\}$. We now need to show that $\mathrm{g}V_i(\pi_{\mathcal{A}^i}(\mathcal{D}^i(t)) \subset V_i(\mathcal{D}(t))$ for $t \in [t_0, t_0 + \frac{\beta}{6v_{\max}}]$. It was shown in [14] that if for all $\|p_i - p_j\| \le R_i^{\text{self}}$, we have $j \in \mathcal{A}^i$, then $\mathrm{g}V_i(\pi_{\mathcal{A}^i}(\mathcal{D}^i)) \subset V_i$. For $t \in [t_0, t_0 + \frac{\beta}{6v_{\max}}]$, we are able to bound

$$\begin{aligned} R_i^{\text{self}}(t) = 2\max_{j\in\mathcal{N}_i}\|p_i(t) - p_j(t)\| &\le R_i^{\text{self}}(t_0) + 4v_{\max}(t - t_0) \\ &\le R_i^{\text{self}}(t_0) + \frac{2}{3}\beta. \end{aligned}$$

Now, consider $k \notin \mathcal{A}^i$, we know $\|p_i(t_0) - p_k(t_0)\| > R_i^{\text{self}}(t_0) + \beta$, and thus

$$\begin{aligned} \|p_i(t) - p_k(t)\| &> R_i^{\text{self}}(t_0) + \beta - 2v_{\max}(t - t_0) \\ &\ge R_i^{\text{self}}(t_0) - \frac{2}{3}\beta \end{aligned}$$

for $t \in [t_0, t_0 + \frac{\beta}{6v_{\max}}]$. Combining these we have

$$\|p_i(t) - p_k(t)\| > R_i^{\text{self}}(t), \quad \forall k \notin \mathcal{A}^i.$$

■

By ensuring the condition of Lemma V.2, we are able to leverage the result in [14, Lemma 5.3] and ensure all required agents are accounted for.

The fully synthesized Team-Triggered Centroid Algorithm is formally presented in Algorithm 4.

Algorithm 4: Team-Triggered Centroid Algorithm

Agent $i \in \{1,\ldots,N\}$ performs:
1: set $D = \pi_{\mathcal{A}^i}(\mathcal{D}^i)$
2: compute $L = \mathrm{g}V_i(D)$ and $U = \mathrm{dg}V_i(D)$
3: compute $q = C_L$ and $r = \mathrm{bnd}(L, U)$

(Requesting new information)
1: **if** $r \ge \max\{\|q - p_i\|, \varepsilon\}$ OR unsolicited information is received OR $\frac{\beta}{2v_{\max}}$ seconds have elapsed since last update **then**
2: request updated information
3: set $\mathcal{A}^i = \{j \mid p_j \in \overline{B}(p_i, R_i)\}$
4: set $D = \pi_{\mathcal{A}^i}(\mathcal{D}^i)$
5: set $L = \mathrm{g}V(D)$ and $U = \mathrm{dg}V(D)$
6: set $q = C_L$ and $r = \mathrm{bnd}(L, U)$
7: **end if**

(Sending new information)
1: **if** there exists j such that $p_i \notin X_i^j$ **then**
2: send current position information p_i to agent j
3: send current control signal u_i to agent j
4: **end if**

(Motion control)
1: compute u_i^* as defined in (10)

Remark V.3 (Dynamic ball-radius promises) In order to minimize communication, it may be more favorable to consider ball-radius promises in Definition V.1 where the radius of the balls evolve alongside the network execution. For instance, an agent i may increase its promise radius δ_i each time it breaks a promise to a neighbor in hopes of breaking them less. Similarly, it may decrease its promise radius if a neighboring agent requests new information before the prior promise is broken. Another possibility is for agents to keep track of how often promises are broken over time and adjust their promise radii based on this information. •

VI. Analysis of team-triggered law

In this section we analyze the asymptotic convergence properties of the Team-Triggered Centroid Algorithm. To properly analyze the trajectories of this system, we must consider the evolution of the entire network's memory $\mathcal{D} = (\mathcal{D}^1,\ldots,\mathcal{D}^N) \subset S^{N^2}$.

We begin by noticing that the promise information $X_j^i(t)$ that an agent i has about an agent j at any given time can be described by 3 parameters (since we assume the ball-radius

δ is fixed by all agents): the position information $p^i_j \in S$ that was last communicated to agent i, the control signal $u^i_j = u_j \in \mathcal{U}$ used at that time, and the uncertainty radius $r^i_j \in \mathbb{R}_{\geq 0}$. With a slight abuse of notation, we say that

$$\mathcal{D} \in S_E^{N^2} = (S \times \mathcal{U} \times \mathbb{R}_{\geq 0})^{N^2}.$$

For convenience, we define the map $\mathrm{loc}(\mathcal{D}) = (p_1, \dots, p_N)$ that extracts the position information of the agents from $\mathcal{D}$.

The Team-Triggered Centroid Algorithm can then be written as a discrete-time map $f_{\mathrm{ttca}} : S_E^{N^2} \to S_E^{N^2}$ which corresponds to one timestep Δt of the composition of a "decision/update-information" map f_{info} and a "move-and-update-uncertainty" map f_{motion}, i.e., $f_{\mathrm{ttca}}(\mathcal{D}) = f_{\mathrm{motion}}(f_{\mathrm{info}}(\mathcal{D}))$ for $\mathcal{D} \in S_E^{N^2}$. Unfortunately, it is difficult to analyze these trajectories directly because the map f_{ttca} is discontinuous.

Our objective is to prove the following result characterizing the asymptotic convergence properties of the trajectories of the Team-Triggered Centroid Algorithm.

Proposition VI.1 *For $\varepsilon \in [0, \mathrm{diam}(S))$, the agents' positions evolving under the Team-Triggered Centroid Algorithm from any initial network configuration in S^N converges to the set of centroidal Voronoi configurations.*

Since the map f_{ttca} is discontinuous, we cannot directly apply the discrete-time LaSalle Invariance Principle. In order to prove Proposition VI.1, we construct a closed, discrete-time set-valued map T_{sync} whose trajectories include the ones of the deterministic f_{ttca}. We are then able to apply the LaSalle Invariance Principle for set-valued maps, e.g., [20].

Next, we define the two components of T_{sync} formally. The first component captures the motion of the agents and propagation of the promise sets, and the second component captures the possibility of communication among agents. For simplicity, we define T_{sync} and provide analysis of Proposition VI.1 for agents using the ball-radius promises with fixed $\delta > 0$ from Definition V.1.

Remark VI.2 (Convergence with arbitrary promises) We note here that the convergence result of Proposition VI.1 holds for any promises $\mathcal{U}^i_j$ of non-zero measure among agents, not just the ball-radius promise. In order for our analysis to hold for different promises, we just require a careful redefining of the set-valued map T_{sync}. However, various definitions of promises does affect the rate of convergence and amount of communication induced among the agents. •

Motion and uncertainty update. We define the set-valued motion and uncertainty update map as $\mathcal{M} : S_E^{N^2} \rightrightarrows S_E^{N^2}$ whose ith component is

$$\begin{aligned}\mathcal{M}_i(\mathcal{D}, \ell) = \big(&(p^i_1, u^i_1, \max\{r^i_1 + \mu^i \Delta t, \mathrm{diam}(S)\}), \dots, \\ &(p^i_i + u^*_i \Delta t, u^*_i, 0), \dots, \\ &(p^i_N, u^i_N, \max\{r^i_N + \mu^i \Delta t, \mathrm{diam}(S)\})\big),\end{aligned}$$

where u^*_i is computed by (10) with $\mathcal{A}^i = \{i\} \cup \mathrm{argmin}_{j \in \{1,\dots,N\} \setminus \{i\}} r^i_j$ and

$$\mu^i \in \begin{cases} \{2v_{\max}\} & \text{if } \exists j \in \mathcal{A}^i \text{ s.t. } p^j_j \notin \overline{B}(p^i_j, r^i_j + \delta \Delta t), \\ \{2v_{\max}, \delta\} & \text{otherwise.} \end{cases} \tag{15}$$

The definition of μ^i ensures that even if promises might have been broken, the uncertainty sets are being properly grown so that the information contained is still accurate. This is important in the result of Lemma VI.3 below. It is easy to see that the map $\mathcal{M}$ is closed (a set-valued map $T : X \rightrightarrows Y$ is closed if $x_k \to x$, $y_k \to y$ and $y_k \in T(x_k)$ imply that $y \in T(x)$).

Acquisition of up-to-date information. In each timestep, agents have the possibility of communicating and receiving updated information from their Voronoi neighbors. This is captured by the set-valued map $\mathcal{I} : S_E^{N^2} \rightrightarrows S_E^{N^2}$ that, to $\mathcal{D} \in S_E^{N^2}$, associates the Cartesian product $\mathcal{I}(\mathcal{D})$ whose ith component is either $\mathcal{D}^i$ (agent i does not get any updated information) or the vector

$$((p'_1, u'_1, r'_1), \dots, (p'_N, u'_N, r'_N))$$

where $(p'_j, u'_j, r'_N) = (p^j_j, u^*_j, 0)$ for $j \in \{i\} \cup \mathcal{N}_i$ and $(p'_j, u'_j, r'_j) = (p^i_j, u^i_j, r^i_j)$ otherwise (agent i gets updated information). Recall that $\mathcal{N}_i$ is the set of neighbors of agent i given the partition $\mathcal{V}(\mathrm{loc}(\mathcal{D}))$. It is not difficult to show that $\mathcal{I}$ is also closed.

We define the set-valued map $T_{\mathrm{sync}} : S_E^{N^2} \rightrightarrows S_E^{N^2}$ by $T_{\mathrm{sync}} = \mathcal{I} \circ \mathcal{M}$. Given that both $\mathcal{M}$ and $\mathcal{I}$ are closed, the map T_{sync} is closed. Moreover, if $\gamma = \{\mathcal{D}(t_\ell)\}_{\ell \in \mathbb{Z}_{\geq 0}}$ is an evolution of the Team-Triggered Centroid Algorithm, then $\gamma' = \{\mathcal{D}'(t_\ell)\}_{\ell \in \mathbb{Z}_{\geq 0}}$, with $\mathcal{D}'(t_\ell) = f_{\mathrm{info}}(\mathcal{D}(t_\ell))$, is a trajectory of

$$\mathcal{D}'(t_{\ell+1}) \in T_{\mathrm{sync}}(\mathcal{D}'(t_\ell)). \tag{16}$$

The following result establishes the monotonic evolution of the objective function $\mathcal{H}$ along the trajectories of T_{sync}.

Lemma VI.3 $\mathcal{H} : S_E^{N^2} \to \mathbb{R}$ *is monotonically nonincreasing along the trajectories of* T_{sync}.

Proof: Let $\mathcal{D} \in S_E^{N^2}$ and $\mathcal{D}' \in T_{\mathrm{sync}}(\mathcal{D})$. For convenience, let $P = \mathrm{loc}(\mathcal{D})$ and $P' = \mathrm{loc}(\mathcal{D}') = \mathrm{loc}(\mathcal{M}(\mathcal{D}))$. To establish $\mathcal{H}(P') \leq \mathcal{H}(P)$, we leverage the inequalities (4) and (5). First, let the partition $\mathcal{V}(P)$ be fixed. For each $i \in \{1, \dots, N\}$, if $\|p^i_i - C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\| \leq \mathrm{bnd}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))$, then $p'^i_i = p^i_i$ because agent i does not move according to the control law (10). If, instead, $\|p^i_i - C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\| > \mathrm{bnd}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))$, then, by the control law (10) and Proposition IV.1, we have that $\|p'^i_i - C_{V_i}\| \leq \|p^i_i - C_{V_i}\|$. In either case, it follows from (5) that $\mathcal{H}(P', \mathcal{V}(P)) \leq \mathcal{H}(P, \mathcal{V}(P))$. It is important to note that the above only holds true if agents are operating with accurate information about one another. This is ensured by the definition of (15) in the motion and uncertainty update map. Second, the optimality of the Voronoi partition stated in (4) guarantees that $\mathcal{H}(P', \mathcal{V}(P')) \leq \mathcal{H}(P', \mathcal{V}(P))$, and the result follows. ■

One can establish the next result using Lemma VI.3 and the fact that T_{sync} is closed and its trajectories are bounded and belong to the closed set $S_E^{N^2}$.

Lemma VI.4 *Let γ' be a trajectory of* (16). *Then, the ω-limit set $\emptyset \neq \Omega(\gamma') \subset S_E^{N^2}$ belongs to $\mathcal{H}^{-1}(c)$, for some $c \in \mathbb{R}$, and is weakly positively invariant for T_{sync}, i.e., for $\mathcal{D} \in \Omega(\gamma')$, $\exists\ \mathcal{D}' \in T_{\mathrm{sync}}(\mathcal{D})$ with $\mathcal{D}' \in \Omega(\gamma')$.*

Proof: Let γ' be a trajectory of (16). It is clear that $\Omega(\gamma') \neq \emptyset$ because γ' is bounded. Let $\mathcal{D}' \in \Omega(\gamma')$. Then there exists a subsequence $\{\mathcal{D}'(t_{\ell_m})\}_{m\in\mathbb{Z}_{\geq 0}}$ of γ' such that $\lim_{m\to+\infty} \mathcal{D}'(t_{\ell_m}) = \mathcal{D}'$. Consider $\{\mathcal{D}'(t_{\ell_{m+1}})\}_{m\in\mathbb{Z}_{\geq 0}}$. Since this sequence is bounded, it must have a convergent subsequence, i.e., there exists $\widehat{\mathcal{D}}'$ such that $\lim_{m\to+\infty} \mathcal{D}'(t_{\ell_{m+1}}) = \widehat{\mathcal{D}}'$. By definition, $\widehat{\mathcal{D}}' \in \Omega(\gamma')$. Since T_{sync} is closed, we have $\widehat{\mathcal{D}}' \in T_{\mathrm{sync}}(\mathcal{D}')$, which implies that $\Omega(\gamma')$ is weakly positively invariant.

Now consider the sequence $\{\mathcal{H}(P(t_\ell))\}_{\ell\in\mathbb{Z}_{\geq 0}}$, where $P(t_\ell) = \mathrm{loc}(\gamma'(t_\ell))$. Since $\mathcal{H}$ is nonincreasing and bounded from below there exists $c \in \mathbb{R}$ such that $\lim_{\ell\to\infty} \mathcal{H}(\gamma(t_\ell)) = c$. Now take any $z \in \Omega(\gamma')$, by definition of the limit set there exists a convergent subsequence of γ' that goes to z. By continuity of $\mathcal{H}$ we conclude that $\mathcal{H}(\mathrm{loc}(z)) = c$. ■

We are now ready to establish our main convergence result.

PROOF OF PROPOSITION VI.1. Let $\gamma = \{\mathcal{D}(t_\ell)\}_{\ell\in\mathbb{Z}_{\geq 0}}$ be an evolution of the Team-Triggered Centroid Algorithm and $\gamma' = \{\mathcal{D}'(t_\ell)\}_{\ell\in\mathbb{Z}_{\geq 0}}$ where $\mathcal{D}'(t_\ell) = f_{\mathrm{info}}(\mathcal{D}(t_\ell))$. Note that $\mathrm{loc}(\mathcal{D}(t_\ell)) = \mathrm{loc}(\mathcal{D}'(t_\ell))$. We now use a contradiction to show that the limit set of γ' is given by

$$\Omega(\gamma') \subseteq \{\mathcal{D} \in S_E^{N^2} \mid \text{for } i \in \{1,\dots,N\}, \quad \|p_i^i - C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\| \leq \mathrm{bnd}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\}. \tag{17}$$

Assume there exists $\mathcal{D} \in \Omega(\gamma)$ and $i \in \{1,\dots,N\}$ such that $\|p_i^i - C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\| > \mathrm{bnd}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))$. Then, Proposition IV.1 and the control law (10) guarantee that $\mathcal{H}$ will strictly decrease under T_{sync}, which is a contradiction with the fact that $\Omega(\gamma')$ is weakly positively invariant for T_{sync}.

Note that the inequality $\mathrm{bnd}_i < \max\{\|p_i^i - C_{\mathrm{g}V_i}\|, \varepsilon\}$ is satisfied at $\mathcal{D}'(t_\ell)$, for all $\ell \in \mathbb{Z}_{\geq 0}$ because this prescribes an update by agent i. By continuity, it also holds on $\Omega(\gamma')$,

$$\mathrm{bnd}(\pi_{\mathcal{A}^i}(\mathcal{D}^i)) \leq \max\{\|p_i^i - C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\mathcal{D}^i))\|, \varepsilon\}, \tag{18}$$

for all $i \in \{1,\dots,N\}$ and all $\mathcal{D} \in \Omega(\gamma')$. We are now interested in showing $\Omega(\gamma') \subseteq \{\mathcal{D} \in S_E^{N^2} \mid \text{for } i \in \{1,\dots,N\}, p_i^i = C_{V_i}\}$. Consider $\widetilde{\mathcal{D}} \in \Omega(\gamma')$. Since $\Omega(\gamma')$ is weakly positively invariant, there exists $\widetilde{\mathcal{D}}_1 \in \Omega(\gamma') \cap T_{\mathrm{sync}}(\widetilde{\mathcal{D}})$. Note that (17) implies that $\mathrm{loc}(\widetilde{\mathcal{D}}_1) = \mathrm{loc}(\widetilde{\mathcal{D}})$, i.e., no agents are moving. There are two reasons this might happen depending on whether or not agents have updated information in $\widetilde{\mathcal{D}}_1$. If agent i does not receive updated information, because of the minimum required communication forced by Lemma V.2, there exists $\widetilde{\mathcal{D}}_m \in T_{\mathrm{sync}}(\widetilde{\mathcal{D}}_{m-1}) \in \dots \in T_{\mathrm{sync}}(\widetilde{\mathcal{D}}_1)$ such that agent i receives updated information. Once agent i gets updated information, then $\mathrm{bnd}(\pi_{\mathcal{A}^i}(\widetilde{\mathcal{D}}_m^i)) = 0$, and consequently, from (17), $p_i^i = p_i'^i = C_{\mathrm{g}V_i}(\pi_{\mathcal{A}^i}(\widetilde{\mathcal{D}}_m^i) = C_{V_i}$, and the result follows.

VII. Simulations

In this section we compare the proposed team-triggered strategy with the self-triggered algorithm from [14] and the periodic communication strategy from [1]. We consider a network of $N = 8$ agents moving in a 4m × 4m square with $v_{\max} = 1$ m/s and a timestep of $\Delta t = 0.025$ s. The density ϕ is a sum of two Gaussian functions

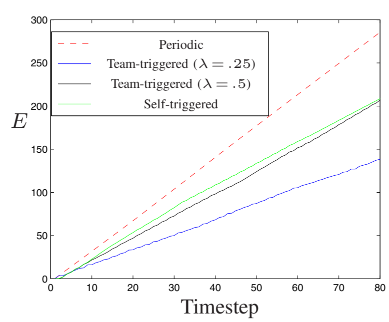
$$\phi(x) = e^{-\|x-q_1\|^2} + e^{-\|x-q_2\|^2},$$

with $q_1 = (2,3)$ and $q_2 = (3,1)$. We use the following model [21] for quantifying the power $\mathcal{P}_i$ used by agent $i \in \{1,\dots,8\}$ to communicate, in $dBmW$ power units,

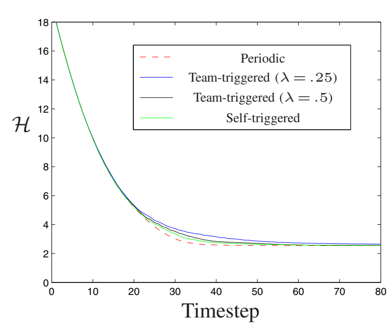
$$\mathcal{P}_i = 10\log_{10}\Big[\sum_{j\in\{1,\dots,N\},i\neq j} \alpha_2 10^{0.1P_{i\to j}+\alpha_1\|p_i-p_j\|}\Big], \tag{19}$$

where $\alpha_1, \alpha_2 > 0$ depend on the properties of the wireless medium and $P_{i\to j}$ is the power received by j of the signal transmitted by i in units of $dBmW$. In our simulations, these values are set to 1.

The ball-radius promises are generated using Definition V.1 with $\delta = 2\lambda v_{\max}$, where $\lambda \in [0,1]$ is a design parameter that captures the 'tightness' of promises. Setting $\lambda = 0$ corresponds to promise sets that are exact trajectories and $\lambda = 1$ corresponds recovers the self-triggered case because promise sets and guaranteed sets defined in 6 become equivalent.



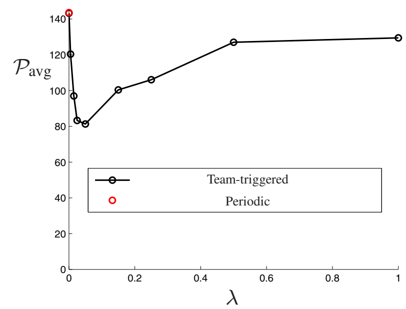
(a) Total energy expenditure



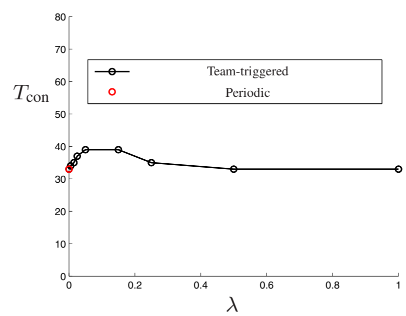
(b) Objective function

Fig. 3. Plots of (a) the total communication energy E in Joules and (b) the evolution of the objective function $\mathcal{H}$ of executions of the periodic, the self-, and team-triggered (for two different tightnesses of promises) strategies.

Figure 3 compare executions of the periodic, self-triggered, and team triggered strategies for two different tightnesses of promises ($\lambda = 0.25$ and $\lambda = 0.5$) starting from the same initial condition. Figure 3(a) shows the total communication energy over time and Figure 3(b) shows the evolution of the objective function $\mathcal{H}$. From these figures we can see that the tightness of promises indeed have an effect on the algorithm executions. For $\lambda = 0.25$, Figure 3 shows that the communication energy is cut in half compared to the periodic strategy without compromising the network performance.



(a)



(b)

Fig. 4. Plots of the average (a) communication power network consumption $\mathcal{P}_{\text{avg}}$ and (b) time to convergence T_{con} for the team-triggered deployment strategy with varying λ. Time to convergence is the time to reach 99% of the final convergence value of the objective function.

Figure 4 demonstrates more clearly the effects of varying the tightness of promises λ in the team-triggered strategy. Figure 4(a) shows the average power consumption in terms of communication energy by the entire network and Figure 4(b) shows the time it takes each execution to reach 99% of the final convergence value of $\mathcal{H}$. Interestingly, for small λ, we see substantially less amounts of required communication while the time to convergence only slightly increases.

VIII. Conclusions

We have proposed the Team-Triggered Centroid Algorithm to solve an optimal deployment problem for a group of mobile sensors. The strategy combines ideas from event- and self-triggered control that provides agents with sufficient autonomy to decide among themselves when communication is necessary. We have analyzed the correctness of the algorithm using tools from computational geometry and set-valued stability analysis. Our result provides the same convergence properties as an algorithm assuming perfect information at all times. Simulations demonstrate the potential benefits of such an algorithm compared to periodic or self-triggered communication strategies. For future work, we plan to rigorously analyze the effects that promises have on the network executions, and methods for optimally constructing these promises with respect to different performance metrics. We also intend to apply similar team-triggered strategies to other distributed coordination tasks.

References

[1] J. Cortés, S. Martínez, 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] A. Kwok and S. Martínez, "Deployment algorithms for a power-constrained mobile sensor network," *International Journal on Robust and Nonlinear Control*, vol. 20, no. 7, pp. 725–842, 2010.

[3] J. L. Ny and G. J. Pappas, "Adaptive deployment of mobile robotic networks," *IEEE Transactions on Automatic Control*, vol. 58, no. 3, pp. 654–666, 2013.

[4] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1291–1297, 2012.

[5] G. Shi and K. H. Johansson, "Multi-agent robust consensus-part II: application to event-triggered coordination," in *IEEE Conf. on Decision and Control*, (Orlando, FL), pp. 5738–5743, Dec. 2011.

[6] X. Meng and T. Chen, "Event based agreement protocols for multi-agent networks," *Automatica*, vol. 49, no. 7, pp. 2125–2132, 2013.

[7] M. Mazo Jr. and P. Tabuada, "On event-triggered and self-triggered control over sensor/actuator networks," in *IEEE Conf. on Decision and Control*, (Cancun, Mexico), pp. 435–440, 2008.

[8] Y. Fan, G. Feng, Y. Wang, and C. Song, "Distributed event-triggered control of multi-agent systems with combinational measurements," *Automatica*, vol. 49, no. 2, pp. 671–675, 2013.

[9] A. Eqtami, D. V. Dimarogonas, and K. J. Kyriakopoulos, "Event-triggered strategies for decentralized model predictive controllers," in *IFAC World Congress*, (Milano, Italy), Aug. 2011.

[10] E. Garcia and P. J. Antsaklis, "Model-based event-triggered control for systems with quantization and time-varying network delays," *IEEE Transactions on Automatic Control*, vol. 58, no. 2, pp. 422–434, 2013.

[11] W. P. M. H. Heemels and M. C. F. Donkers, "Model-based periodic event-triggered control for linear systems," *Automatica*, vol. 49, no. 3, pp. 698–711, 2013.

[12] X. Wang and M. D. Lemmon, "Event-triggering in distributed networked control systems," *IEEE Transactions on Automatic Control*, vol. 56, no. 3, pp. 586–601, 2011.

[13] G. S. Seybotha, D. V. Dimarogonas, and K. H. Johansson, "Event-based broadcasting for multi-agent average consensus," *Automatica*, vol. 49, no. 1, pp. 245–252, 2013.

[14] C. Nowzari and J. Cortés, "Self-triggered coordination of robotic networks for optimal deployment," *Automatica*, vol. 48, no. 6, pp. 1077–1087, 2012.

[15] C. Nowzari and J. Cortés, "Team-triggered coordination for real-time control of networked cyberphysical systems," *IEEE Transactions on Automatic Control*, 2013. Submitted.

[16] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Wiley Series in Probability and Statistics, Wiley, 2 ed., 2000.

[17] J. Sember and W. Evans, "Guaranteed Voronoi diagrams of uncertain sites," in *Canadian Conference on Computational Geometry*, (Montreal, Canada), 2008.

[18] M. Jooyandeh, A. Mohades, and M. Mirzakhah, "Uncertain Voronoi diagram," *Information Processing Letters*, vol. 109, no. 13, pp. 709–712, 2009.

[19] Q. Du, V. Faber, and M. Gunzburger, "Centroidal Voronoi tessellations: Applications and algorithms," *SIAM Review*, vol. 41, no. 4, pp. 637–676, 1999.

[20] F. Bullo, J. Cortés, and S. Martínez, *Distributed Control of Robotic Networks*. Applied Mathematics Series, Princeton University Press, 2009. Electronically available at http://coordinationbook.info.

[21] S. Firouzabadi, "Jointly optimal placement and power allocation in wireless networks," Master's thesis, University of Maryland at College Park, 2007.