Persistent Robotic Tasks: Monitoring and Sweeping in Changing Environments

Stephen L. Smith, *Member, IEEE*, Mac Schwager, *Member, IEEE*, and Daniela Rus, *Fellow, IEEE*

***Abstract*—In this paper, we present controllers that enable mobile robots to persistently monitor or sweep a changing environment. The environment is modeled as a field that is defined over a finite set of locations. The field grows linearly at locations that are not within the range of a robot and decreases linearly at locations that are within range of a robot. We assume that the robots travel on given closed paths. The speed of each robot along its path is controlled to prevent the field from growing unbounded at any location. We consider the space of speed controllers that are parametrized by a finite set of basis functions. For a single robot, we develop a linear program that computes a speed controller in this space to keep the field bounded, if such a controller exists. Another linear program is derived to compute the speed controller that minimizes the maximum field value over the environment. We extend our linear program formulation to develop a multirobot controller that keeps the field bounded. We characterize, both theoretically and in simulation, the robustness of the controllers to modeling errors and to stochasticity in the environment.**

***Index Terms*—Motion control, optimization, path planning for multiple mobile robot systems, surveillance systems.**

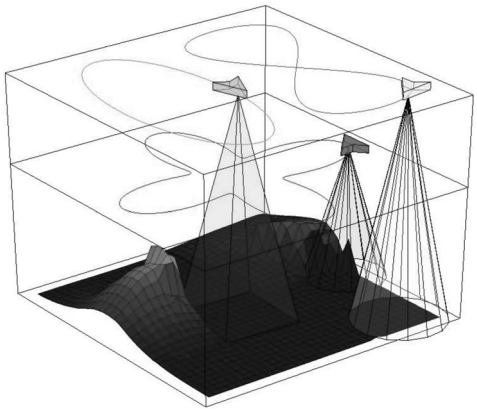

Fig. 1. Persistent monitoring task using three robots with heterogeneous, limited range sensors. The surface shows the accumulation function, indicating the quantity of material to be removed in a cleaning application, or the uncertainty at each point in a sensing application. The accumulation function grows when a robot's footprint is not over it and decreases when the footprint is over it. Each robot controls its speed along its prescribed path to keep the surface as low as possible everywhere.

I. Introduction

In THIS paper, we treat the problem of controlling robots to perpetually act in a changing environment, e.g., to clean an environment where material is constantly collecting or to monitor an environment where uncertainty is continually growing. Each robot has only a small footprint over which to act (e.g., to sweep or to sense). The difficulty is in controlling the robots to move so that their footprints visit all points in the environment regularly, spending more time in those locations where the environment changes quickly, without neglecting the locations, where it changes more slowly. This scenario is distinct from most other sweeping and monitoring scenarios in the literature because the task cannot be "completed." That is to say, the robots must continually move to satisfy the objective. We consider the situation in which robots are constrained to move on fixed paths, along which we must control their speed. We consider both the single-robot and multirobot cases. In Fig. 1, we show three robots monitoring an environment using controllers that are designed with our method.

We model the changing environment with a scalar valued function that is defined over a finite set of points, which we call the *accumulation function*. The function behaves analogously to dust accumulating over a floor. When a robot's footprint is not over a point in the environment, the accumulation function grows at that point at a constant rate, as if the density of dust were increasing at that point. When a robot's footprint is over the point, the accumulation function decreases at a constant rate, as if the dust were being vacuumed by the robot, thereby decreasing the density of dust. The rates of growth and decrease can be different at different points in the environment. This is a general model of a nonnegative scalar quantity accumulating at a set of points, where the growth and decrease of the quantity is linear in time.

This model is applicable to situations in which the task of the robot is to remove material that is collecting in the environment, e.g., cleaning up oil in the ocean around a leaking well [2], vacuuming dirt from the floor of a building [3], or tending to produce in an agricultural field [4]. It is also applicable to monitoring scenarios in which the state of the environment changes at each point, and we want to maintain up-to-date knowledge of this changing state. Some examples include monitoring

Manuscript received February 2, 2011; revised August 12, 2011; accepted October 28, 2011. Date of publication December 12, 2011; date of current version April 9, 2012. This paper was recommended for publication by Associate Editor D. Song and Editor G. Oriolo upon evaluation of the reviewers' comments. This work was supported in part by the Office of Naval Research Multidisciplinary University Initiative under Award N00014-09-1-1051. A preliminary version of this paper was presented at the IEEE International Conference on Robotics and Automation, Shanghai, China, May 2011.

S. L. Smith is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: stephen.smith@uwaterloo.ca).

M. Schwager is with the Department of Mechanical Engineering, Boston University, Boston, MA 02215 USA (e-mail: schwager@seas.upenn.edu).

D. Rus is with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: rus@csail.mit.edu).

Digital Object Identifier 10.1109/TRO.2011.2174493

temperature, salinity, or chlorophyll in the ocean [5], maintaining estimates of coral reef health [6], or monitoring traffic congestion over a city [7]. These applications are all alike in that they are defined by the accumulation of a nonnegative scalar quantity (oil, dirt, produce, or uncertainty). While these quantities may accumulate in a nonlinear fashion, our model gives a linear-in-time approximation to their accumulation. We assume that estimates of the parameters of the linear model are known either from the physics of the environment, from a human expert, or from an initial survey of the environment. Furthermore, we show analytically and in simulation that our controllers are robust to significant model errors. These applications also share the property that they can never be completed, because the accumulation function is always growing. If the robot were to stop moving, the oil would collect beyond acceptable levels, or the knowledge of the ocean temperature would become unacceptably outdated. For this reason, we call these applications the *persistent tasks*.

Ideally, we would optimize the full trajectory of each robot to optimally perform for a persistent task. However, obtaining performance guarantees for this problem is very difficult, as even the path planning component is NP-hard [8]. For this reason, we consider that the robots' paths are preplanned, and we focus on controlling the speed of each robot along its path. The concept of decoupling path planning from speed control is a well-established technique for dealing with complex trajectory planning problems [9], [10].

Although we focus on only controlling the robots' speeds in this study, methods for computation of efficient paths have been developed in our recent work [11]. Furthermore, a particular path may be required in certain applications. An example is ocean sampling, where paths are commonly prespecified by oceanographers [5]. In the case of autonomous aircraft, they may be constrained to fly along a particular path to stay away from commercial air traffic or to avoid being detected by an adversary. Even in the case of vacuuming the floor of a building, the robots may be required to stay along a prescribed path not to interfere with foot traffic.

A. Contributions

Our approach to the problem is to represent the space of all possible speed controllers with a finite set of basis functions, where each possible speed controller is a linear combination of those basis functions. A rich class of controllers can be represented this way. Using this representation as our foundation, the main contributions of this paper are the following.

1) We formally introduce the idea of persistent tasks for robots in changing environments and propose a tractable model for designing robot controllers.
2) Considering the space of speed controllers parametrized by a finite set of basis functions, we formulate a linear program (LP) whose solution is a speed controller, which guarantees that the accumulation function eventually falls below, and remains below, a known bound everywhere. If the LP is infeasible, then there is no controller in the space that will keep the accumulation function bounded.
3) We formulate an LP whose solution is the optimal (within the set spanned by our basis functions) speed controller—which eventually minimizes the maximum of the accumulation function over all locations.
4) We generalize to the multirobot case. We find an LP whose solution is a set of controllers, which guarantee that the accumulation function falls below, and stays below, a known bound. Although the LP is solved in a centralized fashion, the resulting controllers are decentralized in that they do not require communication between robots.

We do not find the optimal controller for the multirobot case, however, as it appears that this controller cannot be found as the solution to an LP. We demonstrate the performance of the controllers in numerical simulations and show both theoretically and in simulation that they are robust to stochastic and deterministic errors in the environment model and to unmodeled robot vehicle dynamics.

It is desirable to cast our speed control problem as an LP since LPs can be solved efficiently with off-the-shelf solvers [12]. This is enabled by our basis function representation. The use of basis functions is a common and powerful method for function approximation [13] and is frequently used in areas, such as compressive sampling [14], adaptive control [15], and machine learning [16]. Our LP formulations also incorporate both maximum and minimum limits on the robot's speed, which can be different at different points on the path. This is important because we may want a robot not to exceed a certain speed around a sharp turn, e.g., while it can go much faster on a long straightaway. In this paper, we expand on our preliminary work [1] by providing rigorous proofs for all results, by considering the multirobot problem, and by studying the robustness of the controllers.

Additionally, in [17], we have implemented these speed controllers for groups of ground robots and for groups of aerial robots, although we do not discuss the results here.

B. Related Work

Our work is related to the large body of existing research on environmental monitoring, sensor sweep coverage, lawn mowing and milling, and patrolling. In the environmental monitoring literature (also called objective analysis in meteorological research [18] and Kriging in geological research [19]), authors often use a probabilistic model of the environment and estimate the state of that model using a Kalman-like filter. Then, robots are controlled so as to maximize a metric on the quality of the state estimate. For example, the work in [20] controls robots to move in the direction of the gradient of mutual information. More recently, in [21] and [22] vehicles are controlled to decrease the variance of their estimate of the state of the environment. This is accomplished in a distributed way by using average consensus estimators to propagate information among the robots. Similarly, in [23], the author proposes a gradient-based controller to decrease the variance of the estimate error in a time changing environment. In [24], sensing robots are coordinated to move in a formation along level curves of the environmental field. In [25], optimal trajectories over a finite

horizon are found if the environmental model satisfies a certain spatial separation property. In addition, in [26], the authors solve a dynamic program (DP) over a finite horizon to find the trajectory of a robot to minimize the variance of the estimate of the environment, and a similar DP approach was employed in [27] over short time horizons. Many other works exist in this vein.

Although these works are well motivated by the uncontested successes of Kalman filtering and Kriging in real-world estimation applications, they suffer from the fact that planning optimal trajectories under these models requires the solution of an intractable DP, even for a static environment. One must resort to myopic methods, such as gradient descent (as in [20]–[24]), or solve the DP approximately over a finite time horizon (as in [25]–[27]). Although these methods have great appeal from an estimation point of view, little can be proved about the comparative performance of the control strategies that are employed in these works. The approach that we take in this paper circumvents the question of estimation by formulating a new model of growing uncertainty in the environment. Under this model, we can solve the speed planning problem over *infinite time*, while maintaining *guaranteed* levels of uncertainty in a *time-changing* environment. Thus, we have used a less sophisticated environment model in order to obtain stronger results on the control strategy. Because our model is based on the analogy of dust collecting in an environment, we also solve infinite horizon sweeping problems with the same method.

Our problem in this paper is also related to sweep coverage, or lawn mowing and milling problems, in which robots with finite sensor footprints move over an environment so that every point in the environment is visited at least once by a robot. Lawn mowing and milling has been treated in [8] and other works. Sweep coverage has recently been studied in [28], and efficient sweep coverage algorithms are proposed for ant robots in [29]. A survey of sweep coverage is given in [30]. Our problem is significantly different from these because our environment is dynamic, thereby requiring continual remilling or resweeping. A different notion of persistent surveillance has been considered in [31] and [32], where a persistent task is defined as one whose completion takes much longer than the life of a robot. While the terminology is similar, our problem is more concerned with the task (sweeping or monitoring) itself than with power requirements of individual robots.

A problem that is more closely related to ours is that of patrolling [33], [34], where an environment must be continually surveyed by a group of robots, such that each point is visited with equal frequency. Similarly, in [35], vehicles must repeatedly visit the cells of a gridded environment. In addition, continual perimeter patrolling is addressed in [36]. In another related work [37], a region is persistently covered by controlling robots to move at constant speed along predefined paths. Our work is different from these, however, in that we treat the situation in which different parts of the environment may require different levels of attention. This is a significant difference as it induces a difficult resource tradeoff problem as one typically finds in queuing theory [38], or dynamic vehicle routing [39], [40]. In [41], the authors consider unequal frequency of visits in a gridded environment, but they control the robots using a greedy method that does not have performance guarantees.

Indeed, our problem can be seen as a dynamic vehicle routing problem with some unique features. Most importantly, we control the speed of the robots along a preplanned path, whereas the typical dynamic vehicle routing problem considers planning a path for vehicles that move at a constant speed. In addition, in our case, all the points under a robot's footprint are serviced simultaneously, whereas typically robots service one point at a time in dynamic vehicle routing. Finally, in our case, servicing a point takes an amount of time proportional to the size of the accumulation function at that point, whereas the service time of points in dynamic vehicle routing is typically independent of that point's wait time.

This paper is organized as follows. In Section II, we set up the problem and define some basic notions. In Section III, two LPs are formulated: one of which gives a stabilizing controller and the other one an optimal controller. Multiple robots are addressed in Section IV. The performance and robustness of the controllers are illustrated in simulations in Section V. Finally, in Section VI, we give conclusions and extensions.

II. Problem Formulation and Stability Notions

In this section, we formalize persistent tasks, introduce the notion of a field-stabilizing controller, and provide necessary and sufficient conditions for field stability.

A. Persistent Tasks

Consider a compact environment,[1] i.e., $\mathcal{E} \subset \mathbb{R}^2$, and a finite set of points of interest, i.e., $Q \subseteq \mathcal{E}$. The environment contains a closed curve, i.e., $\gamma : [0,1] \to \mathbb{R}^2$, where $\gamma(0) = \gamma(1)$ (see Fig. 2 for an illustration). The curve is parametrized by $\theta \in [0,1]$, and we assume without loss of generality that θ is the arc-length parametrization. The environment also contains a single robot (we will generalize to multiple robots in Section IV) whose motion is constrained along the path γ. The robot's position at a time t can be described by $\theta(t)$, its position along the curve γ. The robot is equipped with a sensor with finite footprint $\mathcal{B}(\theta) \subset \mathcal{E}$ (e.g., the footprint could be a disk of radius r centered at the robot's position).[2] Our objective is to control the speed v of the robot along the curve. We assume that for each point θ on the curve, the maximum possible robot speed is $v_{\max}(\theta)$, and the minimum robot speed is $v_{\min}(\theta) > 0$. This allows us to express constraints on the robot speed at different points on the curve. For example, for safety considerations, the robot may be required to move more slowly in certain areas of the environment, or on highly curved sections of the path. To summarize, the robot is described by the triple, i.e., $\mathcal{R} := (\mathcal{B}, v_{\min}, v_{\max})$.

A time-varying field, i.e., $Z : Q \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, which we call the *accumulation function*, is defined on the points of interest

[1] The results in this paper generalize directly to $\mathbb{R}^d$, $d > 0$, but we concentrate on $\mathbb{R}^2$ because of its relevance to real-world scenarios.

[2] For a ground robot or surface vessel, the footprint may be the robot's field of view or its cleaning surface. For a unmanned aerial vehicle (UAV) flying at a constant altitude over a 2-D environment, the footprint could give the portion of the environment that is viewable by the robot.

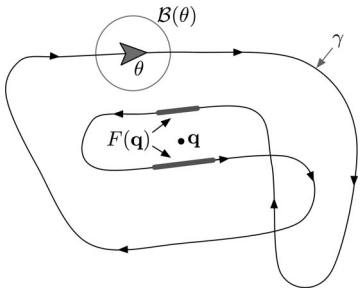

Fig. 2. Curve γ followed by the robot. The robot is located at θ and has footprint $\mathcal{B}(\theta)$. The set $F(\mathbf{q})$ of robot positions θ for which the footprint covers q are shown as thick gray segments of the curve.

Q. This field may describe a physical quantity, such as the amount of oil on the surface of a body of water. Alternatively, the field may describe the robot's uncertainty about the state of each point of interest. We assume that at each point $\mathbf{q} \in Q$, the field Z increases (or is produced) at a constant rate $p(\mathbf{q})$. When the robot footprint is covering $\mathbf{q}$, it consumes Z at a constant rate $c(\mathbf{q})$ so that when a point $\mathbf{q}$ is covered, the net rate of decrease is $p(\mathbf{q}) - c(\mathbf{q})$. Thus, Z evolves according to the following differential equation (with the initial conditions $Z(\mathbf{q}, 0)$ and $\theta(0)$):

$$\dot{Z}(\mathbf{q}, t) = \begin{cases} p(\mathbf{q}), & \text{if } \mathbf{q} \notin \mathcal{B}\big(\theta(t)\big) \\ p(\mathbf{q}) - c(\mathbf{q}), & \text{if } \mathbf{q} \in \mathcal{B}\big(\theta(t)\big) \text{ and } Z(\mathbf{q}, t) > 0 \\ 0, & \text{if } \mathbf{q} \in \mathcal{B}\big(\theta(t)\big) \text{ and } Z(\mathbf{q}, t) = 0 \end{cases} \tag{1}$$

where for each $\mathbf{q} \in Q$, we have $c(\mathbf{q}) > p(\mathbf{q}) > 0$. Note that this is a general equation for the evolution of a nonnegative scalar quantity that can grow or decrease linearly in time at a set of points. We make no assumptions as to the relation of the accumulation at different points based on spatial proximity. In this respect our model is different than typical spatial process models, such as Gaussian processes, spatial splines, or basis function representations. The model has the advantage that it is simple and requires the knowledge of relatively few parameters—only the production and consumption rates, i.e., $p(\mathbf{q})$ and $c(\mathbf{q})$, must be known. Its main limitation is that it is linear in time; a completely general process may accumulate in a temporally nonlinear fashion. However, our model can be viewed as a first-order approximation to processes that accumulate nonlinearly in time. Furthermore, the precision of the model is not critical because we show both analytically and in simulation that our controllers are robust to significant modeling errors.

In this paper, we assume that we know the model parameters $p(\mathbf{q})$ and $c(\mathbf{q})$. It is reasonable to assume knowledge of $c(\mathbf{q})$ since it pertains to the performance of the robot. For example, in an oil cleanup application, the consumption rate of oil of the robot can be measured in a laboratory environment or in field trials prior to control design. As for the production rate $p(\mathbf{q})$, this must be estimated from the physics of the environment, from a human expert (e.g., an oil mining engineer in the case of an oil-well leak), or it can be measured in a preliminary survey of the environment. However, as mentioned earlier, the accuracy of the model is not crucial because of the robustness of our method with respect to errors in $p(\mathbf{q})$. With this notation, we can now formally define a *persistent task*.

Definition 2.1 (Persistent Tasks): A persistent task is a tuple $(\mathcal{R}, \gamma, Q, p, c)$, where $\mathcal{R}$ is the robot model, γ is the curve followed by the robot, Q is the set of points of interest, and p and c are the production and consumption rates of the field, respectively.

In general, for a given persistent task, the commanded speed v could depend on the current position θ, the field Z, the initial conditions $\theta(0)$ and $Z(\mathbf{q}, 0)$, and time t. Thus, defining the set of initial conditions as $\mathrm{IC} := (\theta(0), Z(\mathbf{q}, 0))$, a general controller has the form $v(\theta, Z, \mathrm{IC}, t)$. For convenience, the notation that is used in this paper is summarized in Table I.

B. Field Stability and Feasibility

In this section, we formalize the problem of stabilizing the field in a persistent task. As a first consideration, a suitable controller should keep the field bounded everywhere, independent of the initial conditions. This motivates the following definition of stability.

Definition 2.2 (Field-Stabilizing Controller): A speed controller field stabilizes a persistent task if the field is always eventually bounded, independent of initial conditions. That is, if there exists a $Z_{\max} < +\infty$, such that for every $\mathbf{q} \in Q$ and initial conditions $Z(\mathbf{q}, 0)$ and $\theta(0)$, we have

$$\limsup_{t \to +\infty} Z(\mathbf{q}, t) \leq Z_{\max}.$$

Note that in this definition of stability, for every initial condition, the field eventually enters the interval $[0, Z_{\max}]$.

There are some persistent tasks for which no controller is field stabilizing. This motivates the notion of *feasibility*.

Definition 2.3 (Feasible Persistent Task): A persistent task is feasible if there exists a field-stabilizing speed controller.

As stated earlier, for a given persistent task, a general speed controller can be written as $v(\theta, Z, \mathrm{IC}, t)$. However, in the remainder of this paper, we will focus on a small subset of speed controllers, which we call *periodic position-feedback controllers*. In these controllers, the speed only depends on the robot's current position $\theta \in [0, 1]$. The controllers are periodic in the sense that the speed at a point θ is the same on each cycle of the path. The controller can be written as

$$v : [0, 1] \to \mathbb{R}_{>0},$$

where each $\theta \in [0, 1]$ is mapped to a speed $v(\theta)$ satisfying the bounds $v_{\min}(\theta) \leq v(\theta) \leq v_{\max}(\theta)$. These controllers have the advantage that they do not require information on the current state of the field Z, but only on its model parameters $p(\mathbf{q})$ and $c(\mathbf{q})$. While it may seem restrictive to limit our controllers to this special form, the following result shows that it is not.

Proposition 2.4 (Periodic Position-Feedback Controllers): If a persistent task can be stabilized by a general controller $v(\theta, Z, \mathrm{IC}, t)$, then it can be stabilized by a periodic position-feedback controller $v(\theta)$.

The proof of Proposition 2.4 is given in Appendix A and relies on the statement and proof of the upcoming result in

TABLE I
TABLE OF SYMBOLS

Symbol	Definition
$\mathcal{E}$	the environment.
Q	a set of points of interest in the environment $\mathcal{E}$.
$\mathbf{q}$	a point of interest in Q.
γ	a curve followed by the robot, and parametrized by θ.
$v(\theta)$	the speed of the robot along the curve γ.
$\mathcal{R}$	the robot model, consisting of minimum and maximum speeds $v_{\min}(\theta)$ and $v_{\max}(\theta)$, and a robot footprint $\mathcal{B}(\theta)$.
$\mathcal{B}(\theta)$	The set of points in the environment covered by the robot's footprint when positioned at θ.
$F(\mathbf{q})$	the values of $\theta \in [0,1]$ along the curve γ for which the point $\mathbf{q}$ is covered by the footprint.
$Z(\mathbf{q},t)$	the field (or accumulation function) at point $\mathbf{q} \in Q$ at time t.
$p(\mathbf{q})$	the rate of production of the field Z at point $\mathbf{q}$.
$c(\mathbf{q})$	the rate of consumption of the field Z when $\mathbf{q}$ is covered by the robot footprint.
$\beta(\theta)$	a basis function.
α	a basis function coefficient, or parameter.
T	the time to complete one full cycle of the curve.
$\tau(\mathbf{q})$	the amount of time the point $\mathbf{q} \in Q$ is covered per cycle.
$\mathcal{H}(v)$	cost function (10) giving the steady-state maximum value of $Z(\mathbf{q},t)$

Lemma 2.5. Therefore, we encourage the reader to postpone reading the proof until after Lemma 2.5.

We will now investigate conditions for a controller to be field stabilizing and for a persistent task to be feasible. Let us define a function that maps each point, i.e., $\mathbf{q} \in Q$, to the curve positions θ for which $\mathbf{q}$ is covered by the robot footprint. To this end, we define

$$F(\mathbf{q}) := \{\theta \in [0,1] \mid \mathbf{q} \in \mathcal{B}(\theta)\}.$$

An illustration of the curve, the robot footprint, and the set $F(\mathbf{q})$ is shown in Fig. 2. (In Section V, we discuss how this set can be computed in practice.)

Given a controller $\theta \mapsto v(\theta)$, we define two quantities: 1) cycle time, or period T and 2) coverage time per cycle $\tau(\mathbf{q})$. Since $v(\theta) > 0$ for all θ, the robot completes one full cycle of the closed curve in time

$$T := \int_0^1 \frac{1}{v(\theta)} d\theta. \tag{2}$$

During each cycle, the robot's footprint is covering the point $\mathbf{q}$ only when $\theta(t) \in F(\mathbf{q})$. Thus, the point $\mathbf{q}$ is covered for

$$\tau(\mathbf{q}) := \int_{F(\mathbf{q})} \frac{1}{v(\theta)} d\theta \tag{3}$$

time units during each complete cycle.

With these definitions, we can give a necessary and sufficient condition for a controller to stabilize a persistent task. In Section III, we develop a method for testing if this condition can be satisfied by a speed controller.

Lemma 2.5 (Stability Condition): Given a persistent task, a controller $v(\theta)$ is field stabilizing if and only if

$$c(\mathbf{q}) \int_{F(\mathbf{q})} \frac{1}{v(\theta)} d\theta > p(\mathbf{q}) \int_0^1 \frac{1}{v(\theta)} d\theta \tag{4}$$

for every $\mathbf{q} \in Q$. Applying the definitions in (2) and (3), the condition can be expressed as $\tau(\mathbf{q}) > \big(p(\mathbf{q})/c(\mathbf{q})\big)T$.

The lemma has a simple intuition. For stability, the field consumption per cycle must exceed the field production per cycle, for each point $\mathbf{q} \in Q$. We now prove the result.

Proof: Consider a point $\mathbf{q} \in Q$, and the set of curve positions $F(\mathbf{q})$ for which $\mathbf{q}$ is covered by the robot footprint. Given the speed controller v, we can compute the cycle time T in (2). Then, let us consider the change in the field from $Z(\mathbf{q},t)$ to $Z(\mathbf{q},t+T)$, where $t \geq 0$.

Define an indicator function, i.e., $\mathbf{I} : [0,1] \times Q \to \{0,1\}$, as $\mathbf{I}(\theta,\mathbf{q}) = 1$ for $\theta \in F(\mathbf{q})$ and 0 otherwise. Then, from (1), we have that

$$\dot{Z}(\mathbf{q},t) \geq p(\mathbf{q}) - c(\mathbf{q})\mathbf{I}(\theta(t),\mathbf{q})$$

for all values of $Z(\mathbf{q},t)$, with equality if $Z(\mathbf{q},t) > 0$. By the integration of the aforementioned expression over $[t, t+T]$, we see that

$$\begin{aligned} Z(\mathbf{q},t+T) - Z(\mathbf{q},t) &\geq p(\mathbf{q})T - c(\mathbf{q}) \int_t^{t+T} \mathbf{I}(\theta(\tau),\mathbf{q}) d\tau \\ &= p(\mathbf{q})T - c(\mathbf{q})\tau(\mathbf{q}) \end{aligned} \tag{5}$$

where $\tau(\mathbf{q})$ is defined in (3), and the equality follows from the fact that $\mathbf{I}(\theta(t),\mathbf{q})$ is simply an indicator function on whether or not the footprint is covering $\mathbf{q}$ at a given time. From (5), we see that for the field to be eventually bounded by some $Z_{\max}$ for all initial conditions $Z(\mathbf{q},0)$, we require that $\tau(\mathbf{q}) > p(\mathbf{q})/c(\mathbf{q})T$ for all $\mathbf{q} \in Q$.

To see that the condition is also sufficient, suppose that $\tau(\mathbf{q}) > p(\mathbf{q})/c(\mathbf{q})T$. Then, there exists $\epsilon > 0$, such that $p(\mathbf{q})T - c(\mathbf{q})\tau(\mathbf{q}) = -\epsilon$. If $Z(\mathbf{q},t) > \big(c(\mathbf{q}) - p(\mathbf{q})\big)T$, then the field at the point $\mathbf{q} \in Q$ is strictly positive over the entire interval $[t, t+T]$, implying that

$$Z(\mathbf{q},t+T) = Z(\mathbf{q},t) - \epsilon.$$

Thus, from every initial condition, $Z(\mathbf{q},t)$ moves below $\big(c(\mathbf{q}) - p(\mathbf{q})\big)T$. Additionally, note that for each $\bar{t}$ in the interval $[t, t+T]$, we trivially have that $Z(\mathbf{q},\bar{t}) \leq Z(\mathbf{q},t) + p(\mathbf{q})T$. Thus, we have that there exists a finite time $\bar{t}$, such that for all $t \geq \bar{t}$

$$Z(\mathbf{q},t) \leq \big(c(\mathbf{q}) - p(\mathbf{q})\big)T + p(\mathbf{q})T = c(\mathbf{q})T.$$

Since Q is finite, there exists a single $\epsilon > 0$, such that for every point $\mathbf{q} \in Q$, we have $\tau(\mathbf{q}) - p(\mathbf{q})/c(\mathbf{q})T > \epsilon$. Hence, letting $Z_{\max} = \max_{\mathbf{q} \in Q} c(\mathbf{q})T$, we see that Z is stable for all $\mathbf{q}$, completing the proof. ■

In the following sections, we will address two problems, determining a field-stabilizing controller, and determining a minimizing controller, defined as follows.

Problem 2.6 (Persistent Task Metrics): Given a persistent task, determine a periodic position-feedback controller, i.e., $v : [0,1] \to \mathbb{R}_{>0}$, that satisfies the speed constraints (i.e., $v(\theta) \in \left[v_{\min}(\theta), v_{\max}(\theta)\right]$ for all $\theta \in [0,1]$), and

1) is field stabilizing, or
2) minimizes the maximum steady-state field $\mathcal{H}(v)$:

$$\mathcal{H}(v) := \max_{\mathbf{q}\in Q} \left(\limsup_{t\to+\infty} Z(\mathbf{q},t) \right).$$

In Section III, we will show that by writing the speed controller in terms of a set of basis functions, problems 1 and 2 can be solved using LPs. In Section IV, we will solve problem 1 for multiple robots.

III. Single-Robot Speed Controllers: Stability and Optimality

We focus on environments, which contain a finite set of points of interest, i.e., $Q = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$. These m locations could be specific regions of interest, or they could be a discrete approximation of the continuous space that is obtained by, e.g., laying a grid down on the environment. In Section V, we will show examples of both scenarios. Our two main results are given in Theorems 3.1 and 3.8, which show that a field-stabilizing controller and a controller minimizing $\mathcal{H}(v)$ can be found by solving an appropriate LP.

To begin, it will be more convenient to consider the reciprocal speed controller, i.e., $v^{-1}(\theta) := 1/v(\theta)$, with its corresponding constraints

$$\frac{1}{v_{\max}(\theta)} \le v^{-1}(\theta) \le \frac{1}{v_{\min}(\theta)}.$$

Now, our approach is to consider a finite set of basis functions $\{\beta_1(\theta), \ldots, \beta_n(\theta)\}$. Example basis functions include (a finite subset of) the Fourier basis or Gaussian basis [13]. In what follows, we will use rectangular functions as the basis:

$$\beta_j(\theta) = \begin{cases} 1, & \text{if } \theta \in [(j-1)/n, j/n) \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

for each $j \in \{1, \ldots, n\}$. This basis, which provides a piecewise constant approximation to a curve, has the advantage that we will easily be able to incorporate the speed constraints $v_{\min}(\theta)$ and $v_{\max}(\theta)$ into the controller.

Then, let us consider reciprocal speed controllers of the form

$$v^{-1}(\theta) = \sum_{j=1}^{n} \alpha_j \beta_j(\theta) \tag{7}$$

where $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ are the free parameters that we will use to optimize the speed controller. A rich class of functions can be represented as a finite linear combination of basis functions though not all functions can be represented this way. Limiting our speed controller to a linear parametrization allows us to find an optimal controller within that class, while preserving enough generality to give complex solutions that would be difficult to find in an ad hoc manner. In the following section, we will consider the problem of synthesizing a field-stabilizing controller.

A. Synthesis of a Field-Stabilizing Controller

In this section, we will show that a field-stabilizing speed controller of the form (7) can be found through the solution of a LP. This result is summarized in Theorem 3.1. We remind the reader that a summary of the mathematical symbols and their definitions is shown in Table I.

To begin, let us consider reciprocal speed controllers in the form of (7). Then, for $\mathbf{q}_i \in Q$, the stability condition in Lemma 2.5 becomes

$$\sum_{j=1}^{n} \alpha_j \int_{F(\mathbf{q}_i)} \beta_j(\theta) d\theta > \frac{p(\mathbf{q}_i)}{c(\mathbf{q}_i)} \sum_{j=1}^{n} \alpha_j \int_0^1 \beta_j(\theta) d\theta.$$

By the rearrangement, we obtain

$$\sum_{j=1}^{n} \alpha_j K(\mathbf{q}_i, \beta_j) > 0$$

where we have defined

$$K(\mathbf{q}_i, \beta_j) := \int_{F(\mathbf{q}_i)} \beta_j(\theta) d\theta - \frac{p(\mathbf{q}_i)}{c(\mathbf{q}_i)} \int_0^1 \beta_j(\theta) d\theta. \tag{8}$$

Finally, to satisfy the speed constraints, we have that

$$\frac{1}{v_{\max}(\theta)} \le \sum_{j=1}^{n} \alpha_j \beta_j(\theta) \le \frac{1}{v_{\min}(\theta)}. \tag{9}$$

For the rectangular basis in (6), the speed constraints become

$$\frac{1}{v_{\max}(j)} \le \alpha_j \le \frac{1}{v_{\min}(j)}$$

where

$$v_{\max}(j) = \inf_{\theta\in[(j-1)/n, j/n)} v_{\max}(\theta) \text{ and}$$

$$v_{\min}(j) = \sup_{\theta\in[(j-1)/n, j/n)} v_{\min}(\theta).$$

Thus, we obtain the following result.

Theorem 3.1 (Existence of a Field-Stabilizing Controller): A persistent task is stabilizable by a speed controller of the form (7) if and only if the following LP is feasible:

$$\begin{aligned} \text{minimize} \quad & 0 \\ \text{subject to} \quad & \sum_{j=1}^{n} \alpha_j K(\mathbf{q}_i, \beta_j) > 0 \quad \forall\, i \in \{1, \ldots, m\} \\ & \frac{1}{v_{\max}(j)} \le \alpha_j \le \frac{1}{v_{\min}(j)} \quad \forall\, j \in \{1, \ldots, n\} \end{aligned}$$

where $K(\mathbf{q}_i, \beta_j)$ is defined in (8), and $\alpha_1, \ldots, \alpha_n$ are the optimization variables.

Hence, we can solve for a field-stabilizing controller using a simple LP. The program has n variables (one for each basis function coefficient), and $2n + m$ constraints (two for each basis function coefficient, and one for each point of interest in Q). One can easily solve LPs with thousands of variables and constraints [12]. Thus, the problem of computing a field-stabilizing

controller can be solved for finely discretized environments with thousands of basis functions. Note that in the aforementioned lemma, we are only checking feasibility, and thus, the cost function in the optimization is arbitrary. For simplicity, we write the cost as 0.

In Theorem 3.1, the cost is set to 0 to highlight the feasibility constraints. However, in practice, an important consideration is *robustness* of the controller to uncertainty, and error in the model of the field evolution and in the motion of the robot. Robustness of this type can be achieved by slightly altering the aforementioned optimization to maximize the *stability margin*. This is outlined in the following result.

Corollary 3.2 (Robustness via Maximum Stability Margin): The optimization,

$$\begin{aligned} \text{maximize} \quad & B \\ & \sum_{j=1}^{n} \alpha_j K(\mathbf{q}_i, \beta_j) \geq B \quad \forall\, i \in \{1, \ldots, m\} \\ & \frac{1}{v_{\max}(j)} \leq \alpha_j \leq \frac{1}{v_{\min}(j)} \quad \forall\, j \in \{1, \ldots, n\} \end{aligned}$$

where $\alpha_1, \ldots, \alpha_n$ and B are the optimization variables, yields a speed controller that maximizes the stability margin, i.e., $\min_{\mathbf{q}_i \in Q} \sum_{j=1}^{n} \alpha_j K(\mathbf{q}_i, \beta_j)$. The controller has the following features.

1) It has the largest decrease in $Z(\mathbf{q}_i, t)$ per cycle and, thus, achieves the steady state in the minimum number of cycles.
2) It is robust to errors in estimating the field production rate. If the robot's estimate of the production rate at a field point $\mathbf{q}_i \in Q$ is $\bar{p}(\mathbf{q}_i)$, and the true value is $p(\mathbf{q}_i) \leq \bar{p}(\mathbf{q}_i) + \epsilon$, then the field is stable that is provided that for each $\mathbf{q}_i \in Q$

$$\epsilon < B \cdot c(\mathbf{q}_i) \left(\sum_{j=1}^{n} \alpha_j \int_0^1 \beta_j(\theta) d\theta \right)^{-1}.$$

Proof: The first property follows directly from the fact that $\sum_{j=1}^{n} \alpha_j K(\mathbf{q}_i, \beta_j)$ is the amount of decrease of the field at point $\mathbf{q}_i$ in one cycle.

To see the second property, suppose that the optimization is based on a production rate $\bar{p}(\mathbf{q}_i)$ for each $\mathbf{q}_i \in Q$, but the true production rate is given by $p(\mathbf{q}_i) := \bar{p}(\mathbf{q}_i) + \epsilon$, where $\epsilon > 0$. In solving the optimization that is presented earlier, we obtain a controller with a stability margin of $B > 0$. For the true system to be stable, we require that for each $\mathbf{q}_i \in Q$,

$$\sum_{j=1}^{n} \alpha_j \int_{F(\mathbf{q}_i)} \beta_j(\theta) d\theta > \frac{\bar{p}(\mathbf{q}_i) + \epsilon}{c(\mathbf{q}_i)} \sum_{j=1}^{n} \alpha_j \int_0^1 \beta_j(\theta) d\theta.$$

This can be rewritten as

$$\sum_{j=1}^{n} \alpha_j K(\mathbf{q}_i, \beta_j) > \frac{\epsilon}{c(\mathbf{q}_i)} \sum_{j=1}^{n} \alpha_j \int_0^1 \beta_j(\theta) d\theta$$

from which we see that the true field is stable, provided that $\epsilon < Bc(\mathbf{q}_i) / \left(\sum_{j=1}^{n} \alpha_j \int_0^1 \beta_j(\theta) d\theta \right)$ for each $\mathbf{q}_i \in Q$. ∎

Remark 3.3 (Alternative Basis Functions): The constraints on α_j in (9) ensure that the robot does not violate the speed limits. In general, one would have such a speed constraint at every possible position $\theta \in [0, 1]$, thereby giving an infinite number of constraints. We avoided this problem by using basis functions that are of constant values over an interval, thus all the speed constraints over the interval can be represented by a single constraint. However, for a general set of basis functions, the optimization in Theorem 3.1 cannot be solved exactly because of these infinite constraints. In this case, an effective method is simply to enforce the constraints in (9) for a finite number of θ values, i.e., $\theta_1, \ldots, \theta_w$. Each θ_i generates two constraints in the optimization. Then, we can tighten each constraint by $\xi > 0$, yielding $1/v_{\max}(\theta_i) + \xi \leq \sum_j \alpha_j \beta_j(\theta_i) \leq 1/v_{\min}(\theta_i) - \xi$ for each $i \in \{1, \ldots, w\}$. Choosing the number of θ values w as large as possible, given the available computational resources, we can then increase ξ until the controller satisfies the original speed constraints.

B. Synthesis of an Optimal Controller

In this section, we look at point 2 of Problem 2.6, which is to minimize the maximum value that is attained by the field over the finite region of interest Q. That is, for a given persistent task, our goal is to minimize the following cost function:

$$\mathcal{H}(v) = \max_{\mathbf{q} \in Q} \left(\limsup_{t \to +\infty} Z(\mathbf{q}, t) \right) \tag{10}$$

over all possible speed controllers v. At times, we will represent the maximum steady-state value for a point $\mathbf{q}$ using a speed controller v as $\mathcal{H}(\mathbf{q}, v) := \limsup_{t \to +\infty} Z(\mathbf{q}, t)$.

Our main result of this section, i.e., Theorem 3.8, is that $\mathcal{H}(v)$ can be minimized through a LP. However, we must establish some intermediate results. First, we show that if v is field stabilizing, then for every initial condition, there exists a finite time t^*, such that $Z(\mathbf{q}, t) \leq \mathcal{H}(v)$ for all $t \geq t^*$.

Proposition 3.4 (Steady-State Field): Consider a feasible persistent task and a field-stabilizing speed controller. Then, there is a steady-state field

$$\bar{Z} : Q \times [0, 1] \to \mathbb{R}_{\geq 0}$$

satisfying the following statements for each $\mathbf{q} \in Q$.

1) For every set of initial conditions $\theta(0)$ and $Z(\mathbf{q}, 0)$, there exists a time $t^* \geq 0$, such that

$$Z(\mathbf{q}, t) = \bar{Z}\big(\mathbf{q}, \theta(t)\big), \quad \text{for all } t \geq t^*.$$

2) There exists at least one $\theta \in [0, 1]$, such that $\bar{Z}(\mathbf{q}, \theta) = 0$.

From the aforementioned result, we see that from every initial condition, the field converges in finite time to a steady state $\bar{Z}(\mathbf{q}, \theta)$. In the steady state, the field $Z(\mathbf{q}, t)$ at time t depends only on $\theta(t)$ [and is independent of $Z(\mathbf{q}, 0)$]. Each time the robot is located at θ, the field is given by $\bar{Z}(\mathbf{q}, \theta)$. Moreover, the result states that in the steady state, there is always a robot position at which the field is reduced to zero. In order to prove Proposition 3.4, we begin with the following lemma. Recall that the cycle time for a speed controller v is $T := \int_0^1 1/v(\theta) d\theta$.

Lemma 3.5 (Field Reduced to Zero): Consider a feasible persistent task and a field-stabilizing speed controller. For every $\mathbf{q} \in Q$ and every set of initial conditions $Z(\mathbf{q}, 0)$ and $\theta(0)$,

there exists a time $t^* > T$, such that

$$Z(\mathbf{q}, t^* + aT) = 0 \tag{11}$$

for all nonnegative integers a.

Proof: Consider any $\mathbf{q} \in Q$, and the initial conditions $Z(\mathbf{q}, 0)$ and $\theta(0)$, and suppose by way of contradiction that the speed controller is stable but that $Z(\mathbf{q}, t) > 0$ for all $t > T$. From Lemma 2.5, if the persistent task is stable, then $c(\mathbf{q})\tau(\mathbf{q}) > p(\mathbf{q})T$ for all $\mathbf{q}$. Thus, there exists $\epsilon > 0$, such that $c(\mathbf{q})\tau(\mathbf{q}) - p(\mathbf{q})T > \epsilon$ for all $\mathbf{q} \in Q$. From the proof of Lemma 2.5, we have that

$$Z(\mathbf{q}, t + T) - Z(\mathbf{q}, t) = p(\mathbf{q})T - c(\mathbf{q})\tau(\mathbf{q}) = -\epsilon.$$

Therefore, given $Z(\mathbf{q}, 0)$, we have that $Z(\mathbf{q}, t^*) = 0$ for some finite $t^* > T$, which is a contradiction.

Next, we will verify that if $Z(\mathbf{q}, t^*) = 0$ for some $t^* > T$, then $Z(\mathbf{q}, t^* + T) = 0$. To see this, note that the differential equation (1) is piecewise constant. Given a speed controller $v(\theta)$, the differential equation is time invariant and admits unique solutions.

Based on this, consider the following two initial conditions for (1):

$$Z_1(\mathbf{q}, 0) := Z(\mathbf{q}, t^* - T) \geq 0, \quad \theta_1(0) := \theta(t^* - T) = \theta(t^*)$$

and

$$Z_2(\mathbf{q}, 0) := Z(\mathbf{q}, t^*) = 0, \quad \theta_2(0) := \theta(t^*).$$

Since (1) is time invariant, we have that $Z_1(\mathbf{q}, T) = Z(\mathbf{q}, t^*) = 0$, and $Z_2(\mathbf{q}, T) = Z(\mathbf{q}, t^* + T)$. In addition, by uniqueness of solutions, we have that $Z_1(\mathbf{q}, 0) \geq Z_2(\mathbf{q}, 0)$ implies that $Z_1(\mathbf{q}, T) \geq Z_2(\mathbf{q}, T)$. Thus, we have that $Z(\mathbf{q}, t^*) = 0 \geq Z(\mathbf{q}, t^* + T)$, proving the desired result. ∎

The previous lemma shows that from every initial condition, there exists a finite time t^*, after which the field at a point $\mathbf{q}$ is reduced to zero in each cycle. With this lemma, we can prove Proposition 3.4.

Proof of Proposition 3.4: In Lemma 3.5, we have shown that for every set of initial conditions $Z(\mathbf{q}, 0)$ and $\theta(0)$, there exists at time $t^* > T$, such that $Z(\mathbf{q}, t^* + aT) = 0$ for all nonnegative integers a. Since T is the cycle time for the robot, we also know that $\theta(t^* + aT) = \theta(t^*)$ for all a. Since (1) yields unique solutions, (11) uniquely defines $Z(\mathbf{q}, t)$ for all $t \geq t^*$, with

$$Z(\mathbf{q}, t + T) = Z(\mathbf{q}, t) \quad \text{for all } t \geq t^*.$$

Hence, we can define the steady-state profile $\bar{Z}(\mathbf{q}, \theta)$ as

$$\bar{Z}\big(\mathbf{q}, \theta(t)\big) := Z(\mathbf{q}, t) \quad \text{for all } t \in [t^*, t^* + T).$$

Finally, we need to verify that $\bar{Z}(\mathbf{q}, \theta)$ is independent of initial conditions. To proceed, suppose by way of contradiction that there are two sets of initial conditions $\theta_1(0)$, $Z_1(\mathbf{q}, 0)$, and $\theta_2(0)$, $Z_2(\mathbf{q}, 0)$, which yield different steady-state fields $\bar{Z}_1(\mathbf{q}, \theta)$ and $\bar{Z}_2(\mathbf{q}, \theta)$. That is, there exists $\tilde{\theta}$, such that $\bar{Z}_1(\mathbf{q}, \tilde{\theta}) \neq \bar{Z}_2(\mathbf{q}, \tilde{\theta})$. Without loss of generality, assume that $\bar{Z}_1(\mathbf{q}, \tilde{\theta}) > \bar{Z}_2(\mathbf{q}, \tilde{\theta})$. To obtain a contradiction, we begin by showing that this implies $\bar{Z}_1(\mathbf{q}, \theta) \geq \bar{Z}_2(\mathbf{q}, \theta)$ for all θ. Note that Z_1 and Z_2 reach their steady-state profiles $\bar{Z}_1$ and $\bar{Z}_2$ in finite time. Thus, there exist times $t_1, t_2 \geq 0$ for which $\theta_1(t_1) = \theta_2(t_2) = \tilde{\theta}$, and $Z_1(\mathbf{q}, t_1) = \bar{Z}_1(\mathbf{q}, \tilde{\theta})$, and $Z_2(\mathbf{q}, t_2) = \bar{Z}_2(\mathbf{q}, \tilde{\theta})$. Since $Z_1(\mathbf{q}, t_1) > Z_2(\mathbf{q}, t_2)$, and since Z is a continuous function of time, either 1) $Z_1(\mathbf{q}, t_1 + t) > Z_2(\mathbf{q}, t_2 + t)$ for all $t \geq 0$, or 2) there exists a time $\bar{t} > 0$ for which $Z_1(\mathbf{q}, t_1 + \bar{t}) = Z_2(\mathbf{q}, t_2 + \bar{t})$, which by uniqueness of solutions implies $Z_1(\mathbf{q}, t_1 + t) = Z_2(\mathbf{q}, t_2 + t)$ for all $t \geq \bar{t}$. Thus, $Z_1(\mathbf{q}, t_1 + t) \geq Z_2(\mathbf{q}, t_2 + t)$ for all $t \geq 0$, implying that $\bar{Z}_1(\mathbf{q}, \theta) \geq \bar{Z}_2(\mathbf{q}, \theta)$ for all θ.

From Lemma 3.5, there exists a $\bar{\theta}$ for which $\bar{Z}_1(\mathbf{q}, \bar{\theta}) = 0$. Since, $\bar{Z}_1(\mathbf{q}, \theta) \geq \bar{Z}_2(\mathbf{q}, \theta)$ for all θ, we must have that $\bar{Z}_2(\mathbf{q}, \bar{\theta}) = 0$. However, the value of Z_1 and Z_2 at $\bar{\theta}$ uniquely defines $\bar{Z}_1$ and $\bar{Z}_2$ for all θ, implying that $\bar{Z}_1(\mathbf{q}, \theta) = \bar{Z}_2(\mathbf{q}, \theta)$, which is a contradiction. ∎

From Proposition 3.4, we have shown the existence of a steady-state field $\bar{Z}(\mathbf{q}, \theta)$ that is independent of initial conditions $Z(\mathbf{q}, 0)$ and $\theta(0)$.

Now, consider a point $\mathbf{q} \in Q$ and a field-stabilizing speed controller $v(\theta)$, and let us solve for its steady-state field $\bar{Z}(\mathbf{q}, \theta)$. To begin, let us write $F(\mathbf{q})$ (the set of θ values for which the point $\mathbf{q}$ is covered by the footprint) as a union of disjoint intervals:

$$F(\mathbf{q}) = [x_1, y_1] \cup [x_2, y_2] \cup \cdots \cup [x_\ell, y_\ell] \tag{12}$$

where ℓ is a positive integer, and $y_k > x_k > y_{k-1}$ for each $k \in \{1, \ldots, \ell\}$.[3] Thus, on the intervals $[x_k, y_k]$, the point $\mathbf{q}$ is covered by the robot footprint, and on the intervals $[y_k, x_{k+1}]$, the point $\mathbf{q}$ in uncovered. As an example, in Fig. 2, the set $F(\mathbf{q})$ consists of two intervals, and thus, $\ell = 2$. An example of a speed controller and an example of a set $F(\mathbf{q})$ are shown in Fig. 3(a) and (b).

From differential equation (1), we can write

$$\bar{Z}(\mathbf{q}, x_k) = \bar{Z}(\mathbf{q}, y_{k-1}) + p(\mathbf{q}) \int_{y_{k-1}}^{x_k} \frac{d\theta}{v(\theta)} \tag{13}$$

$$\bar{Z}(\mathbf{q}, y_k) = \left(\bar{Z}(\mathbf{q}, x_k) + \big(p(\mathbf{q}) - c(\mathbf{q})\big) \int_{x_k}^{y_k} \frac{d\theta}{v(\theta)} \right)^+ \tag{14}$$

where for $z \in \mathbb{R}$, we define $(z)^+ := \max\{z, 0\}$. Combining equations (13) and (14), we see that

$$\bar{Z}(\mathbf{q}, y_k) = \left(\bar{Z}(\mathbf{q}, y_{k-1}) + p(\mathbf{q}) \int_{y_{k-1}}^{y_k} \frac{d\theta}{v(\theta)} - c(\mathbf{q}) \int_{x_k}^{y_k} \frac{d\theta}{v(\theta)} \right)^+. \tag{15}$$

For each $b \in \{1, \ldots, \ell\}$, let us define[4]

$$N_{k-b,k}(\mathbf{q}) := p(\mathbf{q}) \int_{y_{k-b}}^{y_k} \frac{d\theta}{v(\theta)} - c(\mathbf{q}) \sum_{w=0}^{b-1} \int_{x_{k-w}}^{y_{k-w}} \frac{d\theta}{v(\theta)}. \tag{16}$$

Note that we can write $\bar{Z}(\mathbf{q}, y_k) = \big(\bar{Z}(\mathbf{q}, y_{k-1}) + N_{k-1,k}(\mathbf{q})\big)^+$, and from (15), we have

$$\bar{Z}(\mathbf{q}, y_k) \geq \bar{Z}(\mathbf{q}, y_{k-b}) + N_{k-b,k}(\mathbf{q}). \tag{17}$$

[3]Note that the number of intervals ℓ and the points $x_1, \ldots, x_\ell$ and $y_1, \ldots, y_\ell$ are a function of $\mathbf{q}$. However, for simplicity of notation, we will omit writing the explicit dependence.

[4]In this definition, and in what follows, addition and subtraction in the indices is performed modulo ℓ. Therefore, if $k = 1$, then $N_{k-1,k} = N_{\ell,1}$.

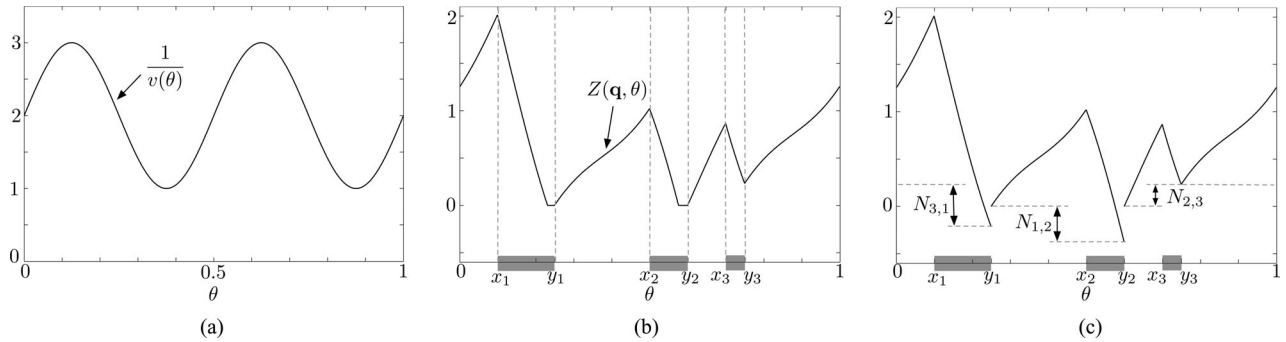

Fig. 3. Steady-state field for a point $\mathbf{q} \in Q$. The set of θ values for which $\mathbf{q}$ is covered is given by $F(\mathbf{q}) = [x_1, y_1] \cup [x_2, y_2] \cup [x_3, y_3]$. The field is produced at a rate $p(\mathbf{q}) = 3$ and is consumed by the footprint at a rate $c(\mathbf{q}) = 8.5$. (a) Sample reciprocal speed controller $v(\theta)$. (b) Steady-state field $\bar{Z}(\mathbf{q}, \theta)$. The set $F(\mathbf{q})$ consists of three intervals, which are shaded on the θ-axis. The steady-state profile is increasing outside of $F(\mathbf{q})$ and decreasing inside $F(\mathbf{q})$. (c) Calculation of quantities $N_{1,2}$, $N_{2,3}$, and $N_{3,1}$. The values represent the maximum reduction from y_{k-1} to y_k. Thus, $N_{1,2}, N_{3,1} < 0$, while $N_{2,3} > 0$.

Moreover

$$\bar{Z}(\mathbf{q}, y_k) = \bar{Z}(\mathbf{q}, y_{k-b}) + N_{k-b,k}(\mathbf{q}) \quad \text{if } Z(\mathbf{q}, y_{k-j}) > 0 \text{ for all } j \in \{1, \ldots, b-1\}. \tag{18}$$

Thus, we see that the quantity $N_{k-b,k}(\mathbf{q})$ gives the maximum reduction in the field between $\theta = y_{k-b}$ and $\theta = y_k$. An example for $b = 1$ is shown in Fig. 3(c). With these definition, we can characterize the steady-state field at the points y_k.

Lemma 3.6 (Steady-State Field at Points y_k): Given a feasible persistent task and a field-stabilizing speed controller, consider a point $\mathbf{q} \in Q$ and the set, i.e., $F(\mathbf{q}) = \cup_{k=1}^{\ell}[x_k, y_k]$. Then, for each $k \in \{1, \ldots, \ell\}$, we have

$$\bar{Z}(\mathbf{q}, y_k) = \max_{b \in \{0, \ldots, \ell-1\}} N_{k-b,k}(\mathbf{q})$$

where $N_{k-b,k}(\mathbf{q})$ is defined in (16), and $N_{k,k}(\mathbf{q}) := 0$.

Proof: Let us fix $k \in \{1, \ldots, \ell\}$. Given a field-stabilizing controller, Proposition 3.4 states that there exists θ, such that $\bar{Z}(\mathbf{q}, \theta) = 0$. It is clear that this must occur for some $\theta \in F(\mathbf{q})$. Therefore

$$\bar{Z}(\mathbf{q}, y_j) = 0 \quad \text{for some } j \in \{1, \ldots, \ell\}.$$

Let b be the smallest nonnegative integer, such that $\bar{Z}(\mathbf{q}, y_{k-b}) = 0$. By (18), we have

$$\bar{Z}(\mathbf{q}, y_k) = N_{k-b,k}(\mathbf{q}) \geq 0.$$

If $b = 0$, then the previous equation simply states that $Z(\mathbf{q}, y_k) = 0$. Now, if $N_{k-d,k}(\mathbf{q}) \leq N_{k-b,k}(\mathbf{q})$ for all $d \in \{0, \ldots, \ell\}$, then $\bar{Z}(\mathbf{q}, y_k) = \max_{b \in \{0, \ldots, \ell-1\}} N_{k-b,k}(\mathbf{q})$, and we have completed the proof.

Suppose by way of contradiction that there is $d \in \{0, \ldots, \ell - 1\}$ for which $N_{k-d,k}(\mathbf{q}) > N_{k-b,k}(\mathbf{q})$. From (17), we have

$$\bar{Z}(\mathbf{q}, y_k) \geq \bar{Z}(\mathbf{q}, y_{k-d}) + N_{k-d,k}(\mathbf{q}) \geq N_{k-d,k}(\mathbf{q})$$

where the second inequality comes from the fact that $\bar{Z}(\mathbf{q}, y_{k-d}) \geq 0$. However, $\bar{Z}(\mathbf{q}, y_k) = N_{k-b,k}(\mathbf{q})$, implying that $N_{k-d,k}(\mathbf{q}) \leq N_{k-b,k}(\mathbf{q})$, which is a contradiction. ■

The aforementioned lemma provides the value of the field in the steady state at each end point y_k. The field decreases from x_k to y_k (since these are the θ values over which the point $\mathbf{q}$ is covered) and, then, increases from y_k to x_{k+1}. Therefore, the maximum steady-state value is attained at an end point x_k for some k. For example, in Fig. 3(b), the maximum is attained at the point x_1. However, the value at x_k can be easily computed from the value at y_{k-1} using (13):

$$\bar{Z}(\mathbf{q}, x_{k+1}) = \max_{b \in \{0, \ldots, \ell-1\}} N_{k-b,k}(\mathbf{q}) + p(\mathbf{q}) \int_{y_k}^{x_{k+1}} \frac{d\theta}{v(\theta)}.$$

From this, we obtain the following result.

Lemma 3.7 (Steady-State Upper Bound): Given a field-stabilizing speed controller v, the maximum steady-state field at $\mathbf{q} \in Q$ [defined in (10)] satisfies

$$\mathcal{H}(\mathbf{q}, v) = \max_{\substack{k \in \{1, \ldots, \ell\} \\ b \in \{0, \ldots, \ell-1\}}} X_{k,b}(\mathbf{q})$$

where

$$X_{k,b}(\mathbf{q}) = p(\mathbf{q}) \int_{y_{k-b}}^{x_{k+1}} \frac{d\theta}{v(\theta)} - c(\mathbf{q}) \sum_{w=0}^{b-1} \int_{x_{k-w}}^{y_{k-w}} \frac{d\theta}{v(\theta)}$$

and $F(\mathbf{q}) = \cup_{k=1}^{\ell}[x_k, y_k]$ with $y_k > x_k > y_{k-1}$ for each k.

The aforementioned lemma provides a closed-form expression (albeit quite complex) for the largest steady-state value of the field. Thus, consider speed controllers of the form of (7), where $\beta_1, \ldots, \beta_n$ are basis functions (e.g., the rectangular basis). For a finite field, i.e., $Q = \{\mathbf{q}_1, \ldots, \mathbf{q}_m\}$, the terms $N_{k-b,k}(\mathbf{q}_i)$ can be written as

$$X_{k,b}(\mathbf{q}_i) = \sum_{j=1}^{n} \alpha_j X_{k,b}(\mathbf{q}_i, \beta_j)$$

where

$$X_{k,b}(\mathbf{q}_i, \beta_j) := p(\mathbf{q}) \int_{y_{k-b}}^{y_k} \beta_j(\theta) d\theta - c(\mathbf{q}) \sum_{w=0}^{b-1} \int_{x_{k-w}}^{y_{k-w}} \beta_j(\theta) d\theta. \tag{19}$$

With these definitions, we can define a LP for the minimization of the maximum of the steady-state field. We will write $\ell(\mathbf{q})$ to denote the number of disjoint intervals on the curve γ over which the point $\mathbf{q}$ is covered, as defined in (12).

Theorem 3.8 (Minimizing the Steady-State Field) Given a feasible persistent task, the solution to the following LP yields a

speed controller v of the form (7) that minimizes the maximum value of the steady-state field $\mathcal{H}(v)$:

$$\begin{aligned}
&\text{minimize} \quad B \\
&\text{subject to} \quad \sum_{j=1}^{n} \alpha_j X_{k,b}(\mathbf{q}_i, \beta_j) \le B \qquad \forall\, i \in \{1,\dots,m\} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad k \in \{1,\dots,\ell(\mathbf{q}_i)\} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad b \in \{0,\dots,\ell(\mathbf{q}_i)-1\} \\
&\qquad \sum_{j=1}^{n} \alpha_j K(\mathbf{q}_i, \beta_j) > 0 \qquad \forall\, i \in \{1,\dots,m\} \\
&\qquad \frac{1}{v_{\max}(j)} \le \alpha_j \le \frac{1}{v_{\min}(j)} \qquad \forall\, j \in \{1,\dots,n\}.
\end{aligned}$$

The optimization variables are α_j and B, and the quantities $X_{k,b}(\mathbf{q}_i, \beta_j)$ and $K(\mathbf{q}_i, \beta_j)$ are defined in (19) and (8).

From the aforementioned theorem, we can minimize the maximum value of the field using a LP. This optimization has $n+1$ variables (n basis function coefficients, and one upper bound B). The number of constraints is $m\sum_{i=1}^{m}\ell(\mathbf{q}_i)^2 + m + 2n$. In practice, $\ell(\mathbf{q}_i)$ is small compared with n and m, and is independent of n and m. Thus, for most instances, the LP has $O(2n+m)$ constraints.

IV. Multirobot Speed Controller

In this section, we turn to the multirobot case. We find that a field-stabilizing controller can again be formulated as the solution of a LP. The resulting multirobot controller does not rely on direct communication between the robots. We also show that the optimal controller (the one that minimizes the steady-state field) for multiple robots cannot be formulated as an LP, as in the single-robot case. Finding the optimal multirobot controller is a subject of ongoing work.

The multiple robots travel on fixed paths, which may be different and may intersect arbitrarily with one another. The robots may have different consumption rates, footprints, and speed limits. Collision avoidance for our controller is not dealt with in this section but has been addressed in [17].

We must first modify our notation to accommodate multiple robots. Consider N robots with closed paths, i.e., $\gamma_r : [0,1] \to \mathbb{R}^2$, $r \in \{1,\dots,N\}$, where γ_r and $\gamma_{r'}$ may intersect with each other arbitrarily (e.g., they may be the same path, share some segments, or be disjoint). We again assume that the parametrization of each curve is an arc-length parametrization, which is normalized to unity. Robot r, which traverses path γ_r at position $\theta_r(t)$, has a consumption rate $c_r(\mathbf{q})$ over a footprint $\mathcal{B}_r\big(\theta_r(t)\big)$ and has a speed controller $v_r(\theta_r)$ with the maximum and minimum speed constraints $v_{r,\min}(\theta_r)$ and $v_{r,\max}(\theta_r)$, respectively. Let $\mathcal{R}_r = (\mathcal{B}_r, v_{r,\min}, v_{r,\max})$ be the tuple containing the parameters for robot r and redefine $\mathcal{R} := (\mathcal{R}_1,\dots,\mathcal{R}_N)$ to be the tuple containing all the robots' tuples of parameters. In addition, the set of points θ_r from which $\mathbf{q}$ is in the footprint of robot r is denoted $F_r(\mathbf{q})$. Furthermore, we let $\gamma := (\gamma_1,\dots,\gamma_N)$ and $c := (c_1,\dots,c_N)$ be the tuple of all robot's paths and all robots' consumption functions, respectively. The persistent task with N robots is now written $(\mathcal{R}, \gamma, Q, p, c)$ as before, and we seek speed controllers $v_r(\theta_r)$ to keep $Z(\mathbf{q},t)$ bounded everywhere, as in Definition 2.2.

We make the assumption that when footprints of multiple robots are over the same point $\mathbf{q}$, their consumption rates are additive. Specifically, let $\mathcal{N}_{\mathbf{q}}(t)$ be the set of robots whose footprints are over the point $\mathbf{q}$ at time t, i.e., $\mathcal{N}_{\mathbf{q}}(t) := \{r \mid \mathbf{q} \in \mathcal{B}_r\big(\theta_r(t)\big)\}$. Then, the rate of change of the function $Z(\mathbf{q},t)$ is given by

$$\dot{Z}(\mathbf{q},t) = \begin{cases} p(\mathbf{q}) - \displaystyle\sum_{r\in\mathcal{N}_{\mathbf{q}}(t)} c_r(\mathbf{q}), & \text{if } Z(\mathbf{q},t) > 0 \\ \Big(p(\mathbf{q}) - \displaystyle\sum_{r\in\mathcal{N}_{\mathbf{q}}(t)} c_r(\mathbf{q})\Big)^+, & \text{if } Z(\mathbf{q},t) = 0. \end{cases} \tag{20}$$

We can reformulate a stability condition analogous to Lemma 2.5 to suit the multirobot setting, but we must first establish the notion of a common period for the speed controllers of all the robots. Let $T_r = \int_0^1 v_r^{-1}(\theta)d\theta$ be the period of robot r, and let $\tau_r(\mathbf{q}) = \int_{F_r(\mathbf{q})} v_r^{-1}(\theta)d\theta$ be the time in that period that $\mathbf{q}$ is in robot r's footprint. The existence of a common period rests on the following technical assumption.

Assumption 4.1 (Rational Periods): We assume that the periods T_r are rational numbers so that there exist integers num_r and den_r, such that $T_r = \text{num}_r/\text{den}_r$.

Assumption 4.1 implies that there exists a common period T, such that $T/T_r \in \mathbb{N}$ for all r. That is, each controller executes a whole number of cycles over the time interval T. Specifically, letting $T = \Pi_{r=1}^{N}\text{num}_r$, we have $T/T_r = \text{den}_r\Pi_{r'=1,r'\neq r}^{N}\text{num}_{r'}$. Now, we can state necessary and sufficient conditions for a field-stabilizing multirobot controller.

Lemma 4.2 (Multirobot Stability Condition): Given a multirobot persistent task, the set of controllers, i.e., $\theta_r \mapsto v_r(\theta_r)$, $r \in \{1,\dots,N\}$, is field stabilizing if and only if

$$\sum_{r=1}^{N} \frac{\tau_r(\mathbf{q})}{T_r} c_r(\mathbf{q}) > p(\mathbf{q}) \tag{21}$$

for every $\mathbf{q} \in Q$, where $T_r = \int_0^1 v_r^{-1}(\theta)d\theta$, and $\tau_r(\mathbf{q}) = \int_{F_r(\mathbf{q})} v_r^{-1}(\theta)d\theta$.

The lemma states an intuitive extension of Lemma 2.5, which is that the total consumption per cycle must exceed the total production per cycle at each point $\mathbf{q} \in Q$.

Proof: The proof closely follows the proof of Lemma 2.5. Consider the change of $Z(\mathbf{q},t)$ at a point $\mathbf{q}$ over any time interval T, where T is a common period of all the rational periods T_r so that $T/T_r \in \mathbb{N}$ for all r. By the integration of (20), we have that

$$\begin{aligned}
Z(\mathbf{q},t+T) - Z(\mathbf{q},t) &\ge Tp(\mathbf{q}) - \sum_{r=1}^{N}\int_t^T \mathbf{I}_r(\tau,\mathbf{q})c_r(\mathbf{q})d\tau \\
&= Tp(\mathbf{q}) - \sum_{r=\mathcal{N}_q}\sum_{k=1}^{T/T_r}\int_{t+(k-1)T_r}^{t+kT_r} \mathbf{I}_r(\tau,\mathbf{q})c_r(\mathbf{q})d\tau \\
&= T\Big(p(\mathbf{q}) - \sum_{r=1}^{N} c_r(\mathbf{q})\frac{\tau_r(\mathbf{q})}{T_r}\Big)
\end{aligned}$$

where $\mathbf{I}_r(t,\mathbf{q})$ takes the value 1 when $\mathbf{q}$ is in the footprint of robot r and 0 otherwise. To simplify notation, define $C(\mathbf{q}) := \sum_{r=1}^{N} c_r(\mathbf{q})$, and $\bar{C}(\mathbf{q}) := \sum_{r=1}^{N} c_r(\mathbf{q})\tau_r(\mathbf{q})/T_r$.

First, we prove necessity. In order to reach a contradiction, assume that the condition in Lemma 4.2 is false, but the persistent task is stable. Then, $T\big(p(\mathbf{q}) - \bar{C}(\mathbf{q})\big) \geq 0$ for some $\mathbf{q}$, implying $Z(\mathbf{q}, t+T) \geq Z(\mathbf{q},t)$ for some $\mathbf{q}$ and for all t, which contradicts stability. In particular, for an initial condition $Z(\mathbf{q},0) > Z_{\max}$, $Z(\mathbf{q},Tk) > Z_{\max}$ for all $k = 1,\ldots$.

Now, we prove sufficiency. If the condition is satisfied, there exists some $\epsilon > 0$, such that $T\big(p(\mathbf{q}) - \bar{C}(\mathbf{q})\big) = -\epsilon$. Suppose that at some time t and some point $\mathbf{q}$, $Z(\mathbf{q},t) > T\big(C(\mathbf{q}) - p(\mathbf{q})\big)$ (if no such time and point exists, the persistent task is stable). Then, for all times in the interval $\tau \in [t, t+T]$, $Z(\mathbf{q}, t+\tau) > 0$, and by (20), we have that $Z(\mathbf{q},t+T) - Z(\mathbf{q},t) = T\big(p(\mathbf{q}) - \bar{C}(\mathbf{q})\big) = -\epsilon$. Therefore, after finitely many periods T, $Z(\mathbf{q},t)$ will become less than $T\big(C(\mathbf{q}) - p(\mathbf{q})\big)$. Now, for a time t and a point $\mathbf{q}$, such that $Z(\mathbf{q},t) < T\big(C(\mathbf{q}) - p(\mathbf{q})\big)$, for all times $\tau \in [t, t+T]$, we have that $Z(\mathbf{q}, t+\tau) \leq Z(\mathbf{q},t) + p(\mathbf{q})T < TC(\mathbf{q})$. Therefore, once $Z(\mathbf{q},t)$ falls below $T\big(C(\mathbf{q}) - p(\mathbf{q})\big)$ (which will occur in finite time), it will never again exceed $TC(\mathbf{q})$. Therefore, the persistent task is stable with $Z_{\max} = \max_{\mathbf{q}\in Q} TC(\mathbf{q})$. ■

Remark 4.3 (Justification of Rational Periods): Assumption 4.1 is required only for the sake of simplifying the exposition. The rational numbers are a dense subset of the real numbers; therefore, for any $\epsilon > 0$, we can find num_r and den_r, such that $\text{num}_r/\text{den}_r \leq T_r \leq \text{num}_r/\text{den}_r + \epsilon$. One could carry ϵ through the analysis to prove our results in general.

A. Synthesis of Field-Stabilizing Multirobot Controllers

A field-stabilizing controller for the multirobot case can again be formulated as the solution of a LP, provided that we parametrize the controller using a finite number of parameters. However, our parametrization will differ from the single-robot case. Because there are multiple robots, each with its own period, we must normalize the speed controllers by their periods. We, then, represent the periods (actually, the inverse of the periods) as separate parameters to be optimized. This way, we maintain the independent periods of the robots, while still writing the optimization over multiple controllers as a single set of linear constraints.

Define a normalized speed controller, i.e., $\bar{v}_r(\theta_r) := T_r v_r(\theta_r)$, and the associated normalized coverage time, i.e., $\bar{\tau}_r(\mathbf{q}_i) := \int_{F_r(\mathbf{q}_i)} \bar{v}_r^{-1}(\theta) d\theta = \tau_r(\mathbf{q}_i)/T_r$. We parametrize $\bar{v}_i^{-1}$ as

$$\bar{v}_r^{-1}(\theta_r) = \sum_{j=1}^{n_r} \alpha_{rj}\beta_{rj}(\theta_r)$$

where n_r is the number of basis functions for the rth robot, and $\alpha_{rj} \in \mathbb{R}$ and $\beta_{rj} : [0,1] \to \mathbb{R}_{\geq 0}$ are jth parameter and basis function of robot r, respectively. It is useful to allow robots to have a different number of basis functions, since they may have paths of different lengths with different speed limits. By the assumption that the basis functions are normalized with $\int_0^1 \beta_{rj}(\theta) d\theta = 1$ for all r and j, we can enforce that $\bar{v}_r$ is a normalized speed controller by requiring

$$\sum_{j=1}^{n_r} \alpha_{rj} = 1 \quad \forall\, r \in \{1,\ldots,N\}.$$

As before, we could use any number of different basis functions, but here we specifically consider rectangular functions of the form (6). We also define the frequency for robot r as $f_r := 1/T_r$ and allow it to be a free parameter so that

$$v_r(\theta_r) = f_r \bar{v}_r(\theta_r) = \frac{f_r}{\sum_{j=1}^{n_r} \alpha_{rj}\beta_{rj}(\theta_r)}. \tag{22}$$

From (21), for the set of controllers to be field stabilizing, we require

$$\sum_{r=1}^{N} c_r(\mathbf{q}_i) \sum_{j=1}^{n_r} \alpha_{rj} \int_{F_r(\mathbf{q}_i)} \beta_{rj}(\theta) d\theta > p(\mathbf{q}_i)$$

for all $r \in \{1,\ldots,N\}$ and $\mathbf{q}_i \in Q$. Thus, defining

$$K_r(\mathbf{q}_i, \beta_j) := c_r(\mathbf{q}_i) \int_{F_r(\mathbf{q}_i)} \beta_{rj}(\theta) d\theta \tag{23}$$

the stability constraints become

$$\sum_{r=1}^{N} \sum_{j=1}^{n_r} \alpha_{rj} K_r(\mathbf{q}_i, \beta_j) > p(\mathbf{q}_i).$$

To satisfy the speed constraints, we also require that $v_{r,\max}^{-1}(\theta_r) \leq v_r^{-1}(\theta_r) \leq v_{r,\min}^{-1}(\theta_r)$, which, from (22), leads to

$$\frac{f_r}{v_{r,\max}(\theta_r)} \leq \sum_{j=1}^{n} \alpha_{rj}\beta_{rj}(\theta_r) \leq \frac{f_r}{v_{r,\min}(\theta_r)}$$

for all $\theta_r \in [0,1]$. For the rectangular basis functions in (6), this specializes to $f_r/v_{r,\min}(j) \leq \alpha_{rj} \leq f_r/v_{r,\min}(j)$ for all r and j, where

$$v_{r,\max}(j) := \inf_{\theta \in [(j-1)/n_r,\, j/n_r)} v_{r,\max}(\theta_r)$$

$$v_{r,\min}(j) := \sup_{\theta \in [(j-1)/n_r,\, j/n_r)} v_{r,\min}(\theta_r).$$

This gives a linear set of constraints for stability, which allows us to state the following theorem.

Theorem 4.4 (Field-Stabilizing Multirobot Controller): A persistent task is stabilizable by a set of multirobot speed controllers $v_r(\theta_r)$, $r \in \{1,\ldots,N\}$, of the form (22) if and only if the following LP is feasible:

$$\begin{aligned}
&\text{minimize} && 0 && \\
&\text{subject to} && \sum_{r=1}^{N}\sum_{j=1}^{n_r} \alpha_{rj} K_r(\mathbf{q}_i, \beta_j) > p(\mathbf{q}_i) && \\
& && && \forall\, i \in \{1,\ldots,m\} \\
& && \sum_{j=1}^{n_r} \alpha_{rj} = 1 && \forall\, r \in \{1,\ldots,N\} \\
& && f_r > 0 && \forall\, r \in \{1,\ldots,N\}
\end{aligned}$$

$$\frac{f_r}{v_{r,\min}(j)} \leq \alpha_{rj} \leq \frac{f_r}{v_{r,\min}(j)} \quad \forall\, j \in \{1,\ldots,n_r\} \quad r \in \{1,\ldots,N\}$$

where each α_{rj} and f_r is an optimization variable, and $K_r(\mathbf{q}_i, \beta_j)$ is defined in (23).

The above LP has $\sum_{r=1}^{N} n_r + N$ variables (one for each basis function coefficient α_{rj}, and one for each frequency f_r), and $m + \sum_{r=1}^{N} 2(n_r + 1)$ constraints. Thus, if we use n basis functions for each of the N robot speed controllers, then the number of variables is $(n+1)N$, and the number of constraints is $m + 2N(n+1)$. Therefore, the size of the LP grows linearly with the number of robots.

Remark 4.5 (Maximizing Stability Margin): As summarized in Corollary 3.2, rather than using the trivial cost function of 0 in the LP in Theorem 4.4, one may wish to optimize for a meaningful criterion. For example, the controller that gives the minimum number of common periods to the steady state can be obtained by maximizing B subject to $\sum_{r=1}^{N} \sum_{j=1}^{n_r} \alpha_{rj} K_r(\mathbf{q}_i, \beta_j) - p(\mathbf{q}_i) \geq B \,\forall\, i \in \{1,\ldots,m\}$, in addition to the other constraints of Theorem 4.4.

Remark 4.6 (Minimizing the Steady-State Field): Note that we do not find the speed controller that minimizes the steady-state field for the multirobot case, as was done for the single-robot case. The reason is that the quantities that are denoted as $N_{k-b,k}$ in the single-robot case would depend on the relative positions of the robots in the multiple-robot case. These quantities would have to be enumerated for all possible relative positions between the multiple robots, and there are infinite such relative positions. Thus, the problem cannot be posed as an LP the same way as the single-robot case. We are currently investigating alternative methods for finding the optimal multirobot controller, such as convex optimization.

Remark 4.7 (Decentralization and Robot Additions and Deletions): The execution of our multirobot controller is decentralized in the most extreme sense, because each robot's controller requires no communication with any other robot. However, the design of the multirobot controller using our LP formulation is centralized. For example, a new LP would have to be solved centrally to obtain new controllers, whenever robots join or leave the group, or when the environment changes. However, this is not necessarily a hinderance. LPs with thousands of constraints can be solved in seconds on modern processors; therefore, it is reasonable to expect a single "leader" robot to recompute the controllers for all robots and broadcast them to the group. In addition, it may be possible to decentralize the LP in Theorem 4.4 to be implemented efficiently across the group of robots. This is an area of the ongoing research.

V. Simulations

In this section, we present simulation results for the single-robot and multirobot controllers. The purpose of this section is threefold: 1) to discuss details that are needed to implement the speed control optimizations, 2) to demonstrate how controllers can be computed for both discrete and continuous fields, and 3) to explore robustness to modeling errors, parameter uncertainty, robot tracking errors, and stochastic field evolution.

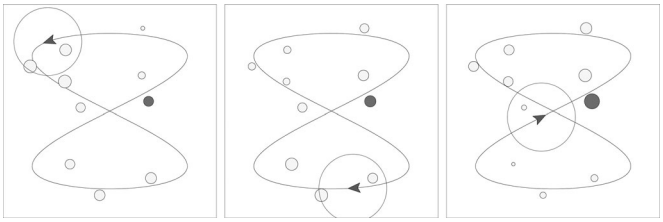

Fig. 4. Example with a field consisting of 10 points. The field $Z(\mathbf{q}, t)$ at each point is indicated by the area of the disk centered at the point. The vehicle footprint is a disk. The time sequence of the three snapshots goes from left to right. The vehicle is moving counter-clockwise around the top half of the figure eight and clockwise around the bottom half. The time evolution of the dark (red) field point is shown in Fig. 5.

The optimization framework was implemented in MATLAB, and the LPs were solved using the freely available SeDuMi (self-dual-minimization) toolbox. To give the readers some feel for the efficiency of the approach, we report the time to solve each optimization on a laptop computer with a 2.66 GHz dual core processor and 4 GB of RAM. The simulations are performed by discretizing time and, thus, converting the field evolution into a discrete-time evolution. To perform the optimization, we need a routine for computing the set $F(\mathbf{q})$ for each field point $\mathbf{q} \in Q$. This is done as follows. We initialize a set $F(\mathbf{q})$ for each point $\mathbf{q}$ and discretize the robot path into a finite set $\{\theta_1,\ldots,\theta_n\}$. For the rectangular basis, this discretization is naturally given by the set of basis functions. We iteratively place the robot footprint at each point θ_i, which is oriented with the desired robot heading at that point on the curve and, then, add θ_i to each set $F(\mathbf{q})$ for which $\mathbf{q}$ is covered by the footprint. By the approximation of the robot footprint with a polygon, we can determine if a point lies in the footprint efficiently. (This is a standard problem in computational geometry.)

In Fig. 4, we show a simulation of a ground robot performing a persistent monitoring task of 10 points (i.e., $|Q| = 10$). The environment is 70 m $\times$ 70 m, and the closed path has a length of 300 m. For all points, we set the consumption rate, i.e., $c(\mathbf{q}) = 1$ (in units of$(\text{fieldheight})/\text{s}$). For each light (yellow) point $\mathbf{q}$ we set $p(\mathbf{q}) = 0.15$, and for the single red point, we set $p(\mathbf{q}) = 0.35$. The robot has a circular footprint with a radius of 12 m, and for all θ, the robot has a minimum speed of $v_{\min} = 0.2$ m/s and a maximum speed of $v_{\max} = 2$ m/s. If the robot were to simply move at a constant speed along the path, then 8 of the 10 field points would be unstable. The speed controller that results from the optimization in Section III-B is shown in Fig. 5(a). A total of 150 rectangular basis functions were used. The optimization was solved in less than 1/10 of a second. Using the speed controller, the cycle time was $T = 420$ s.

The field $Z(\mathbf{q}, t)$ for the red (shaded) point in Fig. 4 is shown as a function of time in Fig. 5(b). One can see that the field converges in finite time to a periodic cycle. In addition, the field goes to zero during each cycle. The periodic cycle is the steady state as characterized in Section III-B.

In Fig. 5(c), we compare the performance of the optimized controller of Fig. 5(a) with a constant speed controller and a

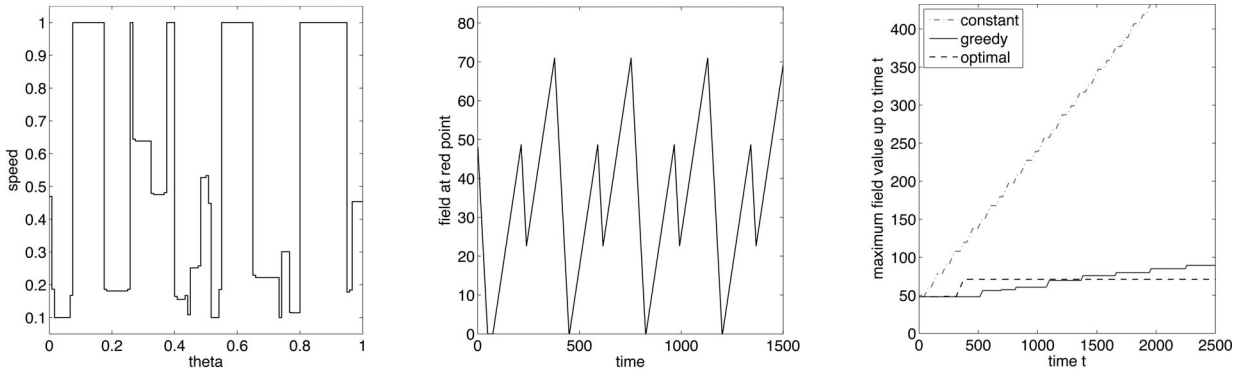

Fig. 5. Speed controller that minimizes the maximum field value for the setup that is shown in Fig. 4. The minimum speed is $v_{\min} = 0.22$ m/s, and the maximum speed is $v_{\max} = 2$ m/s. The middle figure shows the field $Z(\mathbf{q}, t)$ for the red (dark) point in Fig. 4. The right figure shows the maximum field value over all points in the interval $[0, t]$ as a function of t for the optimized controller and compares it with a constant speed and a greedy controller. (a) Optimal speed controller. (b) Field $Z(\mathbf{q}, t)$ at the red (dark) point. (c) Comparison of three controllers.

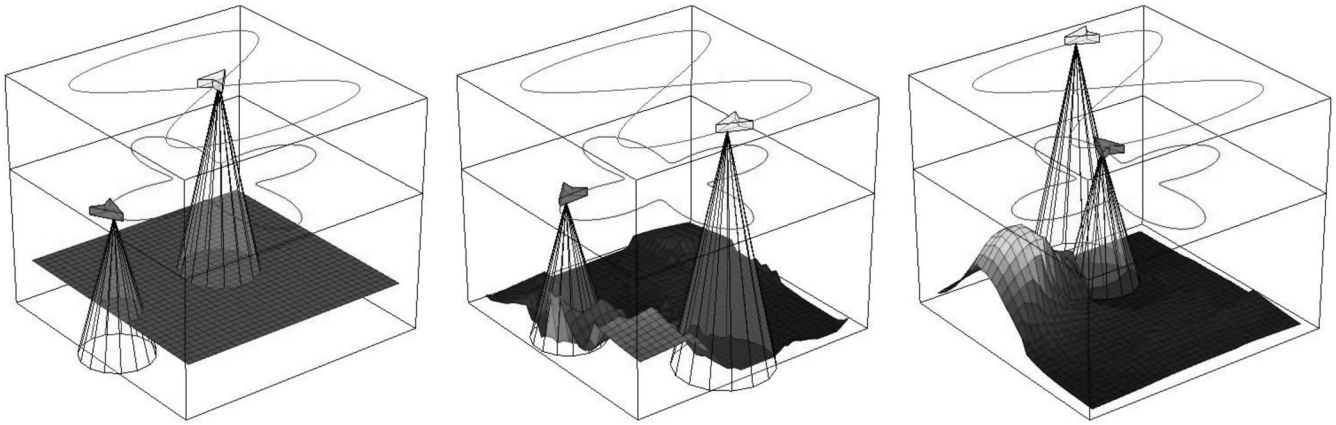

Fig. 6. Example of using a discretized approximation of a continuous field for two robots. The grid is 32×32, and the field $Z(\mathbf{q}, t)$ at each point is shown as the surface. The footprint of the higher robot (yellow) is 4/3 of the radius of the footprint of the lower (red) robot. The snapshot sequence goes from left to right with the left snapshot showing the initial condition of the robots and field.

greedy controller. The plot shows the maximum value of the field to time t as a function of t. If the field is stable under a controller, then its corresponding curve should become constant. For constant speed controller, in which the robot moves at the maximum speed around the curve, we see that the field is highly unstable. In the greedy controller, the robot moves at a speed proportional to $Z_{\max}(t) - Z_{\max,\text{ftprnt}}(t)$, where $Z_{\max}(t)$ is the maximum field value at time t, and $Z_{\max,\text{ftprnt}}(t)$ is the maximum field value in the robot's footprint at time t. Thus, the robot moves at its minimum speed, when its footprint is covering the point with the maximum field value, and moves at a maximum speed, when it is covering points with low field values. This controller performs much better, but the field is still unstable.

In Fig. 6, a simulation is shown for the case, when the entire continuous environment must be monitored by two aerial robots. For multiple robots, we synthesized a field-stabilizing controller by the maximization of the stability margin, as discussed in Remark 4.5. The continuous field is defined over a 690 m $\times$ 690 m square and was approximated using a 32×32 grid. The error that is introduced in such a discretization can be rigorously characterized as a function of the spatial variation of $p(\mathbf{q})$ and $c(\mathbf{q})$ and the robot footprint. Because of space limitations, we omit a more detailed discussion. For all points $\mathbf{q} \in Q$, we set the consumption rate, i.e., $c(\mathbf{q}) = 1$, and the production function is shown in Fig. 7. Robot 1 followed a figure-eight path, which has a length of 2630 m, while robot 2 followed a four-leaf clover path with a length of 2250 m. The footprint for robot 1, the higher (yellow)

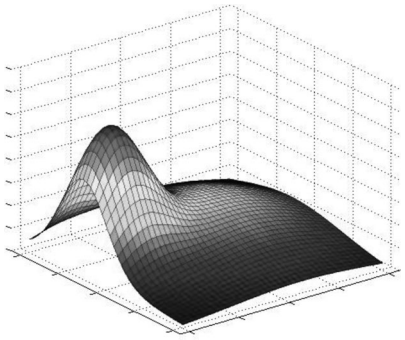

Fig. 7. Production function $p(\mathbf{q})$ for the simulations in Figs. 6 and 8.

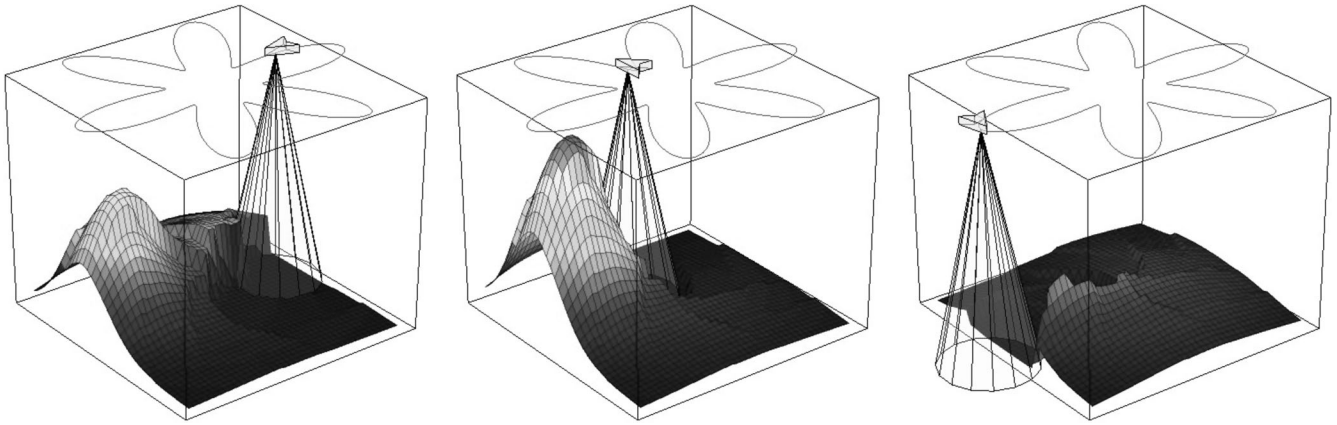

Fig. 8. Example of using a discretized approximation of a continuous field. The field $Z(\mathbf{q}, t)$ at each point is shown as the surface. The footprint of the vehicle is a disk, and the robot's trajectory is given by the "six-leaf" clover. The time sequence of the three snapshots goes from left to right. In the rightmost snapshot, the vehicle has just finished reducing the large peak that forms in the left and center snapshots.

robot had a radius of 100 m, and the speed constraints were given by $v_{\min,1} = 1.5$ m/s and $v_{\max,1} = 15$ m/s. The footprint for robot 2, the lower (red) robot, had a radius of 133 m, and the speed constraints were given by $v_{\min,2} = 2$ m/s and $v_{\max,2} = 20$ m/s. The cycle time for robot 1 was $T_1 = 519$ s, and the cycle time for robot 2 was $T_2 = 443$ s. A total of 150 rectangular basis functions were used for each robot's speed controller. The optimization was solved in approximately 10 s. A snapshot for a simulation with three robots is shown in Fig. 1. In this case, the green robot flying at a higher altitude has a square footprint, which is oriented with the robot's current heading.

A. Case Study in Robustness

In this section, we demonstrate how we can compute a speed controller that exhibits robustness to motion errors, modeling uncertainty, or stochastic fluctuations. This is important since the speed controller is computed offline. It should be noted, however, that, in practice, the controller can be recomputed periodically during the robot's execution, taking into account new information about the field evolution.

As shown in Corollary 3.2, we can maximize stability margin of a speed controller. The corollary showed that in the maximization of this metric, we obtain some robustness to error. To explore the robustness properties of this controller, consider the single-robot example in Fig. 8. In this example, the square 665 m $\times$ 665 m environment must be monitored by an aerial robot. We approximated the continuous field using a 32 $\times$ 32 grid. The consumption rate was set to $c(\mathbf{q}) = 5$, for each field point $\mathbf{q} \in Q$. The production rate of the field was given by the function that is shown in Fig. 7. The maximum production rate of the field was 0.74 and the average was 0.21. The robot had a circular footprint with a radius of 133 m, a minimum speed of $v_{\min} = 1.5$ m/s, and a maximum speed of $v_{\max} = 15$ m/s. The path had a cycle length of 4200 m. If the robot followed the path at constant speed, then 80 of the points would be unstable.

For the speed controller, we used 280 rectangular basis functions and solved the optimization as described in Corollary 3.2, resulting in a stability margin of $B = 97.8$. The optimization was solved in approximately 10 s. The time for the robot to complete one cycle of the path using this controller was $T = 439$ s.

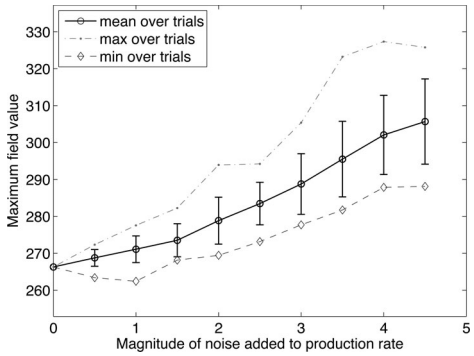

Fig. 9. Robustness of the speed controller to noise in the production rate. For each value of noise $n_{\max}$, the plot shows statistics on the maximum value that is reached by the field over 20 independent trials.

Stochastic Field Evolution: Now, suppose that we add zero-mean noise to the production function. Thus, the robot based its speed controller on the "nominal" production function $\bar{p}(\mathbf{q})$ (shown in Fig. 7), but the actual production function is $p(\mathbf{q}, t) = \bar{p}(\mathbf{q}) + n(t, \mathbf{q})$, where $n(t, \mathbf{q})$ is noise. For the simulation, at each time instant t, and for each point $\mathbf{q} \in Q$, we independently draw $n(t, \mathbf{q})$ uniformly from the set $[-n_{\max}, n_{\max}]$, where $n_{\max} > 0$ is the maximum value of the noise.

The simulations were carried out as follows. We varied the magnitude of the noise $n_{\max}$ and studied the maximum value that is reached by the field. For each value $n_{\max}$, we performed 20 trials, and in each trial, we recorded the maximum value that is reached by the field on a time horizon of 2500 s. In Fig. 9, we display statistics from the 20 independent trials at each noise level, namely the mean, minimum, and maximum, as well as the standard deviation. With zero added noise, one can see that all, i.e., mean, minimum, and maximum, coincide. As noise is added to the evolution, the difference between the minimum and the maximum value grows. However, it is interesting to note that while the performance degrades (i.e., the mean increases with increasing noise), the system remains stable. Thus, the

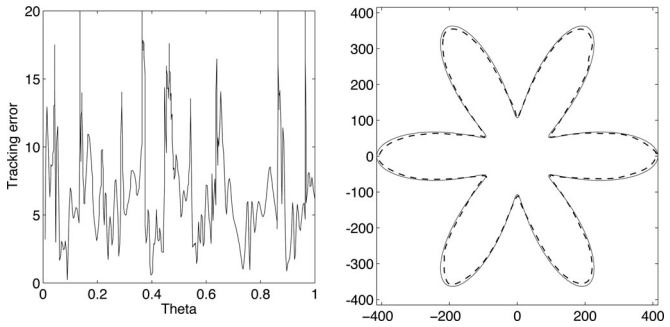

Fig. 10. Robustness of the speed controller to tracking errors. (*Left*) Absolute tracking error in meters. (*Right*) Solid line is the desired path, and the dashed line is the path that is executed by the unicycle robot.

simulation demonstrates some of the robustness properties of the proposed controller.

Parameter Errors: We can also consider robustness to parameter errors. In particular, consider the case where the robot bases its optimization on a production rate of $\bar{p}(\mathbf{q})$ (shown in Fig. 7), but the actual production rate is given by $p(\mathbf{q}) = \bar{p}(\mathbf{q}) + \epsilon$, where $\epsilon > 0$ (note that the field is trivially stable for any $\epsilon \leq 0$). By the maximization of the stability margin, we obtain some level of robustness against this type of parameter uncertainty. The amount of error that we can tolerate is directly related to the stability margin $B = 97.8$, as shown in Corollary 3.2. In particular, we obtain $\epsilon < Bc(\mathbf{q}_i)/\left(\sum_{j=1}^{n} \alpha_j \int_0^1 \beta_j(\theta) d\theta\right) = 0.074$. We performed simulations of monitoring task for successively larger values of ϵ. From these data, we verified that the field remains stable for any $\epsilon \leq 0.07$. For this example, the average value of $p(\mathbf{q})$ over all points $\mathbf{q}$ is 0.21, and therefore, we can handle the uncertainty in the magnitude of the production rate on the order of 30%.

Tracking Error: After running the speed optimization, a robot has a desired trajectory, which consists of the prespecified path and the optimized speed along the path. In practice, this trajectory will be the input to a tracking controller, which takes into account the robot dynamics. Since there are inevitably tracking errors, the stability margin of the controller is needed to ensure field stability. As an example, we considered a unicycle model for an aerial robot. In this model, the robot's configuration is given by a heading ϕ and a position (x, y). The control inputs are the linear and angular speeds $\dot{x} = v \cos\phi$, $\dot{y} = v \sin\phi$, and $\dot{\phi} = \omega$. The linear speed had bounds of $v_{\min} = 1.5$ m/s and $v_{\max} = 15$ m/s, and the angular speed was upper bounded by 0.5 rad/s. We used the same speed controller as in the previous two examples (maximizing the stability margin). For trajectory tracking, we used a dynamic feedback linearization controller [42]. We chose conservative controller gains of 0.5 for the proportional and derivative control in order to accentuate the tracking error. The results are shown in Fig. 10. Because of the stability margin of 97.8, the field remains stable, even in the presence of this tracking error. However, in simulation, the maximum field height increased by about 13% from 268 (as shown for the zero-noise case in Fig. 9) to 305.

VI. Conclusion, Extensions, and Open Problems

A. Conclusion

In this paper, a model for persistent sweeping and monitoring tasks has been proposed, and controllers for robots to accomplish those tasks have been derived. Specifically, the case in which robots are confined to prespecified, closed paths, along which their speed must be controlled, has been considered. An LP has been formulated, whose solution provides speed controllers that keep the accumulation bounded everywhere in the environment for single robots and multiple robots. For single robots, a different LP has been formulated to give the optimal controller—the one that keeps the accumulation function as low as possible everywhere. We see this as a solution to one kind of persistent task for robots, but many open problems remain.

B. Extensions and Open Problems

We are interested in the general problem of solving persistent tasks, which we broadly define as tasks that require perpetual attention by a robotic agent. The main objective of a persistent task is to maintain the accumulation of some undesirable quantity at as low a value as possible over an environment using one or multiple robotic agents. The difficulty of this problem depends on what is known by the robots and precisely what the robots' capabilities are. Let us enumerate several possible dimensions for extension on this problem.

1) *Trajectory versus path versus speed:* One might consider controlling only the speed over a prescribed path, as we have done in this paper, only the path with a prescribed speed, or the complete trajectory planning.
2) *Single versus multiple robots:* There may be only one robot, or there may be a team of robots, potentially with heterogeneous capabilities and constraints.
3) *Known versus unknown production rate:* The robot may know (or be able to sense) the production rate, or it may have to learn it over time from measurements.
4) *Constant versus time-varying production rate:* The production rate may be constant in time, or it may change, indicating a change in the environment's dynamics.
5) *Finite versus continuum points of interest:* The points of interest in the environment may be viewed as a finite set of discrete points over which the accumulation is controlled or as an infinite continuum in which we would like to control the accumulation at every point.

In this paper, we specifically considered *speed* control over a prescribed path of both *single and multiple* robots in a *finite* environment with a *known*, *constant* production rate.

One interesting direction, to expand this study, is to consider planning full trajectories for robots to carry out persistent tasks. The high dimensionality of the space of possible trajectories makes this a difficult problem. However, if the robot's path is limited by some inherent constraints, then this problem may admit solutions with guarantees. For example, underwater gliders are commonly constrained to take piecewise linear paths, which can be parametrized with a low number of parameters. Another direction of extension is to have a robot solve the LP for its

controller online. This would be useful if, e.g., the production rate is not known before hand, but can be sensed over the sensor footprint of the robot. Likewise if the production rate changes in time, it would be useful for a robot to be able to adjust its controller online to accommodate these changes. A promising approach for this is to repeatedly solve for the LP in a receding horizon fashion, using newly acquired information to update the model of the field evolution. We continue to study problems of persistent tasks for robots in these and other directions.

APPENDIX

PERIODIC POSITION-FEEDBACK CONTROLLERS

Consider a general speed controller $(Z, \mathrm{IC}, t) \mapsto v(\theta, Z, \mathrm{IC}, t)$, where $\mathrm{IC} := (Z(\mathbf{q}, 0), \theta(0))$ is the set of initial conditions. Since $v_{\min}(\theta) > 0$ for all $\theta \in [0, 1]$, the value of θ strictly monotonically increases from 0 to 1 for every valid controller (once it reaches 1, it then wraps around to 0). In addition, the evolution of Z is deterministic and is uniquely determined by the initial conditions and the speed controller, as given in (1). Because of this, every controller of the form $v(\theta, Z, \mathrm{IC}, t)$ can be written as an infinite sequence of controllers $v_1(\theta, \mathrm{IC}), v_2(\theta, \mathrm{IC}), \ldots$, where controller $v_k(\theta, \mathrm{IC})$ is the controller that is used during the kth period (or cycle).

With the discussion made earlier in place, we are now ready to prove Proposition 2.4.

Proof of Proposition 2.4: To begin, consider a feasible persistent monitoring task and a field-stabilizing controller of the form $v(\theta, Z, \mathrm{IC}, t)$, where $\mathrm{IC} := (Z(\mathbf{q}, 0), \theta(0))$. Without loss of generality, we can assume that $\theta(0) := 0$. From the discussion made earlier, we can write the general controller as a sequence of controllers $v_1(\theta, \mathrm{IC}), v_2(\theta, \mathrm{IC}), \ldots$, where controller $v_k(\theta, \mathrm{IC})$, $k \in \mathbb{N}$, is used on the kth period (or cycle).

Since the controller is stable, there exists a $Z_{\max}$, such that for every set of initial conditions IC, we have $\limsup_{t \to +\infty} Z(\mathbf{q}, t) \leq Z_{\max}$. Let us fix $\epsilon > 0$ and fix the initial conditions to a set of values $\overline{\mathrm{IC}}$, such that $Z(\mathbf{q}, 0) > Z_{\max} + \epsilon$ for all $\mathbf{q} \in Q$. Now, we will prove the result by the construction of a periodic position-feedback controller.

Let $t_1 = 0$, and define

$$t_k = t_{k-1} + \int_0^1 \frac{1}{v_k(\theta, \overline{\mathrm{IC}})} d\theta$$

for each integer $k \geq 2$. Thus, controller v_k is used during the time interval $[t_k, t_{k+1}]$. Following the same argument as in the proof of Lemma 2.5, we have that for each $\mathbf{q} \in Q$,

$$Z(\mathbf{q}, t_k) - Z(\mathbf{q}, t_1) \geq p(\mathbf{q}) \sum_{\ell=1}^{k} \int_0^1 \frac{1}{v_\ell(\theta, \overline{\mathrm{IC}})} d\theta - c(\mathbf{q}) \sum_{\ell=1}^{k} \int_{F(\mathbf{q})} \frac{1}{v_\ell(\theta, \overline{\mathrm{IC}})} d\theta.$$

Now, $Z(\mathbf{q}, t_1) > Z_{\max} + \epsilon$ is the initial condition, and $\limsup_{t \to +\infty} Z(\mathbf{q}, t) \leq Z_{\max}$. Thus, for every fixed $\delta \in (0, \epsilon)$, there exists a finite k, such that $Z(\mathbf{q}, t_k) \leq Z_{\max} + \delta$. Since this must hold for every $\mathbf{q} \in Q$, we see that there exists an integer k, such that

$$p(\mathbf{q}) \sum_{\ell=1}^{k} \int_0^1 \frac{1}{v_\ell(\theta, \overline{\mathrm{IC}})} d\theta - c(\mathbf{q}) \sum_{\ell=1}^{k} \int_{F(\mathbf{q})} \frac{1}{v_\ell(\theta, \overline{\mathrm{IC}})} d\theta < 0$$

for every $\mathbf{q} \in Q$. By the rearrangement of the previous equation, we obtain

$$c(\mathbf{q}) \int_{F(\mathbf{q})} \sum_{\ell=1}^{k} \frac{1}{v_\ell(\theta, \overline{\mathrm{IC}})} d\theta > p(\mathbf{q}) \int_0^1 \sum_{\ell=1}^{k} \frac{1}{v_\ell(\theta, \overline{\mathrm{IC}})} d\theta. \tag{24}$$

Therefore, let us define the periodic controller

$$v(\theta) = k \left(\sum_{\ell=1}^{k} \frac{1}{v_\ell(\theta, \overline{\mathrm{IC}})} \right)^{-1}$$

for each $\theta \in [0, 1]$. Note that if $v_{\min}(\theta) \leq v_\ell(\theta, \overline{\mathrm{IC}}) \leq v_{\max}(\theta)$ for all $\theta \in [0, 1]$ and all $\ell \in \mathbb{N}$, then $v_{\min}(\theta) \leq v(\theta) \leq v_{\max}(\theta)$. However, combining the definition of $v(\theta)$ with (24), we immediately observe that $v(\theta)$ satisfies the stability condition in Lemma 2.5, and thus, $v(\theta)$ is a field-stabilizing position-feedback controller. ■

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their comments and suggestions. This study was performed primarily at the Massachusetts Institute of Technology, Cambridge.

REFERENCES

[1] S. L. Smith, M. Schwager, and D. Rus, "Persistent monitoring of dynamic environments using a robot with limited range sensing," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011, pp. 5448–5455.

[2] N. M. P. Kakalis and Y. Ventikos, "Robotic swarm concept for efficient oil spill confrontation," *J. Hazardous Mater.*, vol. 154, pp. 880–887, 2008.

[3] D. MacKenzie and T. Balch, "Making a clean sweep: Behavior-based vacuuming," presented at the AAAI Fall Symp., Instantiating Real-world Agents, Raleigh, NC, Mar. 1993.

[4] N. Correll, N. Arechiga, A. Bolger, M. Bollini, B. Charrow, A. Clayton, F. Dominguez, K. Donahue, S. Dyar, L. Johnson, H. Liu, A. Patrikalakis, T. Robertson, J. Smith, D. Soltero, M. Tanner, L. White, and D. Rus, "Building a distributed robot garden," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, St. Louis, MO, 2009, pp. 1509–1516.

[5] R. N. Smith, M. Schwager, S. L. Smith, B. H. Jones, D. Rus, and G. S. Sukhatme, "Persistent ocean monitoring with underwater gliders: Adapting sampling resolution," *J. Field Robot.*, vol. 28, no. 5, pp. 714–741, 2011.

[6] M. Dunbabin, J. Roberts, K. Usher, and P. Corke, "A new robot for environmental monitoring on the Great Barrier Reef," presented at the Australasian Conf. Robot. Autom., Australian National Univ., Canberra, Australia, Dec. 2004.

[7] S. Srinivasan, H. Latchman, J. Shea, T. Wong, and J. McNair, "Airborne traffic surveillance systems: Video surveillance of highway traffic," in *Proc. Int. Workshop Video Surveillance Sens. Netw.*, New York, Oct.2004, pp. 131–135.

[8] E. M. Arkin, S. P. Fekete, and J. S. B. Mitchell, "Approximation algorithms for lawn mowing and milling," *Comput. Geometry: Theory Appl.*, vol. 17, no. 1–2, pp. 25–50, 2000.

[9] K. Kant and S. W. Zucker, "Toward efficient trajectory planning: The path-velocity decomposition," *Int. J. Robot. Res.*, vol. 5, no. 3, pp. 72–89, 1986.

[10] J. Peng and S. Akella, "Coordinating multiple robots with kinodynamic constraints along specified paths," *Int. J. Robot. Res.*, vol. 24, no. 4, pp. 295–310, 2005.

[11] S. L. Smith and D. Rus, "Multirobot monitoring in dynamic environments with guaranteed currency of observations," in *Proc. IEEE Conf. Decision Contr.*, Atlanta, GA, Dec.2010, pp. 514–521.
[12] D. G. Luenberger, *Linear and Nonlinear Programming*, 2nd ed. Reading, MA: Addison-Wesley, 1984.
[13] E. W. Cheney, *Introduction to Approximation Theory*, 2nd ed. New York: AMS Chelsea, 2000.
[14] E. J. Candes and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.
[15] R. Sanner and J. Slotine, "Gaussian networks for direct adaptive control," *IEEE Trans. Neural Netw.*, vol. 3, no. 6, pp. 837–863, Nov. 1992.
[16] T. Poggio and S. Smale, "The mathematics of learning: Dealing with data," *Notices Amer. Math. Soc.*, vol. 50, no. 5, pp. 537–544, 2003.
[17] D. E. Soltero, S. L. Smith, and D. Rus, "Collision avoidance in trajectory tracking for multirobot persistent tasks," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, San Francisco, CA, Sep. 2011, pp. 3645–3652.
[18] L. S. Gandin, *Objective Analysis of Meteorological Fields*. Jerusalem, Israel: Israeli Program for Scientific Translations, 1966 (originally published in Russian in 1963 by Gidrometeor, Leningrad, USSR).
[19] N. Cressie, "The origins of Kriging," *Math. Geol.*, vol. 22, no. 3, pp. 239–252, 1990.
[20] B. Grocholsky, "Information-theoretic control of multiple sensor platforms," Ph.D. dissertation, Dept. Aerospace, Mechatronic, Mech. Eng., Univ. Sydney, Sydney, Australia, 2002.
[21] K. M. Lynch, I. B. Schwartz, P. Yang, and R. A. Freeman, "Decentralized environmental modeling by mobile sensor networks," *IEEE Trans. Robot.*, vol. 24, no. 3, pp. 710–724, Jun. 2008.
[22] P. Yang, R. A. Freeman, and K. M. Lynch, "Mulit-agent coordination by decentralized estimation and control," *IEEE Trans. Automat. Contr.*, vol. 53, no. 11, pp. 2480–2496, Dec. 2008.
[23] J. Cortés, "Distributed kriged Kalman filter for spatial estimation," *IEEE Trans. Automat. Contr.*, vol. 54, no. 12, pp. 2816–2827, Dec. 2009.
[24] F. Zhang and N. E. Leonard, "Cooperative filters and control for cooperative exploration," *IEEE Trans. Automat. Contr.*, vol. 55, no. 3, pp. 650–663, Mar. 2010.
[25] R. Graham and J. Cortés, "Adaptive information collection by robotic sensor networks for spatial estimation," *IEEE Trans. Automat. Contr.*, to be published.
[26] J. Le Ny and G. J. Pappas, "On trajectory optimization for active sensing in Gaussian process models," in *Proc. IEEE Conf. Decision Contr.*, Shanghai, China, Dec. 2009, pp. 6282–6292.
[27] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte, "Information based adaptive robotic exploration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Lausanne, Switzerland, Oct. 2002, pp. 540–545.
[28] I. Rekleitis, V. Lee-Shue, A. P. New, and H. Choset, "Limited communication multirobot team based coverage," in *Proc. IEEE Int. Conf. Robot. Autom.*, New Orleans, LA, Apr. 2004, pp. 3462–3468.
[29] S. Koenig, B. Szymanski, and Y. Liu, "Efficient and inefficient ant coverage methods," *Ann. Math. Artif. Intell.*, vol. 31, pp. 41–76, 2001.
[30] H. Choset, "Coverage for robotics: A survey of recent results," *Ann. Math. Artif. Intell.*, vol. 31, no. 1–4, pp. 113–126, 2001.
[31] B. Bethke, J. P. How, and J. Vian, "Group health management of UAV teams with applications to persistent surveillance," in *Proc. Amer. Contr. Conf.*, Seattle, WA, Jun. 2008, pp. 3145–3150.
[32] B. Bethke, J. Redding, J. P. How, M. A. Vavrina, and J. Vian, "Agent capability in persistent mission planning using approximate dynamic programming," in *Proc. Amer. Contr. Conf.*, Baltimore, MD, Jun. 2010, pp. 1623–1628.
[33] Y. Chevaleyre, "Theoretical analysis of the multiagent patrolling problem," in *Proc. IEEE/WIC/ACM Int. Conf. Intell. Agent Technol.*, Beijing, China, Sep. 2004, pp. 302–308.
[34] Y. Elmaliach, N. Agmon, and G. A. Kaminka, "Multirobot area patrol under frequency constraints," in *Proc. IEEE Int. Conf. Robot. Autom.*, Roma, Italy, Apr. 2007, pp. 385–390.
[35] N. Nigram and I. Kroo, "Persistent surveillance using multiple unmanned air vehicles," in *Proc. IEEE Aerosp. Conf.*, Big Sky, MT, May 2008, pp. 1–14.
[36] D. B. Kingston, R. W. Beard, and R. S. Holt, "Decentralized perimeter surveillance using a team of UAVs," *IEEE Trans. Robot.*, vol. 24, no. 6, pp. 1394–1404, Dec. 2008.
[37] P. F. Hokayem, D. Stipanović, and M. W. Spong, "On persistent coverage control," in *Proc. IEEE Conf. Decision Contr.*, New Orleans, LA, Dec. 2007, pp. 6130–6135.
[38] L. Kleinrock, *Queueing Systems: Theory*. vol. 1, New York: Wiley, 1975.
[39] D. J. Bertsimas and G. J. van Ryzin, "Stochastic and dynamic vehicle routing in the Euclidean plane with multiple capacitated vehicles," *Oper. Res.*, vol. 41, no. 1, pp. 60–76, 1993.
[40] S. L. Smith, M. Pavone, F. Bullo, and E. Frazzoli, "Dynamic vehicle routing with priority classes of stochastic demands," *SIAM J. Contr. Optim.*, vol. 48, no. 5, pp. 3224–3245, 2010.
[41] M. Ahmadi and P. Stone, "A multirobot system for continuous area sweeping tasks," in *Proc. IEEE Int. Conf. Robot. Autom.*, Orlando, FL, May 2006, pp. 1724–1729.
[42] G. Oriolo, A. D. Luca, and M. Vendittelli, "WMR control via dynamic feedback linearization: Design, implementation, and experimental validation," *IEEE Trans. Contr. Syst. Technol.*, vol. 10, no. 6, pp. 835–852, Nov. 2002.

Stephen L. Smith (S'05–M'09) received the B.Sc. degree from Queen's University, Kingston, ON, Canada, in 2003, the M.A.Sc. degree from the University of Toronto, Toronto, ON, in 2005, and the Ph.D. degree from the University of California, Santa Barbara, in 2009.

From 2009 to 2011, he was a Postdoctoral Associate with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON. His main research interest includes the control of autonomous systems, with a particular emphasis on robotic motion planning and distributed coordination.

Mac Schwager (S'04–M'10) received the B.S. degree the Stanford University, Stanford, CA, in 2000 and the M.S. and Ph.D. degrees from the Massachusetts Institute of Technology (MIT), Cambridge, in 2005 and 2009, respectively.

From 2010 and 2011, he worked as a Postdoctoral Researcher jointly with the General Robotics, Automation, Sensing, and Perception Laboratory, University of Pennsylvania, Philadelphia, and the Computer Science and Artificial Intelligence Laboratory at MIT. He is currently an Assistant Professor with the Department of Mechanical Engineering, Boston University, Boston MA. His research interests include distributed algorithms for control, estimation, and model learning in groups of robots and animals.

Daniela Rus (F'10) received the Ph.D. in computer science from Cornell University Ithaca, NY, in 1992.

She is currently a Professor with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, where she co-directs the Center for Robotics, Computer Science, and Artificial Intelligence Laboratory and holds the Singapore Research Professor of Autonomous Systems Chair. She was a Professor with the Department of Computer Science, Dartmouth College, Hanover, NH, where she founded and directed two laboratories in robotics and mobile computing. Her research interests include distributed robotics and mobile computing, and her application focus includes transportation, security, environmental modeling and monitoring, underwater exploration, and agriculture.

Dr. Rus received the National Science Foundation Career Award. She is an Alfred P. Sloan Foundation Fellow. She is a Class of 2002 MacArthur Fellow and a fellow of the Association for the Advancement of Artificial Intelligence.