A cooperative deployment strategy for optimal sampling in spatiotemporal estimation

Rishi Graham Jorge Cortés

***Abstract*— This paper considers a network composed of robotic agents and static nodes performing spatial estimation of a dynamic physical processes. The physical process is modeled as a spatiotemporal random field with finite spatial correlation range. We propose a distributed coordination algorithm to optimize data acquisition across time. The robotic agents take measurements of the processes and relay them to the static nodes. The static nodes collectively compute directions of maximum descent of the estimation uncertainty, and relay them back to the robotic agents. The technical approach combines tools from geostatistics, parallel computing, and systems and control. We illustrate the soundness of the algorithm in simulation.**

I. Introduction

Problem statement: This paper considers a network of static nodes and robotic sensors taking sequential measurements of a dynamic physical process. We model the underlying process as a spatiotemporal random field. Our objective is to determine trajectories for the robots which optimize data acquisition in order to best estimate the field. This problem has applications in environmental monitoring, oceanographic surveying, and atmospheric sampling.

Literature review: Kriging [1], [2] is a standard geostatistical technique for estimating spatiotemporal random fields. Given a set of point measurements, kriging produces a predictor of the field throughout the environment, along with a measure of the uncertainty associated with the predictor. Under certain conditions on the covariance structure, data taken far from the prediction site have very little impact on the kriging predictor [3]. When the spatiotemporal random field does not have a covariance structure with finite spatial correlation, an approximation may be generated via covariance tapering [4]. The optimal design literature [5], [6] deals with the problem of determining sets of locations where data should be taken in order to optimize the resulting kriging estimation. The work [7] examines the effect that addition and deletion of measurement locations has on the error kriging variance, and how this relates to optimal design.

The field of cooperative control for mobile sensor networks has received much recent attention. The work [8] introduces performance metrics for oceanographic surveys by autonomous underwater vehicles. The work [9] considers a network of robotic sensors with centralized control estimating a static field from measurements with both sensing and localization error. The work [10] considers choosing the optimal sampling trajectories from a parameterized set of paths. In [11], [12] the focus is on estimating deterministic fields when the measurements taken by individual robots are uncorrelated. The tracking of level curves in a noisy scalar field is discussed in [13].

Rishi Graham is with the Department of Applied Mathematics and Statistics, University of California, Santa Cruz, CA 95064, USA, rishig@ams.ucsc.edu

Jorge Cortés is with the Department of Mechanical and Aerospace Engineering, University of California, San Diego, CA 92093, USA, cortes@ucsd.edu

Statement of contributions: We consider a robotic network comprised of static nodes and robotic sensor agents. The combination of static nodes and mobile robots allows us to distribute the burden associated with sensing, communication, and computing. The environment is partitioned into regions, and each static node is responsible of maintaining an approximation of the spatial field on its region. The nodes are deployed so that their communication topology is connected, and any robotic agent can communicate to at least one node at any given time. The robots are responsible for taking measurements of the field and relaying them back to the nearest nodes along with position information.

The main contribution of this paper is the design of a distributed coordination algorithm to optimally sample dynamic physical processes modeled as spatiotemporal random fields. As a criterion for optimality, we consider the spatiotemporal average of the kriging variance. This function has the natural interpretation of an aggregate objective function that measures the uncertainty about the knowledge of the random field. Under the assumption of a finite correlation range in space, we provide an upper bound on the kriging variance, which in turn induces an upper bound on our objective function amenable to distributed optimization. The static nodes compute the gradient of the approximate average kriging variance and relay simple control vectors back to the robotic agents. This guarantees that the next measurements are taken at positions which decrease the approximate overall uncertainty of the estimation. We do not pay attention to how the estimation is actually implemented, but rather focus on how to minimize the uncertainty of the estimate so that data acquisition is optimized. For brevity, some proofs are omitted and will appear elsewhere.

Organization: Section II introduces basic notation and tools from constrained optimization and kriging estimation. Section III introduces the robotic network model and details the overall network objective. The following two sections present important ingredients in the ulterior algorithm design. Section IV specifies the regions of allowed motion for the robotic agents at each step, while Section V describes an upper bound of the spatiotemporal average of the kriging variance. Section VI presents the distributed coordination algorithm that the robotic network executes to optimize data acquisition, along with some illustrative simulations. Section VII contains our conclusions and ideas for future work.

II. Preliminary notions

In this section we introduce some preliminary concepts and notation. Let $\mathbb{R}$, $\mathbb{R}_{>0}$, and $\mathbb{R}_{\geq 0}$ denote the set of reals, positive reals and nonnegative reals, respectively. For $p \in \mathbb{R}^d$ and $r \in \mathbb{R}_{>0}$, we let $\overline{B}(p, r)$ denote the *closed ball* of radius r centered at p. Given two vectors $u = (u_1, \ldots, u_a)^T$, $a \in \mathbb{Z}_{>0}$, and $v = (v_1, \ldots, v_b)^T$, $b \in \mathbb{Z}_{>0}$, we denote by (u, v) its concatenation $(u, v) = (u_1, \ldots, u_a, v_1, \ldots, v_b)^T$. We denote by ∂S the boundary of a set S. The *ϵ-contraction* of a set S, with $\epsilon > 0$, is the set $S_\epsilon = \{q \in S \mid \mathrm{d}(q, \partial S) \geq \epsilon\}$. A *convex polytope* is the convex hull of a finite point set. For a bounded set $S \subset \mathbb{R}^d$, we let $\mathrm{CR}(S)$ denote the *circumradius* of S, that is, the radius of the smallest-radius d-sphere enclosing S. We denote by $\mathbb{F}(S)$ the collection of finite subsets of S.

We are concerned with operations on a compact and connected set $\mathcal{D}$ of Euclidean space $\mathbb{R}^d$, $d \in \mathbb{N}$. Since we deal with a process which varies over time, let $\mathcal{D}_e = \mathcal{D} \times \mathbb{R}$ denote the space of points over $\mathcal{D}$ and time. In general, we use $s \in \mathcal{D}$ to denote spatial position, $t \in \mathbb{R}$ to denote continuous time, $k \in \mathbb{Z}_{\geq 0}$ to denote discrete time increments, and $h = (s, t) \in \mathcal{D}_e$ to denote locations in space and time. When disambiguation is required, we use the superscript notation $f^{(k)}$ when referring to the function f at the kth timestep, and the subscript notation f_j when referring to the jth component of a vector valued function. To denote a range of timesteps, we use the notation $f^{(k_1:k_2)}$, $k_1 < k_2$, to indicate the function f at timesteps k_1 through k_2.

A *partition* of $\mathcal{D}$ is a collection of n polygons $\mathcal{W} = \{W_1, \ldots, W_n\}$ with disjoint interiors whose union is $\mathcal{D}$. The *Voronoi partition* $\mathcal{V}(\boldsymbol{s}) = (V_1(\boldsymbol{s}), \ldots, V_n(\boldsymbol{s}))$ of $\mathcal{D}$ generated by the points $\boldsymbol{s} = (s_1, \ldots, s_n)$ is defined by

$$V_i(\boldsymbol{s}) = \{q \in \mathcal{D} \mid \|q - s_i\| \leq \|q - s_j\|,\ \forall j \neq i\}.$$

Each $V_i(\boldsymbol{s})$ is called a *Voronoi cell*. Two points s_i and s_j are *Voronoi neighbors* if their Voronoi cells share a boundary.

A. Projected gradient descent

Next, we describe the constrained optimization technique known as projected gradient descent [14]. This technique, combined with generalized Armijo step sizes, may be used to iteratively find minima of an objective function.

Let $m \in \mathbb{N}$, and let Ω denote a nonempty, closed, and convex subset of $\mathbb{R}^m$. Let $F : \mathbb{R}^m \to \mathbb{R}_{\geq 0}$, and assume that the objective is to minimize F. Further assume that the gradient ∇F is globally Lipschitz on Ω. Let $\mathrm{proj}_\Omega : \mathbb{R}^m \to \Omega$ denote the orthogonal projection onto the set Ω, i.e.,

$$\mathrm{proj}_\Omega(x) = \operatorname*{argmin}_{y \in \Omega} \|x - y\|.$$

Consider a sequence $\{x_k\} \in \Omega$, $k \in \mathbb{N}$, which satisfies

$$x_{k+1} = \mathrm{proj}_\Omega\left(x_k - a_k \nabla F(x_k)\right),\ x_1 \in \Omega, \tag{1}$$

where the step size, a_k, is chosen according to the line search algorithm described in Table I, evaluated at $x = x_k$.

Here the grid size τ determines the granularity of the line search. The tolerance θ may be adjusted for a more (larger θ) or less (smaller θ) strict gradient descent. Note that as long as $\theta > 0$, the line search algorithm must terminate in finite time, while a larger value of θ will decrease the number of iterations. The condition in step 8, known as the Armijo condition, ensures that the decrease in F is commensurate with the magnitude of its gradient. A sequence $\{x_k\}_{k=1}^{\infty}$ satisfying these requirements converges in the limit as $k \to \infty$ to stationary points of F, see [14, Proposition 1].

Name:	Line Search Algorithm
Goal:	Determine step size for projected gradient descent algorithm (1)
Input:	$x \in \Omega$
Assumes:	(i) grid size $\tau \in (0, 1)$ (ii) tolerance $\theta \in (0, 1)$ (iii) maximum step size $\alpha_{\max} \in \mathbb{R}_{>0}$
Output:	$\alpha \in \mathbb{R}_{\geq 0}$

1: $\alpha = \alpha_{\max}$
2: **repeat**
3: $\quad x_{\text{new}} = \mathrm{proj}_\Omega\left(x - \alpha \nabla F(x)\right)$
4: $\quad \nu = \frac{\theta}{\alpha}\|x - x_{\text{new}}\|^2 + F(x_{\text{new}}) - F(x)$
5: $\quad$ **if** $\nu > 0$ **then**
6: $\quad\quad \alpha = \alpha\tau$
7: $\quad$ **end if**
8: **until** $\nu \leq 0$

TABLE I
Line Search Algorithm.

B. Estimation via Kriging interpolation

This section reviews the geostatistical kriging procedure for the estimation of spatial processes, see e.g., [2], [15]. A time-varying random process δ on $\mathcal{D}_e$ is *second-order stationary* if it has constant mean, and its covariance is of the form $\mathrm{Cov}(\delta(h_1), \delta(h_2)) = C(h_1, h_2)$, where $C : \mathcal{D}_e \times \mathcal{D}_e \to \mathbb{R}_{\geq 0}$ is a positive definite covariance function which only depends on the difference $h_1 - h_2$. The covariance matrix of the vector of points $\boldsymbol{h} = (h_1, \ldots, h_l) \in \mathcal{D}_e^l$, $l \in \mathbb{N}$, is $\boldsymbol{\Sigma} = \Sigma(\boldsymbol{h}) = [C(h_i, h_j)]_{i,j=1}^l \in \mathbb{R}^{l \times l}$. When it is clear from the context, we use bold face to denote explicit dependence on $\boldsymbol{h}$. We define $c : \mathcal{D}_e \times \mathcal{D}_e^l \to \mathbb{R}^l$ to be the vector of covariances between a single point, $h \in \mathcal{D}_e$ and the vector $\boldsymbol{h}$, i.e., $\boldsymbol{c} = c(h, \boldsymbol{h}) = (C(h, h_1), \ldots, C(h, h_l))^T$.

We assume that the random process Z is of the form

$$Z(h) = \mu(h) + \delta(h), \quad h \in \mathcal{D}_e, \tag{2}$$

where μ is the mean function and δ is a zero-mean second-order stationary random process with a known covariance function, C. We assume that C has a *finite spatial range* $r \in \mathbb{R}_{>0}$, such that

$$C\left((s_1, t_1), (s_2, t_2)\right) = 0, \text{ if } \|s_2 - s_1\| > r.$$

We also assume that measurement data $\boldsymbol{y} = y(\boldsymbol{h}) = (Y(h_1), \ldots, Y(h_l))^T$ are corrupted with errors according to

$$Y(h_i) = Z(h_i) + \epsilon_i, \qquad \epsilon_i \overset{\text{iid}}{\sim} N\left(0, \sigma_\epsilon^2\right),\ \sigma_\epsilon \in \mathbb{R}. \tag{3}$$

The constant variance in measurement error models identical sensors. The covariance between $Y(h_i)$ and $Y(h_j)$ is written

$$\mathrm{Cov}[Y(h_i), Y(h_j)] = \begin{cases} C(h_i, h_j) + \sigma_\epsilon^2, & \text{if } i = j, \\ C(h_i, h_j), & \text{otherwise.} \end{cases}$$

Note that the covariance matrix of $\boldsymbol{h}$ with respect to the noisy process Y may be written $\boldsymbol{\Sigma}_\epsilon = \Sigma_\epsilon(\boldsymbol{h}) = \Sigma(\boldsymbol{h}) + \sigma_\epsilon^2 I_n$, where I_n denotes the $n \times n$ identity matrix.

1) The simple kriging predictor: Assuming that the mean function μ is known, the *simple kriging predictor* at $h \in \mathcal{D}_e$ from the data measured at locations $\boldsymbol{h}$ is the predictor that minimizes the error variance,

$$\sigma^2(Z(h); \boldsymbol{h}) = \mathrm{Var}\left[Z(h) - \mathrm{pred}(Z(h); \boldsymbol{h})\right], \qquad (4)$$

among all unbiased predictors of the form $\mathrm{pred}(Z(h); \boldsymbol{h}) = \sum_{i=1}^{l} \alpha_i Y(h_i) + k$. Let $\boldsymbol{\mu} = (\mu(h_1), \ldots, \mu(h_l))^T$. The explicit expression of the simple kriging predictor of Z at $h \in \mathcal{D}_e$ is

$$\hat{z}_{\mathrm{SK}}(h; \boldsymbol{h}) = \mu(h) + \boldsymbol{c}^T \boldsymbol{\Sigma}_\epsilon^{-1}(\boldsymbol{y} - \boldsymbol{\mu}), \qquad (5)$$

with error variance

$$\sigma^2_{\mathrm{SK}}(Z(h); \boldsymbol{h}) = \sigma_Z^2(h) - \boldsymbol{c}^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{c}. \qquad (6)$$

Here $\sigma_Z^2(h) = C(h, h)$ denotes the variance of $Z(h)$, while $\boldsymbol{c}\boldsymbol{\Sigma}_\epsilon^{-1}\boldsymbol{c}$ represents the variance of the simple kriging predictor $\hat{z}_{\mathrm{SK}}(h; \boldsymbol{h})$. Under the assumption that Z is stationary, σ_Z^2 is constant, and we drop the dependence on h.

2) The universal kriging predictor: Relaxing the assumption that the mean function μ is known, consider a linear expansion upon a set of $p \in \mathbb{N}$ known basis functions $f_1, \ldots, f_p : \mathcal{D}_e \to \mathbb{R}$. We write $\mu(h) = f(h)^T \beta$, where $f(h) = (f_1(h), \ldots, f_p(h))^T$ and $\beta = (\beta_1, \ldots, \beta_p)^T \in \mathbb{R}^p$. The *universal kriging predictor* of Z at $h \in \mathcal{D}_e$ is the predictor that minimizes the error variance (4) among all unbiased predictors of the form $\mathrm{pred}(Z(h); \boldsymbol{h}) = \sum_{i=1}^{l} \alpha_i Y(h_i)$. The explicit expression at $h \in \mathcal{D}_e$ is

$$\hat{z}_{\mathrm{UK}}(h; \boldsymbol{h}) = \left(\boldsymbol{c} + \boldsymbol{F}\left(\boldsymbol{F}^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{F}\right)^{-1}\left(f - \boldsymbol{F}^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{c}\right)\right)^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{y}, \qquad (7)$$

where $\boldsymbol{F}$ denotes the matrix whose ith row is $f(h_i)^T$. The error variance of $\hat{z}_{\mathrm{UK}}(h, \boldsymbol{h})$ is

$$\sigma^2_{\mathrm{UK}}(Z(h); \boldsymbol{h}) = \sigma_Z^2 - \boldsymbol{c}^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{c} + \left(f - \boldsymbol{F}^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{c}\right)^T \left(\boldsymbol{F}^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{F}\right)^{-1} \left(f - \boldsymbol{F}^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{c}\right). \qquad (8)$$

Note that if $p > n$, then the matrix $\boldsymbol{F}^T \boldsymbol{\Sigma}_\epsilon^{-1} \boldsymbol{F}$ is not full rank, and the universal kriging predictor is not well-defined.

In this paper, unless explicitly stated otherwise, our results make no distinction between simple and universal kriging. To simplify notation, we drop the subscript and use $\hat{z}$ to denote both kriging estimators with associated error variance σ^2. Note that both $\hat{z}$ and σ^2 only depend on the positions of the measurements in $\mathcal{D}_e$, not the actual values. Also note that σ^2 is invariant under permutations of $h_1, \ldots, h_l$. This guarantees that the value of σ^2 remains the same no matter how the elements of the set are ordered. Thus without loss of precision, we will evaluate σ^2 at a set, instead of at a tuple.

III. Problem statement

In Section III-A we introduce the robotic network model and in Section III-B we detail the overall network objective.

A. Robotic sensor network model

Consider a group $\{S_1, \ldots, S_m\}$ of $m \in \mathbb{N}$ static nodes deployed in a convex polytope $\mathcal{D} \subset \mathbb{R}^d$. Let $Q = (q_1, \ldots, q_m) \in \mathcal{D}^m$ denote the positions of the static nodes. Assume that each node has a limited communication radius, $R \in \mathbb{R}_{>0}$, and that they are positioned so that each one can communicate with its Voronoi neighbors.

In addition to the static nodes, consider a group $\{R_1, \ldots, R_n\}$ of n robotic sensor agents. The position of robot $i \in \{1, \ldots, n\}$ at time $t \in \mathbb{R}$ is denoted by $p_i(t) \in \mathcal{D}$. We assume that robots take measurements of the spatial field at discrete instants of time in $\mathbb{Z}_{\geq 0}$. Between measurement instants, each robot moves according to the discrete dynamics

$$p_i(k+1) = p_i(k) + u_i(k),$$

where $\|u_i\| \leq u_{\max}$ for some $u_{\max} \in \mathbb{R}_{>0}$. The communication radius of the robotic agents is also R. Each node will need to be able to communicate with any robot which may be within covariance range of the points in its Voronoi region at the following timestep. To that end, we assume that

$$R \geq \max_{i \in \{1, \ldots, m\}} \{\mathrm{CR}(V_i(Q))\} + r + u_{\max}. \qquad (9)$$

The robots are also assumed to have some limited capability of sensing each other, so that a robot knows the positions of any other robots within a distance of $2u_{\max}$. At discrete timesteps, each robot communicates the measurement and location to static nodes within communication range, along with the locations of any other sensed robots. The nodes are then responsible for computing an estimate of the field, and relaying control-specific information back to those robotics within communication range. Our implementation does not require direct communication between robotic agents.

B. The average kriging variance as objective function

Given the communication, sensing, and motion capabilities of the network described in Section III-A, our objective is to design a coordination algorithm that optimizes the estimation of the spatial field Z. Here, we introduce the network objective function that we seek to optimize.

Assume that the experiment has been run for $k_{\max} \in \mathbb{Z}_{\geq 0}$ timesteps and a sequence of measurements taken at time intervals $\{1, \ldots, k_{\max}\}$, at space-time locations $\boldsymbol{h} \in (\mathcal{D}_e^n)^{k_{\max}}$, are available. With these measurements a kriging estimate at $h \in \mathcal{D}_e$ has associated error variance

$$\sigma^2(Z(h); \boldsymbol{h}) = \mathrm{Var}[\bar{z}(h; \boldsymbol{h})],$$

where $\bar{z}(h; \boldsymbol{h}) = Z(h) - \hat{z}(h; \boldsymbol{h})$ is a shorthand notation. There are a number of ways to define optimality of a kriging estimator. Consider a kriging estimation $\hat{z}(h; \boldsymbol{h})$ made on $\mathcal{D}$ over the interval $T = [1, k_{\max}]$. The average error variance of the estimator $\hat{z}(h; \boldsymbol{h})$ over $s \in \mathcal{D}$ and $t \in T$ is given by

$$\mathcal{A} = \int_T \int_{\mathcal{D}} \sigma^2(Z(s,t); \boldsymbol{h})\, ds\, dt.$$

The question becomes how to choose the best locations to take measurements in order to minimize $\mathcal{A}$. Since the measurements are taken sequentially, and each set is restricted to a region nearby the previous measurements, one cannot

simply optimize over $(\mathcal{D}_e^n)^{k_{\max}}$. Additionally, if $k_{\max}$ may not be known at the start of the experiment.

Consider, instead, a greedy approach in which we use measurements taken at timesteps $\{1,\dots,k\}$ to choose the positions for the $(k+1)$-st set of measurements. Let $\boldsymbol{h}^{(1:k)} \in (\mathcal{D}_e^n)^k$ be the vector of measurement location and time pairs for all measurements up to time k, ordered in blocks corresponding to timesteps, i.e.,

$$\boldsymbol{h}^{(1:k)} = ((p_1(1),1),\dots,(p_n(1),1),\dots, \\ (p_1(k),k),\dots,(p_n(k),k))^T.$$

Let $P = (p_1,\dots,p_n) \in \mathcal{D}_e^n$ denote the current positions of the robotic agents, and with a slight abuse of notation we will use $(P, k+1)$ to denote the vector of space-time locations at spatial positions P and time $k+1$. Let $\mathcal{A}^{(k)} : \mathcal{D}^n \to \mathbb{R}$ map the next set of locations to the average error variance of the kriging estimate built with measurements up to and including time $k+1$, i.e.,

$$\mathcal{A}^{(k)}(P) = \int_T \int_{\mathcal{D}} \sigma^2\left(Z(s,t);(\boldsymbol{h}^{(1:k)},(P,k+1))\right) ds\, dt.$$

The objective is to choose the set of measurement locations P at time $k+1$ so as to maximally decrease the value of $\mathcal{A}^{(k)}$. Unfortunately, the gradient of $\mathcal{A}^{(k)}$ cannot be computed in a distributed way over the communication graph of the static nodes because of the requirement of inverting matrices which depend on all measurement locations. Instead, we construct an upper bound to $\mathcal{A}^{(k)}$, whose gradient is distributed, and design an algorithm to optimize it.

IV. Voronoi contraction for collision avoidance

We begin by specifying the region of allowed movement for the robotic agents. As noted in Section III-A, each robot's motion is restricted by a maximum velocity at each timestep, as well as the requirement that it must stay within the physical region $\mathcal{D}$. Here we provide an additional requirement which ensures that robots do not collide. Beyond the benefit of damage avoidance, this restriction ensures that even under the assumption of zero sensor error, the kriging error function is well-defined over the space of possible configurations.

Let $\omega \in \mathbb{R}_{\geq 0}$ be a desired buffer width. We assume that ω is small compared to the size of $\mathcal{D}$. One might use the diameter of the physical robotic agents as a basis for this width, adding space for robustness. To ensure that the distance between two robots is never smaller than ω, we introduce a contraction of the Voronoi diagram.

Consider the locations $P = (p_1,\dots,p_n)$ of the n robotic agents at the kth timestep. Let $(V_i(P))_{\omega/2}$ denote the $\frac{\omega}{2}$-contraction of $V_i(P)$. Let $\Omega_i^{(k)} \subset \mathcal{D}$ such that

$$\Omega_i^{(k)} = (V_i(P))_{\omega/2} \cap \overline{B}(p_i, u_{\max}).$$

For each $j \neq i \in \{1,\dots,n\}$, we have $\mathrm{d}(\Omega_i^{(k)},\Omega_j^{(k)}) \geq \omega$. Between timesteps k and $k+1$, we will restrict R_i to the region $\Omega_i^{(k)}$. Figure 1 shows an example in $\mathbb{R}^2$ of this set.

Let $\Omega^{(k)} = \prod_{i=1}^n \Omega_i^{(k)} \subset (\mathbb{R}^d)^n$ denote the region of allowed movement of all robotic agents at timestep $k \in \mathbb{Z}_{\geq 0}$. Note that $\Omega^{(k)}$ is closed, bounded, and convex. Next we present some results regarding $\Omega^{(k)}$. The following lemma describes some useful results concerning $\Omega^{(k)}$.

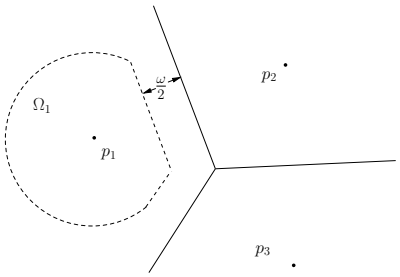

Fig. 1. Comparison of the regions $\{\Omega_i^{(k)}\}_{i=1}^n$ (dashed) with the Voronoi partition (solid).

Lemma IV.1 *For the network described in Section III-A, let the initial condition satisfy $\|p_i(0) - p_j(0)\| \geq \omega$ for all $i \neq j \in \{1,\dots,n\}$, and assume that $P^{(k+1)} \in \Omega^{(k)}$, for all $k \in \mathbb{Z}_{\geq 0}$ throughout the evolution. Then the following holds:*

(i) *$\|p_i(k) - p_j(k)\| \geq \omega$ for all $i \neq j \in \{1,\dots,n\}$ and all $k \in \mathbb{Z}_{\geq 0}$,*

(ii) *for all $i \in \{1,\dots,n\}$ and $j \in \{1,\dots,m\}$, we have that $\mathrm{d}(p_i(k+1), V_j(Q)) \leq r \implies \|p_i(k) - q_j\| \leq R$.*

Proof: Fact (i) follows directly from the definition of ω as the minimum separation distance between any two sets $\Omega_i^{(k)}$ and $\Omega_j^{(k)}$. To show fact (ii), we write

$$\begin{aligned} &\mathrm{d}(p_i(k+1), V_j(Q)) \leq r \\ &\quad\implies \mathrm{d}(p_i(k), V_j(Q)) \leq r + u_{\max} \\ &\quad\implies \|p_i(k) - q_j\| \leq \mathrm{CR}(V_j(Q)) + r + u_{\max} \\ &\quad\implies \|p_i(k) - q_j\| \leq R. \end{aligned}$$

In the second implication, we have used the fact that $\mathrm{CR}(V_j(Q))$ is the maximum distance from q_j to any point in $V_j(Q)$. The final implication is a direct consequence of assumption (9). ∎

V. Local approximation of the average kriging variance

In this section, we compute an upper bound on the average kriging variance. The idea is to construct an approximation of the error variance at each location, using only nearby measurements. We begin by providing a useful result that isolates the effect of a subset of measurements on the kriging variance. Before presenting it, we introduce the following notation: given $\boldsymbol{h}_1 \in \mathcal{D}_e^l$ and $\boldsymbol{h}_2 \in \mathcal{D}_e^m$, with $i_{\mathbb{R}}(\boldsymbol{h}_1) \cap i_{\mathbb{R}}(\boldsymbol{h}_2) = \emptyset$, let

$$\hat{y}(\boldsymbol{h}_2;\boldsymbol{h}_1) = [\hat{y}(h_{21};\boldsymbol{h}_1),\dots,\hat{y}(h_{2m};\boldsymbol{h}_1)]^T,$$
$$\bar{y}(\boldsymbol{h}_2;\boldsymbol{h}_1) = y(\boldsymbol{h}_2) - \hat{y}(\boldsymbol{h}_2;\boldsymbol{h}_1).$$

We are now ready to present an upper bound on the kriging variance at any point.

Lemma V.1 (Upper bound on kriging variance) *Let $\boldsymbol{h} = (\boldsymbol{h}_1,\boldsymbol{h}_2)$ denote a full set of distinct measurement locations, with $\boldsymbol{h}_1 = (h_1,\dots,h_l) \in \mathcal{D}_e^l$ and $\boldsymbol{h}_2 = (h_{l+1},\dots,h_n) \in \mathcal{D}_e^m$, with $l+m=n$. Then,*

$$\sigma^2(Z(h);\boldsymbol{h}) = \sigma^2(Z(h);\boldsymbol{h}_1) - \operatorname{Cov}[\bar{z}(h;\boldsymbol{h}_1),\bar{y}(\boldsymbol{h}_2;\boldsymbol{h}_1)]\cdot \\ \cdot \operatorname{Var}[\bar{y}(\boldsymbol{h}_2;\boldsymbol{h}_1)]^{-1}\operatorname{Cov}[\bar{y}(\boldsymbol{h}_2;\boldsymbol{h}_1),\bar{z}(h;\boldsymbol{h}_1)] \\ \leq \sigma^2(Z(h);\boldsymbol{h}_1),$$

with equality if $\operatorname{Cov}[Z(h),y(\boldsymbol{h}_2)]$ *and* $\operatorname{Cov}[y(\boldsymbol{h}_1),y(\boldsymbol{h}_2)]$.

This result can be proven using [16, Proposition 8.2.4] to conveniently decompose the inverse covariance matrix. Note that $\sigma^2(Z(h);\boldsymbol{h}_1)$ corresponds to the error variance of a predictor computed with the information at locations $\boldsymbol{h}_1$.

The following result provides a method to calculate an upper bound on the average error variance in a distributed way over the communication graph of the static nodes.

Proposition V.2 (Spatial approximation for distributed implementation) *Define* $CS^{(\leq k+1)}:\mathbb{Z}_{\geq 0}\times\mathcal{D}^n\to\mathbb{F}(\mathcal{D}_e)$ *by*

$$CS^{(\leq k+1)}(j,P) = \\ \left\{(s,t)\in i_{\mathbb{F}}\left(\boldsymbol{h}^{(1:k)},(P,k+1)\right) \mid \mathrm{d}(s,V_j(Q))\leq r\right\},$$

i.e., the subset of measurement locations up to time $k+1$ *which are correlated in space to the Voronoi cell of the static node* j. *Let* $\tilde{\mathcal{A}}_j^{(k)}:\mathcal{D}^n\to\mathbb{R}$ *be defined by*

$$\tilde{\mathcal{A}}_j^{(k)}(P) = \int_{\mathcal{T}}\int_{V_j(Q)}\sigma^2\left(Z(s,t);CS^{(\leq k+1)}(j,P)\right)\,ds\,dt.$$

Then the sum, $\tilde{\mathcal{A}}^{(k)}=\sum_{j=1}^m\tilde{\mathcal{A}}_j^{(k)}$ *satisfies*

$$\mathcal{A}^{(k)}(P)\leq\tilde{\mathcal{A}}^{(k)}(P).$$

In addition, if, for all $j\in\{1,\dots,m\}$, *the points in* $CS^{(\leq k+1)}(j,P)$ *are not correlated to other measurement locations outside it, then equality holds.*

Proof: Using the Voronoi partition, the overall average error variance can be decomposed as

$$\mathcal{A}^{(k)}(P)=\sum_{j=1}^m\int_{\mathcal{T}}\int_{V_j(Q)}\sigma^2\Big(Z(s,t);\left(\boldsymbol{h}^{(1:k)},(P,k+1)\right)\Big)\,ds\,dt.$$

Since $\mathrm{CS}^{(\leq k+1)}(j,P)\subset i_{\mathbb{F}}\left(\boldsymbol{h}^{(1:k)},(P,k+1)\right)$, we may use Lemma V.1 to show the upper bound,

$$\sum_{j=1}^m\int_{\mathcal{T}}\int_{V_j(Q)}\sigma^2\Big(Z(s,t);\left(\boldsymbol{h}^{(1:k)},(P,k+1)\right)\Big)\,ds\,dt\leq \\ \sum_{j=1}^m\int_{\mathcal{T}}\int_{V_j(Q)}\sigma^2\Big(Z(s,t);\mathrm{CS}^{(\leq k+1)}(j,P)\Big)\,ds\,dt.$$

The condition of equality also follows directly from the statement of Lemma V.1. ■

Remark V.3 ($\tilde{\mathcal{A}}_j^{(k)}$ may be calculated with local information only) Note that the location p_i contributes only to $\tilde{\mathcal{A}}^{(k)}(P)$ in those Voronoi regions $V_j(Q)$ for which $\mathrm{d}(p_i,V_j(Q))\leq r$. Thus the requirement (9) ensures that S_j can calculate $\tilde{\mathcal{A}}_j^{(k)}$. As with σ^2, without loss of precision we will evaluate $\tilde{\mathcal{A}}_j^{(k)}$ at a set, rather than a tuple. •

Remark V.4 (Universal kriging with too few measurements) It should be noted here that in the universal kriging case the function $\tilde{\mathcal{A}}_j^{(k)}(P)$ is only well-defined if the number of measurement locations available to each node is greater than or equal to the number of basis functions. In this paper we assume that this holds at all times. •

Our next step is to characterize the smoothness properties of the function $\tilde{\mathcal{A}}^{(k)}$. Let us start by introducing some useful notation. For each $i\in\{1,\dots,n\}$, let

$$\mathrm{CS}_{-i}^{(\leq k+1)}(j,P)=\mathrm{CS}^{(\leq k+1)}(j,P)\setminus\{(p_i,k+1)\}.$$

Define the maps $C_i:\{1,\dots,m\}\times\mathcal{D}^n\times\mathcal{D}\times T\to\mathbb{R}$ and $V_i:\{1,\dots,m\}\times\mathcal{D}\to\mathbb{R}$ by

$$C_i(j,P,s,t)=\operatorname{Cov}[\bar{y}((p_i,k+1),\mathrm{CS}_{-i}^{(\leq k+1)}(j,P)), \\ \bar{z}((s,t),\mathrm{CS}_{-i}^{(\leq k+1)}(j,P))], \\ V_i(j,P)=\operatorname{Var}[\bar{y}((p_i,k+1),\mathrm{CS}_{-i}^{(\leq k+1)}(j,P))],$$

where $\bar{y}(h;\boldsymbol{h})=Y(h)-\hat{y}(h;\boldsymbol{h})$ is a convenient shorthand notation. For $s\in\mathcal{D}$ and $t\in T$, let $\nabla_iC_i(j,P,s,t)$ and $\nabla_iV_i(j,P)$ denote the partial derivative of C_i and V_i with respect to p_i. We are now ready to state our result on the gradient of $\tilde{\mathcal{A}}^{(k)}$.

Proposition V.5 *Assume that the covariance of* Z *is* C^1 *with respect to the spatial position of its arguments. In the universal kriging scenario, further assume that the mean basis functions* $f_1,\dots,f_p$ *are* C^1 *with respect to the spatial position of their arguments. Then* $\tilde{\mathcal{A}}^{(k)}$ *is* C^1 *on* $\Omega^{(k)}$ *and the* i*th component of its gradient is of the form*

$$\nabla_i\tilde{\mathcal{A}}^{(k)}(P)=\sum_{j=1}^m\nabla_i\tilde{\mathcal{A}}_j^{(k)}(P),$$

where

$$\nabla_i\tilde{\mathcal{A}}_j^{(k)}(P)=\frac{\int_{\mathcal{T}}\int_{V_j(Q)}C_i(j,P,s,t)^2\,ds\,dt\,\nabla_iV_i(j,P)}{V_i(j,P)^2}- \\ -\frac{\int_{\mathcal{T}}\int_{V_j(Q)}C_i(j,P,s,t)\nabla_iC_i(j,P,s,t)\,ds\,dt}{V_i(j,P)}.$$

Note that for each $i\in\{1,\dots,n\}$, $\nabla_i\tilde{\mathcal{A}}_j(P)$ may be computed by node j, and thus $\nabla_i\tilde{\mathcal{A}}^{(k)}(P)$ may be computed in a manner distributed across the network of nodes.

Existence of the gradient is only part of the picture. In order to satisfy the convergence criteria for the projected gradient algorithm, the gradient must be globally Lipschitz. This is what the next result states.

Proposition V.6 *Under the assumptions of Proposition V.5, make the following additional assumptions,*

- $\nabla_i\operatorname{Cov}[Z(p_i,k+1),Z(s_2,t_2)]$ *is globally Lipschitz on* $\Omega_i^{(k)}$ *for each* $i\in\{1,\dots,n\}$*;*
- *in the universal kriging case, further assume that the partial derivatives* $\frac{\partial}{\partial s}f_j$ *are globally Lipschitz on* $\Omega_i^{(k)}$.

Then the gradient, $\nabla\tilde{\mathcal{A}}^{(k)}$ *is globally Lipschitz on* $\Omega^{(k)}$, *i.e., there exists* $L\in\mathbb{R}_{\geq 0}$ *such that*

$$\|\nabla\tilde{\mathcal{A}}^{(k)}(P_1)-\nabla\tilde{\mathcal{A}}^{(k)}(P_2)\|\leq L\|P_1-P_2\|,\ \forall P_1,P_2\in\Omega^{(k)}.$$

VI. Optimizing information retrieval via gradient descent

In this section, we design a coordination algorithm to follow the gradient of $\tilde{\mathcal{A}}^{(k)}$. Under the network model of Section III-A, we consider a system in which each static node is responsible for calculating control vectors for the robotic agents within the region of influence. We use the formulation of the approximate average error presented in Proposition V.2, and follow a projected gradient descent building on Section II. The current timestep, k, is held fixed through the section and, therefore, to reduce notation, we leave off the superindex which indicates timestep where unnecessary. We use the notation $P = (p_1, \ldots, p_n)$ for the current positions of the robots.

A. Distributed optimization of the approximate average kriging variance

Ideally, at the kth timestep, we would like the robots to move to the minimum of the map $P \mapsto \tilde{\mathcal{A}}(P)$ over the region Ω. Finding such a minimum over the whole region is a difficult problem. Instead, we use a distributed version of the projected gradient descent algorithm, which is at least guaranteed to converge to a stationary point. Since each node is working with a reduced set of robot positions, we define the following notation for use in the algorithm. Let $P'_j : \mathbb{R}\times\mathcal{D}^n \to \mathbb{F}(\mathcal{D})$ map a stepsize and the current configuration to the set of next locations as calculated by S_j, i.e.,

$$P'_j(\alpha, P) = \Big\{ \operatorname{proj}_{\Omega_i}\left(p_i + \alpha\nabla_i\tilde{\mathcal{A}}(P)\right), \quad \text{foreach } i \text{ s.t. } \mathrm{d}\left(p_i, V_j(Q)\right) \leq r + u_{\max} + \omega \Big\}.$$

Let $\mathrm{d}_j : \mathbb{R}\times\mathcal{D}^n \to \mathbb{R}_{\geq 0}$ denote the total distance traveled by robots entering $V_j(Q)$, i.e.,

$$\mathrm{d}_j(\alpha, P) = \sum_{\substack{i\in\{1,\ldots,n\} \text{ such that} \\ \operatorname{proj}_{\Omega_i}\left(p_i+\alpha\nabla_i\tilde{\mathcal{A}}(P)\right)\in V_j(Q)}} \|\operatorname{proj}_{\Omega_i}\left(p_i + \alpha\nabla_i\tilde{\mathcal{A}}(P)\right) - p_i\|^2.$$

Globally, let $P' : \mathbb{R}\times\mathcal{D}^n \to \mathcal{D}^n$ be defined as $P'(\alpha, P) = \operatorname{proj}_{\Omega}(P + \alpha\nabla\tilde{\mathcal{A}}(P))$.

Table II describes a distributed version of the Line Search Algorithm, assuming a starting position of $P \in \Omega$. The line search starts with a scaling factor $\alpha_{\max}$ which scales the smallest nonzero partial to $u_{\max}$, ensuring that all robots with nonzero partial derivatives have the opportunity to move the maximum distance. In other words, we write

$$\alpha_{\max} = \frac{u_{\max}}{\min_{\substack{i\in\{1,\ldots,n\}\\ \delta(i)\neq 0}}\{\delta(i)\}}, \text{ where} \tag{10a}$$

$$\delta(i) = \left\|\nabla_i\tilde{\mathcal{A}}(P)\right\|. \tag{10b}$$

Lemma VI.1 *The* Distributed Line Search Algorithm *is equivalent to the* Line Search Algorithm *with $x = P$, and $F = \tilde{\mathcal{A}}$.*

Proof: By identifying terms, we can see that the Armijo conditions in the two algorithms match. Under the assumption that the network of nodes is connected, the summations in step 5 may be calculated via consensus. Thus it suffices to show the following equalities

$$\|P'(\alpha, P) - P\|^2 = \sum_{j=1}^{m} \mathrm{d}_j(\alpha, P)^2, \tag{11a}$$

$$= \tilde{\mathcal{A}}\left(P'(\alpha, P)\right) \sum_{j=1}^{m} \tilde{\mathcal{A}}_j\left(P'_j(\alpha, P)\right), \tag{11b}$$

$$\tilde{\mathcal{A}}(P) = \sum_{j=1}^{m} \tilde{\mathcal{A}}_j(P). \tag{11c}$$

We may decompose the squared norm in the following way

$$\|P'(\alpha, P) - P\|^2 = \sum_{i=1}^{n} \|\operatorname{proj}_{\Omega_i}\left(p_i + \alpha\nabla_i\tilde{\mathcal{A}}(P)\right) - p_i\|^2 = \sum_{j=1}^{m} \mathrm{d}_j(\alpha, P)^2.$$

Here the first equality stems from the definition of Ω as the product of Ω_i's. The second equality is a result of the fact that $\mathcal{V}(Q)$ is a partition of $\mathcal{D}$, and d_j depends only on those elements of $P'(\alpha, P)$ which lie in $V_j(Q)$. This proves Equation (11a). Equation (11b) follows from Remark V.3, while Equation (11c) is a result of the definition of $\tilde{\mathcal{A}}$. ■

We are now ready to present our technique for a greedy optimization algorithm. At timestep k, the nodes follow a

Name:	Distributed Line Search Algorithm
Goal:	Compute step size for gradient descent of $\tilde{\mathcal{A}}$
Input:	Configuration, $P = (p_1, \ldots, p_n) \in \mathcal{D}^n$
Assumes:	(i) Connected network of static nodes (ii) S_j knows p_i, $\nabla_i\tilde{\mathcal{A}}$ and Ω_i for each robot within communication range (iii) Step size τ and tolerance $\theta \in (0, 1)$ known a priori by all static nodes
Output:	Step size, $\alpha \in \mathbb{R}_{\geq 0}$

Initialization

1: Via a consensus algorithm, $S_1, \ldots, S_m$ calculate $\alpha_{\max}$ as defined in Equations (10).

For $j \in \{1, \ldots, m\}$, node S_j executes concurrently

1: $\alpha = \alpha_{\max}$
2: **repeat**
3: calculates $\tilde{\mathcal{A}}_j\left(P'_j(\alpha, P)\right) - \tilde{\mathcal{A}}_j(P)$
4: calculates $\mathrm{d}_j(\alpha, P)^2$
5: execute consensus algorithm to get the following:

$$\tilde{\mathcal{A}}(P'(\alpha, P)) - \tilde{\mathcal{A}}(P) = \sum_{j=1}^{m} \tilde{\mathcal{A}}_j\left(P'_j(\alpha, P)\right) - \tilde{\mathcal{A}}_j(P)$$

$$\left\|P - P'(\alpha, P)\right\|^2 = \sum_{j=1}^{m} \mathrm{d}_j(\alpha, P)^2$$

6: $\nu = \frac{\theta}{\alpha}\|P - P'(\alpha, P)\| + \tilde{\mathcal{A}}(P'(\alpha, P)) - \tilde{\mathcal{A}}(P)$
7: **if** $\nu > 0$ **then**
8: $\alpha = \alpha\tau$
9: **end if**
10: **until** $\nu \leq 0$

TABLE II
Distributed Line Search Algorithm.

gradient descent algorithm to define a sequence of configurations, $\{P_\gamma^\dagger\}$, $\gamma \in \mathbb{N}$, such that

$$P_1^\dagger = P^{(k)}$$
$$P_{\gamma+1}^\dagger = \text{proj}_\Omega\left(P_\gamma^\dagger - \alpha\nabla\tilde{\mathcal{A}}(P_\gamma^\dagger)\right),\ \alpha \in \mathbb{R}_{\geq 0},$$

where α is chosen via the DISTRIBUTED LINE SEARCH ALGORITHM. When $|\tilde{\mathcal{A}}(P_{\gamma+1}^\dagger) - \tilde{\mathcal{A}}(P_\gamma^\dagger)| = 0$, the algorithm terminates, and the nodes set $P^{(k+1)} = P_{\gamma+1}^\dagger$. By the end of this calculation, each node knows the identity of robotic agents that belong to its Voronoi cell at timestep $k+1$. Node S_j transmits $p_i(k+1)$ to R_i. Each robot then moves to $p_i(k+1)$ between timesteps. The overall gradient descent algorithm is summarized in Table III, where we have returned to the superscript notation to avoid confusion.

Proposition VI.2 *Under the assumptions of Proposition V.6, if the network follows the* DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM*, then the following holds:*

- *the algorithm is distributed over the network described in Section III-A,*
- *the robots will not collide,*
- *at each timestep after the first, measurements will be taken at configurations which are stationary points of the map $P \mapsto \tilde{\mathcal{A}}^{(k)}(P)$ over $\Omega^{(k)}$.*

Proof: The first statement follows by construction: each robotic agent and each static node uses only information acquired by itself or communicated by its neighbors, as described in the model of Section III-A. The second statement is a direct result of the construction of Ω. Between the kth and $k+1$st timesteps, R_i moves within Ω_i, with $\Omega_i \cap \Omega_j = \emptyset$, $\forall i \neq j \in \{1, \ldots, n\}$. Therefore the robots will not collide. The third statement is a result of [14, Proposition 1]. By Proposition V.6, $\nabla\tilde{\mathcal{A}}^{(k)}$ is globally Lipschitz on Ω. By Lemma VI.1, the step size calculation matches that in the LINE SEARCH ALGORITHM. Thus at each timestep, the nodes generate a sequence of configurations, $P_\gamma^\dagger$, which converges in the limit as $\gamma \to \infty$ to stationary points of $\tilde{\mathcal{A}}$ on Ω. ■

Remark VI.3 (Robustness to failure) One benefit to this algorithm over trajectories determined a priori is that it is robust to agent failures. If an agent stops sending position information to the nodes, it will cease to receive new control vectors, remaining in place. Meanwhile the network will naturally fill in the gaps. •

B. Simulation results

We performed simulations with the following parameters: $m = 5$ static nodes, $n = 10$ robotic agents, and the domain $\mathcal{D} = \{(0,.1),\ (2.5,.1),\ (3.45,1.6),\ (3.5,1.7),\ (3.45,1.8),\ (2.7,2.2),\ (1,2.4),\ (0.2,1.3)\}$. We used the separable covariance function defined by $\text{Cov}[Z(s_1,t_1), Z(s_2,t_2)] = C_{\text{tap}}(\|s_1 - s_2\|, 0.4)C_{\text{tap}}(|t_1 - t_2|, 9.5)$, where

$$C_{\text{tap}}(\delta, r) = \begin{cases} e^{-\frac{\delta}{r}}\left(1 - \frac{3\delta}{2r} + \frac{\delta^3}{2r^3}\right) \text{ if } \delta \leq r, \\ 0 \text{ otherwise.} \end{cases}$$

This is a tapered exponential function belonging to the class of covariance functions suggested in [4].

We compared the performance of our algorithm against two naive strategies. The first strategy was a static configuration in which the robots remained motionless in an incenter Voronoi configuration, i.e., a configuration such that each robot is located at the incenter of its own Voronoi cell. The second strategy was a lawnmower-type approach, in which we divided the environment vertically among the robots, and had them march back and forth along horizontal trajectories, avoiding the boundary of the region. Finally, we ran the DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM from the same starting position as the lawnmower-type approach. Each experiment ran for $k_{\max} = 20$ steps. Agent R_2 stopped transmitting measurements at time $k = 3$, while R_7 stopped at $k = 5$. Figure 2 shows the trajectory traveled by the gradient descent algorithm. Note that the two agents which

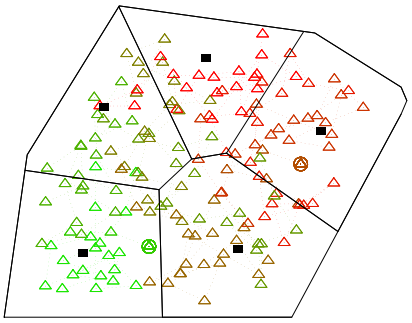

Fig. 2. Trajectory traced by the projected gradient descent algorithm. The squares represent the (static) positions of the nodes, Q, with the region partitioned according to $\mathcal{V}(Q)$. The triangles represent the locations at which measurements were taken, the circles represent measurements lost (not incorporated into the calculations).

stopped sending measurements ceased to move. The other agents avoided colliding with them, but filled in around them due to the gradient. Let $\mathcal{A}_S^{(k)}, \mathcal{A}_L^{(k)}, \mathcal{A}_G^{(k)} \in \mathbb{R}_{\geq 0}$ denote the average error as calculated at the kth step of the static, lawnmower, and gradient descent algorithms respectively. Figure 3 shows a plot of these errors as k in creases from 1 to $k_{\max}$. It can be seen that the gradient descent algorithm

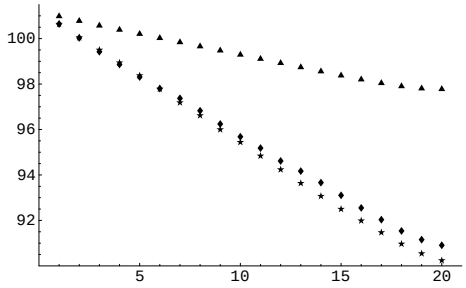

Fig. 3. Average errors, $\mathcal{A}_S^{(k)}$ (triangle), $\mathcal{A}_L^{(k)}$ (diamond), and $\mathcal{A}_G^{(k)}$ (star) as k increases.

has smaller error than either the static configuration or the lawnmower-type approach.

VII. CONCLUSIONS AND FUTURE WORK

We have considered a network composed of robotic sensors and static nodes performing spatial estimation tasks. We

Name: DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM
Goal: Find a local minimum of $\tilde{\mathcal{A}}^{(k)}$ within $\Omega^{(k)}$.

Assumes: (i) Connected network of static computing nodes and mobile robotic sensing agents (ii) Static nodes deployed over $\mathcal{D}$ such that $R \geq \max_{i\in\{1,\ldots,m\}} \{\mathrm{CR}(V_i(Q))\} + r + u_{\max}$ (iii) Step size τ and tolerance value $\theta \in (0,1)$ known a priori by all nodes (iv) Some termination marker known to all nodes and robots which may be sent to mark the end of a gradient descent loop.

Uses: (i) Each node uses the temporary vectors P_{cur}, respectively P_{next} to hold the configuration at the current, respectively next step of the gradient projection algorithm. For ease of exposition, we use global notation although S_j only calculates and uses the parts of these vectors which correspond to agents currently within communication range.

At time $k \in \mathbb{Z}_{\geq 0}$, node S_j executes:
1: sets $R_{\text{cov}}(j) = \{R_i \mid \mathrm{d}(p_i(k), V_j(Q)) \leq r\}$
2: collects measurements and locations from all robots in $R_{\text{cov}}(j)$
3: sets $P_{\text{next}} = P^{(k)}$
4: **repeat**
5: sets $P_{\text{cur}} = P_{\text{next}}(j)$
6: calculates $-\nabla\tilde{\mathcal{A}}_j^{(k)}(P_{\text{cur}})$
7: transmits vector $\nabla_i\tilde{\mathcal{A}}_j^{(k)}(P_{\text{cur}})$ to all robots in $R_{\text{cov}}(j)$
8: collects sum $\nabla_i\tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})$ from all robots in $R_{\text{cov}}(j)$
9: executes the DISTRIBUTED LINE SEARCH ALGORITHM at P_{cur} to calculate α
10: sets $P_{\text{next}} = P_{\text{cur}} + \alpha\nabla\tilde{\mathcal{A}}^{(k)}(P_{\text{cur}})$
11: calculates $|\tilde{\mathcal{A}}(P_{\text{next}}) - \tilde{\mathcal{A}}(P_{\text{cur}})|$ via consensus
12: **until** $|\tilde{\mathcal{A}}(P_{\text{next}}) - \tilde{\mathcal{A}}(P_{\text{cur}})| = 0$
13: sets $P^{(k+1)} = P_{\text{next}}$
14: sends a termination marker to all robots currently in $V_j(Q)$
15: conveys $p_i(k+1)$ to robots that currently belong to $V_j(Q)$

At time $k \in \mathbb{Z}_{\geq 0}$, robot R_i executes:
1: sets $S_{\text{cov}}(i) = \{S_j \mid \mathrm{d}(p_i(k), V_j(Q)) \leq r\}$
2: takes measurement at $p_i(k)$
3: sends measurement and position to all nodes in $S_{\text{cov}}(i)$
4: **repeat**
5: receives vectors $\nabla_i\tilde{\mathcal{A}}_j^{(k)}(P^{(k)})$ from all nodes in $S_{\text{cov}}(i)$
6: calculates sum $\nabla_i\tilde{\mathcal{A}}^{(k)}(P^{(k)})$
7: sends $\nabla_i\tilde{\mathcal{A}}^{(k)}(P^{(k)})$ to all nodes in $S_{\text{cov}}(i)$
8: **until** receives termination marker from any node
9: receives next location $p_i(k+1)$
10: moves to $p_i(k+1)$.

TABLE III
DISTRIBUTED PROJECTED GRADIENT DESCENT ALGORITHM.

have focused on the problem of optimizing data acquisition in order to better estimate a spatiotemporal random field. We have used the average error variance of the kriging estimator as a metric for the design of optimal measurement trajectories of the robots. In our approach, mobile robots take measurements of the environment and static nodes are responsible for collecting the measurements and computing locally optimal configurations for estimation. The design of the overall coordination algorithm combines Voronoi partitions, distributed projected gradient descent, and kriging interpolation technique to design. We have compared in simulations the performance of our approach against a static network configuration and a lawnmower-based approach.

Future work will focus on the investigation of theoretical guarantees on the performance and robustness to failure of the proposed coordination algorithm, the development of statistically-sound techniques for the case when, in universal kriging, any particular robot only has a small number of measurements available to it, and the quantification of the communication requirements of the proposed approach.

REFERENCES

[1] M. L. Stein, *Interpolation of Spatial Data. Some Theory for Kriging*. Springer Series in Statistics, New York: Springer Verlag, 1999.
[2] N. A. C. Cressie, *Statistics for Spatial Data*. New York: Wiley, 1993. revised edition.
[3] M. L. Stein, "The screening effect in kriging," *The Annals of Statistics*, vol. 30, pp. 298–323, Feb. 2002.
[4] R. Furrer, M. G. Genton, and D. Nychka, "Covariance tapering for interpolation of large spatial datasets," *Journal of Computational and Graphical Statistics*, vol. 15, no. 3, pp. 502–523, 2006.
[5] F. Pukelsheim, *Optimal Design of Experiments*, vol. 50 of *Classics in Applied Mathematics*. Philadelphia, PA: SIAM, 2006.
[6] E. P. Liski, N. K. Mandal, K. R. Shah, and B. K. Sinha, *Topics in Optimal Design*, vol. 163 of *Lecture Notes in Statistics*. New York: Springer Verlag, 2002.
[7] H. Gao, J. Wang, and P. Zhao, "The updated kriging variance and optimal sample design," *Mathematical Geology*, vol. 28, no. 3, pp. 295–313, 1996.
[8] J. S. Willcox, J. G. Bellingham, Y. Zhang, and A. B. Baggeroer, "Performance metrics for oceanographic surveys with autonomous underwater vehicles," *IEEE Journal of Oceanic Engineering*, vol. 26, no. 4, pp. 711–725, 2001.
[9] D. O. Popa, K. Sreenath, and F. L. Lewis, "Robotic deployment for environmental sampling applications," in *International Conference on Control and Automation*, (Budapest, Hungary), pp. 197–202, June 2005.
[10] N. E. Leonard, D. Paley, F. Lekien, R. Sepulchre, D. M. Fratantoni, and R. Davis, "Collective motion, sensor networks, and ocean sampling," *Proceedings of the IEEE*, vol. 45, pp. 48–74, Jan. 2007. Special Issue on Networked Control Systems.
[11] K. M. Lynch, I. B. Schwartz, P. Yang, and R. A. Freeman, "Decentralized environmental modeling by mobile sensor networks," *IEEE Transactions on Robotics*, 2006. Submitted.
[12] S. Martínez, "Distributed representation of spatial fields through an adaptive interpolation scheme," in *American Control Conference*, (New York City, NY), pp. 2750–2755, 2007.
[13] F. Zhang, E. Fiorelli, and N. E. Leonard, "Exploring scalar fields using multiple sensor platforms: tracking level curves," in *IEEE Conf. on Decision and Control*, (New Orleans, LA), pp. 3579–3584, Dec. 2007.
[14] D. P. Bertsekas, "On the Goldstein-Levitin-Polyak gradient projection method," *IEEE Transactions on Automatic Control*, vol. 21, no. 2, pp. 174–184, 1976.
[15] P. Abrahamsen, "A review of Gaussian random fields and correlation functions," Technical Report 917, Norwegian Computing Center, Oslo, Norway, 1997. Electronically available at http://publications.nr.no/917_Rapport.pdf.
[16] D. S. Bernstein, *Matrix Mathematics*. Princeton, NJ: Princeton University Press, 2005.