Unifying Geometric, Probabilistic, and Potential Field Approaches to Multi-robot Coverage Control

Mac Schwager, Jean-Jacques Slotine, and Daniela Rus

Abstract. This paper unifies and extends several different existing strategies for multi-robot coverage control, including ones based on Voronoi partitions, probabilistic models, and artificial potential fields. We propose a cost function form for coverage problems that can be specialized to fit different distributed sensing and actuation scenarios. We show that controllers based on Voronoi partitions can be approximated arbitrarily well by simpler methods that do not require the computation of a Voronoi partition. The performance of three different controllers designed with the new approach is compared in simulation. We also formally delineate two classes of multi-agent problems: consensus problems and non-consensus problems. We show that coverage control is a non-consensus problem and that it requires the optimization of a nonconvex cost function.

1 Introduction

One of the fundamental problems of multi-robot control is how to deploy a group of robots to spread out over an environment to carry out sensing, surveillance, data collection, or distributed servicing tasks. This operation is called coverage control, and several methods have been proposed to accomplish it in a distributed and efficient way. In this paper we introduce a unifying principle that ties together a number of common ways of accomplishing coverage control. We show that many of the existing methods can be described as special instances of gradient descent on a cost

Mac Schwager · Daniela Rus
Computer Science and Artificial Intelligence Laboratory, MIT,
77 Massachusetts Avenue, Cambridge, MA 02139, USA
e-mail: schwager@mit.edu, rus@csail.mit.edu

Jean-Jacques Slotine
Nonlinear Systems Laboratory, MIT, 77 Massachusetts Avenue,
Cambridge, MA 02139, USA
e-mail: jjs@mit.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 21–38.
springerlink.com

function. We propose a cost function form that specializes to give some common algorithms for coverage control, including Voronoi-based controllers, as in [7], controllers based on probabilistic models, as in [15], and artificial potential field-based controllers, as in [12].

Multi-robot coverage control serves as a prototype for many applications involving distributed sensing and distributed actuation. As examples of distributed sensing tasks, coverage control can be used to deploy underwater robots evenly over a coral reef to monitor coral health, or to deploy wheeled robots with cameras to spread out over a room for surveillance. Coverage control can also be used for multi-robot actuation tasks. For example, a coverage controller can be used to position oil clean-up robots over an oil spill so that they clean up the spill in minimum time. As another example, coverage control can be used to position de-mining robots to service a mine field in minimum time.

We describe a framework that is relevant to both sensing and actuation scenarios. We argue that Voronoi based methods are best suited to distributed *actuation* tasks, while a continuous approximation to the Voronoi decomposition is more appropriate for distributed *sensing* tasks. Furthermore, even in distributed actuation tasks, using a continuous approximation to the Voronoi cell improves the robustness and decreases the computational complexity of the controller. The continuous approximation is easy to compute regardless of the dimension of the space, while an exact Voronoi computation becomes unwieldy in higher dimensions.

As is typical for gradient based coverage control, the controllers we describe are provably convergent, robust to individual robot failures, and can adapt to environments that change slowly with respect to the speed of the robots. The controllers require that robots know the geometry of the environment and they know their own position in it using, for example, GPS or an indoor localization system. We also discuss how to accommodate the constraints of a communication network topology, but do not analyze this aspect of the problem in detail.

Related Work

Cortés et al. [7] introduced a controller for multi-robot coverage that works by continually driving the robots toward the centroids of their Voronoi cells. This inherently geometric strategy has seen many recent extensions to robots with a limited sensing radius in [6], to heterogeneous groups of robots and nonconvex environments in [18], and to incorporate learning of unknown environments in [21]. A recent text that presents much of this work in a cohesive fashion is [3] and an excellent overview is given in [16]. Coverage controllers also have been successfully implemented on robotic systems in [20, 19]. In this work we adopt notational conventions from the Voronoi based coverage control literature. Other common methods for coverage control take a probabilistic perspective. For example [15] proposes an algorithm for positioning robots to maximize the probability of detecting an event that occurs in the environment. Distributed dynamic vehicle routing scenarios are considered in [1, 17], in which events occur according to a random process and are

serviced by the robot closest to them. Another common coverage control method is for robots to drive away from one another using artificial potential fields [12]. Despite the rather different models and objectives in these works, there are two common points which motivate us to find a unifying principle: 1) they all rely upon an optimization, and 2) they all use controllers that solve this optimization through the evolution of a dynamical system.

Some existing approaches do not fit under the framework we propose in this paper. A significant body of work has looked at coverage control as a motion planning problem. A survey of this work can be found in [5], and some significant contributions can be found in, for example, [4, 14] and the citations therein. Other authors have proposed information theoretic algorithms which consider placing sensors sequentially rather than driving them with a controller. Works such as [10, 13] position sensor nodes to maximize information for the sake of estimating a Gaussian random process in the environment.

Contributions

In the present work we focus on multi-agent deployment as an optimization problem. This is advantageous because it is amenable to geometric, probabilistic, and analytical interpretations, all of which have been seen in a separate light in the past. Our optimization approach ties together much of the existing literature on coverage control. Specifically, our contributions are: 1) We propose a cost function, putting particular emphasis on the role of a *mixing function*, a previously unrecognized component that captures critical assumptions about the coverage task. 2) We introduce a family of mixing functions with a free parameter, α, and show that different values of the parameter correspond to different assumptions about the coverage task, specifically showing that a minimum variance solution (i.e. a probabilistic strategy) is obtained with a parameter value of $\alpha = -1$, and Voronoi coverage (a geometric strategy) is recovered in the limit $\alpha \to -\infty$. 3) We prove a new result linking the convexity of a cost function to the multi-agent phenomenon of consensus. We show that coverage tasks are fundamentally different from consensus, and that they require the optimization of a nonconvex cost function. This suggests inherent limitations to gradient descent controller designs, which are pervasive in the coverage control literature.

The paper is organized as follows. In Section 2 we introduce the cost function, describing the purpose of each of its parts including the mixing function. We then produce a class of provably stable distributed coverage controllers by taking the gradient of the cost function. In Section 3 we derive three special cases of the controller; a Voronoi controller, a minimum variance controller, and a potential field controller. Section 4 presents our results on the relation between the convexity of a cost function, and multi-agent consensus. Simulation results are given in Section 5 and conclusions are in Section 6.

2 Generalized Coverage

In this section we introduce a general multi-agent cost function. We will use this cost function to define a new class of multi-agent coverage controllers by introducing a *mixing function*, which describes how information from different robots should be combined. We use the cost function to derive a stable gradient descent controller.

2.1 Coverage Cost Function

Let there be n robots[1], and let robot i have a position $p_i \in \mathscr{P} \subset \mathbb{R}^{d_p}$, where $\mathscr{P}$ is the state space of a robot, and d_p is the dimension of the space. The vector of all robot positions is denoted $P = [p_1^T, \ldots, p_n^T]^T \in \mathscr{P}^n$, and we will call P the *configuration* of the robots. We want our robots to cover a bounded region $Q \subset \mathbb{R}^{d_q}$, which may or may not be related to the position space $\mathscr{P}$ of the robots. For example, the robots may be constrained to move in the space that they cover, so $\mathscr{P} = Q$ as in [7], or the robots may hover over a planar region that they cover with cameras, so $\mathscr{P} \subset \mathbb{R}^3$ and $Q \subset \mathbb{R}^2$, as in [19].

For each robot, a cost of sensing, or servicing, a point $q \in Q$ is given by a function $f(p_i, q)$. For simplicity of analysis we assume that $f(p_i, q)$ takes on only non-negative values, and that it is differentiable with respect to p_i.[2] The sensor measurements of the n robots are combined in a function $g(f(p_1, q), \ldots, f(p_n, q))$, which we will call the *mixing function*. The mixing function embodies assumptions about the coverage task; that is, by changing the mixing function we can derive Voronoi based coverage control, probabilistic coverage control, and a variety of other kinds of distributed controllers.

Combining these elements, we propose to use a cost function of the form

$$\mathscr{H}(P) = \int_Q g(f(p_1, q), \ldots, f(p_n, q))\phi(q)\, dq. \tag{1}$$

where $\phi : \mathbb{R}^{d_q} \mapsto \mathbb{R}_{>0}$ (we use the notation $\mathbb{R}_{>0}$ to mean the set of positive real numbers and $\mathbb{R}^d_{>0}$ the set of vectors whose components are all positive, and likewise for $\mathbb{R}_{\geq 0}$ and $\mathbb{R}^d_{\geq 0}$) is a weighting of importance over the region Q. Intuitively, the cost of the group of robots sensing at a single arbitrary point q is represented by the integrand $g(f(p_1, q), \ldots, f(p_n, q))$. Integrating over all points in Q, weighted by their importance $\phi(q)$ gives the total cost of a configuration of the robots. We want to find controllers that stabilize the robots around configurations P^* that minimize $\mathscr{H}$. We will see in Section 4 that for coverage, and many other multi-agent problems, $\mathscr{H}$ is necessarily nonconvex, therefore gradient based controllers will yield *locally* optimal robot configurations. The cost function (1) will be shown to subsume

[1] We will use the term robot throughout, though the framework is suitable for general mobile sensing agents, including biological ones.

[2] This requirement can be generalized considerably as in [6] to the case where $f(p_i, q)$ is piece-wise continuous with a finite number of jump discontinuities. A finite sensor footprint can be modeled with a single jump discontinuity.

several different kinds of existing coverage cost functions. Drawing out the relations between these different coverage algorithms will suggest new insights into when one algorithm should be preferred over another.

2.2 Mixing Function

The mixing function $g_\alpha : \mathbb{R}^n_{\geq 0} \mapsto \mathbb{R}$ describes how information from different robots should be combined to give an aggregate cost of the robots sensing at a point q. This is shown graphically in Fig. 1 where the overlap of the two sensors is shown for illustrative purposes as the intersection of two circles. We propose a mixing function of the form

$$g_\alpha(f_1, \ldots, f_n) = \Big(\sum_{i=1}^{n} f_i^\alpha\Big)^{\frac{1}{\alpha}}, \tag{2}$$

with a free parameter $\alpha < 0$. The arguments $f_i \geq 0$ are real valued, and in our context they are given by evaluating the sensor function $f(p_i, q)$, hence the notation f_i. To be precise, the expression in (2) is undefined when $f_j = 0$ for some j, therefore in this case we define g_α by its limit, $g_\alpha(f_1, \ldots, 0, \ldots, f_n) = \lim_{f_j \to 0} \left(\sum_{i=1}^n f_i^\alpha\right)^{\frac{1}{\alpha}} = 0$.

This mixing function has several important properties. Firstly, notice that for $\alpha \geq 1$ it is the p-norm of the vector $[f_1 \cdots f_n]^T$. Specifically, it is convex for $\alpha \geq 1$ and as $\alpha \to \infty$, $g_\alpha(\cdot) \to \max_i(\cdot)$, which is the ℓ^∞ norm. However, we are interested in the regime where $\alpha < 0$. In this case $g_\alpha(\cdot)$ is not a norm because it violates the triangle inequality. In this regime it is also nonconvex, leading to a nonconvex cost function, which is a necessary attribute of coverage problems, as we will prove in Section 4.

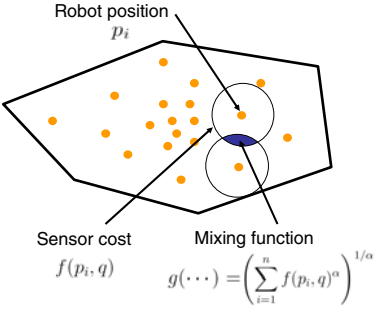
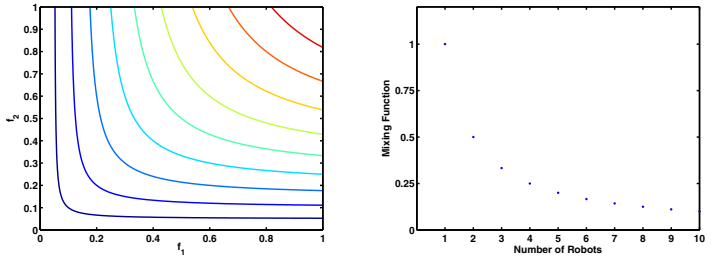


Fig. 1 The mixing function is illustrated in this figure. The mixing function determines how information from the sensors of multiple robots is to be combined, shown graphically as the intersection of the two circles in the figure.

One can readily verify[3] that as $\alpha \to -\infty$, $g(\cdot) \to \min_i(\cdot)$. From an intuitive point of view, with $\alpha < 0$, $g_\alpha(\cdot)$ is smaller than any of its arguments alone. That is, the cost of sensing at a point q with robots at p_i and p_j is smaller than the cost of sensing with either one of the robots individually. Furthermore, the decrease in g_α from the addition of a second robot is greater than that from the addition of a third robot, and so on. There is a successively smaller benefit to adding more robots. This property is often called supermodularity, and has been exploited in a rather different way in [13]. Plots of level sets of $g_\alpha(f_1, f_2)$ for $\alpha = -1$ are shown in Fig. 2(a) and the decrease in $g_\alpha(\cdot)$ as the number of arguments grows is shown in Fig. 2(b). In this work we consider the number of robots to be fixed, but it is useful to illustrate the supermodularity property of the mixing function by considering the successive addition of new robots. Including this mixing function in the cost function from (1) gives

$$\mathscr{H}_\alpha = \int_Q \Big(\sum_{i=1}^{n} f(p_i, q)^\alpha \Big)^{\frac{1}{\alpha}} \phi(q)\, dq. \tag{3}$$

To model scenarios with a finite sensor footprint, we can also let $f(p_i, q)$ be infinite in some areas, in which case to keep the cost function bounded and differentiable it becomes necessary to include a prior w in the mixing function, yielding the variation $g_\alpha(f_1, \ldots, f_n) = \left(\sum_{i=1}^{n} f_i^\alpha + w^\alpha \right)^{1/\alpha}$. An application of this case was explored in [19] to design a controller for positioning multiple flying robots with downward facing cameras.



(a) Mixing Function Contours (b) Mixing Function Supermodularity

Fig. 2 The proposed mixing function with $\alpha = -1$ is shown in this figure. Fig. 2(a) shows a contour plot of the mixing function with two arguments. The nonlinear decrease in the function as more sensors are added, a property known as supermodularity, is shown in Fig. 2(b). The plot was generated with each robot fixed at 1 unit from the point q.

[3] We know $\lim_{\beta\to\infty}[\sum_i h_i^\beta]^{1/\beta} = \max_i h_i$. Write $\lim_{\alpha\to-\infty}[\sum_i f_i^\alpha]^{1/\alpha}$ as $\lim_{\beta\to\infty}[[\sum_i h_i^\beta]^{1/\beta}]^{-1}$ with $h_i = 1/f_i$ and $\beta = -\alpha$. We have $\lim_{\beta\to\infty}[[\sum_i h_i^\beta]^{1/\beta}]^{-1} = [\max_i h_i]^{-1} = [\frac{1}{\min_i f_i}]^{-1} = \min_i f_i$.

2.3 Gradient Control

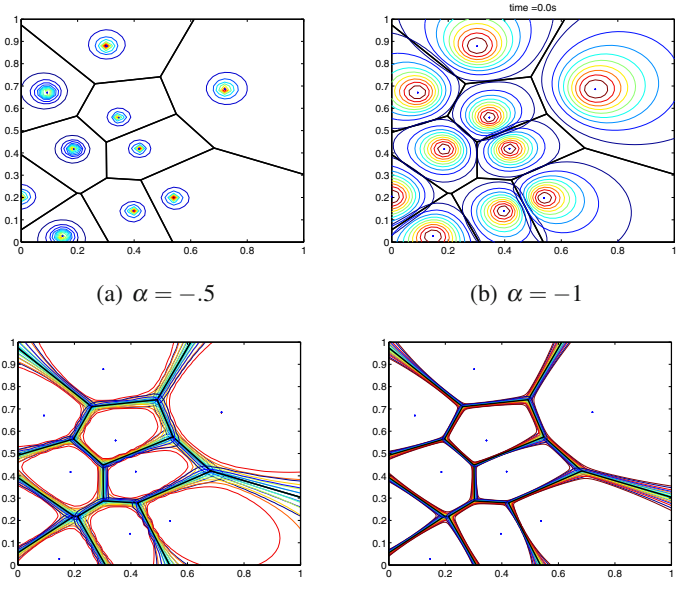
In order to derive a gradient descent controller, we take the derivative of the cost function $\mathscr{H}_\alpha$ with respect to the state of robot i to get

$$\frac{\partial \mathscr{H}_\alpha}{\partial p_i} = \int_Q \left(\frac{f(p_i,q)}{g_\alpha} \right)^{\alpha-1} \frac{\partial f(p_i,q)}{\partial p_i} \phi(q)\, dq.$$

To provide some intuition about the meaning of this function, notice that in the case that $f(p_i,q)$ is strictly increasing, the function inside the integral $(f(p_i,q)/g_\alpha)^{\alpha-1}$ gives an approximation to the indicator function[4] of the Voronoi cell of agent i, the approximation improving as $\alpha \to -\infty$. This is shown graphically in Fig. 3. It can be readily verified that this function is continuous, and that at $f(p_i,q) = 0$ it takes the value 1, and at $f(p_j,q) = 0$ and $j \neq i$ it takes the value 0.

For simplicity, we choose the function $f(p_i,q)$ to be

$$f(p_i,q) = \frac{1}{2}\|q - p_i\|^2, \quad \text{so that} \quad \frac{\partial f(p_i,q)}{\partial p_i} = -(q - p_i).$$



(a) $\alpha = -.5$ (b) $\alpha = -1$

(c) $\alpha = -5$ (d) $\alpha = -10$

Fig. 3 Contour plots of $(f(p_i,q)/g_\alpha)^{\alpha-1}$ are shown for a configuration of ten agent positions. The Voronoi tessellation is shown as well for comparison. As the parameter α approaches $-\infty$, $(f(p_i,q)/g_\alpha)^{\alpha-1}$ becomes closer to the indicator function of the Voronoi cell V_i.

[4] The indicator function for a set $S \subset Q$ returns 1 for $q \in S$, and 0 otherwise.

Other choices of $f(p_i,q)$ were investigated in [6] and could be used here as well, including functions with discrete jumps that model a finite senor footprint. This function represents the cost of a single robot i sensing at the position q. Therefore the quadratic form is appropriate for light based sensors, such as cameras or laser scanners. Light intensity drops off as the inverse square of the distance from the source, so it is reasonable for the cost to be proportional to the square of the distance. For tasks in which robots have to drive to a point q for servicing, and we want the cost to be proportional to the distance travelled, it would be more appropriate to use $f(p_i,q) = \|q - p_i\|$, for example.

We propose to use a gradient-based controller

$$\dot{p}_i = -k\frac{\partial \mathscr{H}_\alpha}{\partial p_i} = k\int_Q \left(\frac{f(p_i,q)}{g_\alpha}\right)^{\alpha-1}(q-p_i)\phi(q)\,dq, \tag{4}$$

where $k > 0$ is a positive control gain. We assume that the robots have integrator dynamics, $\dot{p}_i = u_i$, so we can control their velocity directly. We have found experimentally, in [20] for ground vehicles and [19] for quadrotor air vehicles, that this is a fair assumption as long as a fast inner control loop is in place to track the desired $\dot{p}_i$.

Stability can be proved by a direct application of the following result, the first part of which is stated in [11] Chapter 9, Section 4 as a corollary to Theorem 1.

Theorem 1 (Stability and Gradient Systems). *Let P^* be a critical point of $\mathscr{H}_\alpha$. Then P^* is a locally asymptotically stable equilibrium of the gradient system $\dot{P} = -\partial \mathscr{H}_\alpha / \partial P$ if and only if P^* is an isolated minimum of $\mathscr{H}_\alpha$.*

Corollary 1 (Coverage Control Stability). *For a group of robots with closed-loop dynamics*

$$\dot{p}_i = k\int_Q \left(\frac{f(p_i,q)}{g_\alpha}\right)^{\alpha-1}(q-p_i)\phi(q)\,dq$$

only configurations P^* for which $\mathscr{H}_\alpha(P^*)$ is an isolated local minimum are locally asymptotically stable.

Proof: The corollary follows directly from Theorem 1 and the fact that $\dot{P} = [\dot{p}_1^T \cdots \dot{p}_n^T]^T$ is the negative gradient of $k\mathscr{H}_\alpha$, which has the same critical points at $\mathscr{H}_\alpha$.

Remark 1 (Stability Vs. Convergence). Corollary 1 is somewhat different from typical stability results in the coverage control literature. It is more common to use LaSalle's invariance principle to prove convergence to a configuration where $\partial \mathscr{H}_\alpha / \partial P = 0$. The standard proof of this convergence applies also to the controller in (4). Unfortunately, such configurations are not necessarily local minima; they may be saddle points or local maxima. Our proof specifies that the system prefers local minima, not saddle points or maxima.

Remark 2 (Network Requirements). The computation of the controller requires that robot i knows the states of all the robots in the network. For this to be feasible there must either be a global supervisor or a fully connected network communication topology. It would be more useful if the controller depended only upon the states of robots with which it communicates. We suggest two methods to accomplish this, but we do not analyze them in detail in this paper. First, robot i can approximate its control law simply by computing (4) using only the states of the robots with which it is in communication. We expect this to give a good approximation because the function $(f(p_j,q)/g_\alpha)^{\alpha-1}$ depends weakly upon the states of agents that are not Voronoi neighbors, especially for small values of α, as evident from Fig. 3 . A rigorous stability analysis of this approximation scheme is difficult, however. A second option is for a robot i to use an estimated configuration vector, $\hat{P}$, in its calculation of the control law. The estimated configuration can be updated online using a standard distributed consensus algorithm (a so called "consensus estimator"). We expect that such a scheme may be amenable to a rigorous stability proof as its architecture is similar to adaptive control architectures. The investigation of these matters is left for future work.

3 Deriving Special Cases

In this section we show how the cost function 1 can be specialized to give three common kinds of coverage controllers, a Voronoi controller, which is geometric in nature, a minimum variance controller, which has a probabilistic interpretation, and a potential field controller. We conjecture that other coverage objectives beyond these three can be achieved with different choices of the mixing function parameter α.

Voronoi Coverage, $\alpha \to -\infty$

The Voronoi-based coverage controller described in [7] is based on a gradient descent of the cost function

$$\mathscr{H}_{\mathscr{V}} = \sum_{i=1}^{n} \int_{V_i} \frac{1}{2}\|q-p_i\|^2 \phi(q)\, dq,$$

where $V_i = \{q \in Q \mid \|q-p_i\| \leq \|q-p_j\|, \forall j \neq i\}$ is the Voronoi cell of robot i and the use of the subscript $\mathscr{V}$ is to distinguish it from $\mathscr{H}$ and $\mathscr{H}_\alpha$. The Voronoi partition can equivalently be written using the min function as

$$\mathscr{H}_{\mathscr{V}} = \int_Q \min_i (\frac{1}{2}\|q-p_i\|^2)\phi(q)\, dq,$$

because a point q is in the Voronoi cell V_i if and only if $\|q-p_j\|$ is minimized for $j=i$. As noted in Section 2.2, $\lim_{\alpha\to-\infty} g_\alpha(f_1,\ldots,f_n) = \min_i f_i$. Therefore $\mathscr{H}_{\mathscr{V}}$ is a special instance of (3) with the mixing function $g_{-\infty} = lim_{\alpha\to-\infty} g_\alpha$ and $f(p_i,q) = 1/2\|q-p_i\|^2$.

The choice of the min function for a mixing function now warrants some reflection. Consider a distributed actuation scenario in which we want to position robots so as to service an event that occurs randomly at some point in the environment q. Suppose any robot is equally capable of rendering the service, robots have to physically travel to the event to render the service, and our objective is to service an event as quickly as possible. Naturally, an event should be serviced by the robot that is closest to it, as it will reach the event the most quickly. In this case, the min function is the appropriate choice for a mixing function. By using the min function we are saying that the cost incurred by all the robots due to the event at q is the same as that incurred by the robot that is closest to q.

On the other hand, consider a sensing task in which an event of interest occurs randomly at a point q and is sensed at a distance by sensors located on the robots. In this case the use of the min function is more difficult to justify. Using the min function in this instance would imply that even though both p_i and p_j have some sensory information about the event, the cost function only counts the information from the one that is closest to q. This seems to be a poor choice of cost function for sensing, since in such cases we would want to capture the intuition that two sensors are better than one. The mixing function (2) captures this intuition. Furthermore, even in distributed actuation tasks, using a continuous approximation to the Voronoi cell improves the robustness of the controller. The discrete, geometric nature of the Voronoi computation combined with the continuous controller can lead to chattering, and small sensing errors can result in large changes in the control input. Fortunately, the Voronoi tessellation can be approximated arbitrarily well by choosing a small value of α, thereby preserving the Voronoi controller behavior while improving robustness.

Minimum Variance Coverage, $\alpha = -1$

We show in this section that setting the mixing function parameter to $\alpha = -1$ causes the robots to minimize the expected variance of their measurement of the location of a target of interest. As a side effect, we will formulate an optimal Bayesian estimator for the location of the target given the measurements of the agents.

Suppose our agents are equipped with sensors that give a noisy measurement of the position of a target in the environment. Let the target position be given by a random variable q that takes on values in Q, and agent i gives a measurement $y_i = q + w$, where $w \sim N(0, I_2\sqrt{f(p_i,q)})$ is a bi-variate normally distributed random variable, and where I_2 is the 2×2 identity matrix. The variance of the measurement, $f(p_i,q)$, is a function of the position of the sensor and the target. Intuitively one would expect a sensor to localize a target with more precision the closer the target is to the sensor. Then the measurement likelihood of agent i is $\mathbb{P}(y_i \mid q : p_i) = 1/(2\pi f(p_i,q))\exp\{-\|y_i - q\|^2/(2f(p_i,q))\}$, and the notation $\mathbb{P}(\cdot : p_i)$ is to emphasize that the distribution is a function of the agent position. Assume the measurements of different agents conditioned on the target position are independent. Also, let $\phi(q)$ be the prior distribution of the target's position. Then Bayes rule gives the posterior distribution,

$$\mathbb{P}(q \mid y_1, \ldots, y_n) = \frac{\prod_{i=1}^{n} \mathbb{P}(y_i \mid q : p_i)\phi(q)}{\int_Q \prod_{i=1}^{n} \mathbb{P}(y_i \mid q : p_i)\phi(q)\, dq}. \tag{5}$$

One can use the posterior to obtain a Bayesian estimate of the position of the event q given the measurements. For example, one may choose to estimate q using the mean, the median, or the maximum of the posterior in (5).

Our interest here, however, is not in estimating q. Instead we are interested in positioning the robots so that whatever estimate of q is obtained is the best possible one. To this end, we seek to position the robots to minimize the variance of their combined sensor measurements. The product of measurement likelihoods in the numerator of (5) can be simplified to a single likelihood function, which takes the form of an un-normalized Gaussian

$$\prod_{i=1}^{n} \mathbb{P}(y_i \mid q : p_i) = A \exp\left\{ -\frac{\|\bar{y} - q\|^2}{2g_{-1}(\cdot)} \right\},$$

whose variance is equivalent to our mixing function $g_{-1}(\cdot) = \left(\sum_{i=1}^{n} f(p_i, q)^{-1}\right)^{-1}$. The values of A and $\bar{y}$ are not important in this context. If we want to position the robots so as to obtain the most decisive information from their sensors, we should move them to minimize this variance. Notice, however, that $g_{-1}(f(p_1, q), \ldots, f(p_n, q))$ is a random variable since it is a function of q. Taking the expectation over q of the likelihood variance gives our original cost function,

$$\mathscr{H}_{-1} = \mathbb{E}_q[g_{-1}(f(p_1, q), \ldots, f(p_n, q))] = \int_Q g_{-1}(f(p_1, q), \ldots, f(p_n, q))\phi(q)\, dq. \tag{6}$$

Thus we can interpret the coverage control optimization as finding the agent positions that minimize the expected variance of the likelihood function for an optimal Bayes estimator of the position of the target.

A more theoretically appealing criterion would be to position the agents to minimize the variance of the *posterior* distribution in (5). This gives a considerably more complicated cost function. The complication of this cost function and the fact that gradients can not be easily computed makes it a less practical option.

Potential Field Coverage, $\phi(q) = \sum_{i=1}^{n} \delta(\|q - p_i\|)$

The third type of coverage controller we consider is significantly different from the previous two in that it does not involve an integral over the environment. Instead it relies on the idea that robots should push away from one another to spread out over an environment, but should not move too far from one another or else they will become disconnected. Surprisingly, however, we will show that this rather different coverage philosophy can be reconciled with our generalized coverage cost function $\mathscr{H}$ in (1).

Let the importance function, $\phi(q)$, be given as a sum of delta-Dirac functions centered at each of the robot positions

$$\phi(q) = \sum_{i=1}^{n} \delta(\|q - p_i\|).$$

Substituting this for $\phi(q)$ in (1), the integral in $\mathscr{H}$ can then be evaluated analytically to give

$$\mathscr{H}_{\text{pot}} = \sum_{i=1}^{n} g(f(p_1, p_i), \ldots, f(p_n, p_i)),$$

and setting $g(f(p_1, p_i), \ldots, f(p_n, p_i)) = \sum_{j=1, j\neq i}^{n} f(p_j, p_i)$ gives a cost function for potential field based coverage (as well as models for flocking and herding)

$$\mathscr{H}_{\text{pot}} = \sum_{i=1}^{n} \sum_{j=1, j\neq 1}^{n} f(p_j, p_i), \tag{7}$$

where $f(p_j, p_i)$ can be interpreted as an inter-agent potential function. One choice for $f(p_j, p_i)$ is

$$f(p_j, p_i) = \frac{1}{6}\|p_j - p_i\|^{-2} - \|p_j - p_i\|^{-1}$$

which, taking the gradient of (7), yields the controller

$$\dot{p}_i = k \sum_{j=1, j\neq i}^{n} \left(\|p_j - p_i\|^{-2} - \frac{1}{3}\|p_j - p_i\|^{-3} \right) \frac{p_j - p_i}{\|p_j - p_i\|}. \tag{8}$$

Controllers similar to this one have been studied in a number of works, for example [12, 9, 22, 8]. There are numerous variations on this simple theme in the literature.

Computational Complexity

The gradient controllers described in this work must inevitably be discretized and implemented in a discrete time control loop. A common criticism of the Voronoi based coverage controller is that it is computationally intensive. At each iteration of the loop, a robot must re-compute its Voronoi cell. Additionally, the controller must compute spatial integrals over a region. In general, a discretized approximation must be used to compute the integral of $\phi(q)$ over the Voronoi cell, which is again computationally intensive. The two parameters that are important for computation time are the number of robots n and the number of grid squares in the integral computation, which we will call m. The typical decentralized algorithm for a single robot to compute its Voronoi cell (from [7]) runs in $O(n)$ time. The time complexity for computing a discretized integral is linear in the number of grid squares, and at each grid square requires a check if the center point is in the Voronoi cell, which is an $O(n)$ operation. Therefore the time complexity of the integral is in $O(nm)$. The Voronoi cell must be computed first, followed by the discretized integral, therefore the standard Voronoi controller has time complexity $O(n(m+1))$ at each step of the control loop.

Our controller in (4) does not require the computation of a Voronoi cell, but it does require the discretized spatial integral over the environment. We do not have

to check if a point is in a polygon, but the integrand we evaluate, namely g_α is linear in n. Therefore the integral computation still has time complexity $O(nm)$, which is the time complexity of the controller at each step of the control loop. Yet as α decreases, the behavior of the controller approaches that of the Voronoi controller. The controller we propose in this paper is therefore significantly simpler in implementation (since it does not require the Voronoi computation), and it is faster computationally. Also, as the dimension of the state space increases, it becomes more difficult to compute and even to store the Voronoi decomposition, whereas the computation of g_α is straightforward in a space of any dimensionality.

4 Convexity and Consensus

Since we treat the multi-agent coverage problem as an optimization, it is natural to ask what sort of optimization we are dealing with, and what optimization tools can be brought to bear to solve it. We show in this section that the cost function in (3) is nonconvex, and that nonconvexity is a required feature of a large class of multi-agent problems, however undesirable this may be from an optimization perspective. Specifically, we demonstrate a link between the *convexity* of a cost function and the multi-agent phenomena known as *consensus*. For our purposes, consensus describes a multi-agent configuration in which all agents take on the same state, $p_1 = p_2 = \ldots = p_n$. Consensus is geometrically represented in the state space $\mathscr{P}^n$ as a d-dimensional hyperplane that passes through the origin (from the $d_p(n-1)$ independent equality constraints). This is illustrated by the diagonal line in Fig. 4 in a simplified 2D setting. We will prove, with some technical assumptions, that a multi-agent problem with a *convex* cost function admits at least one globally optimal consensus solution.

We begin with some basic definitions and facts from convex optimization which can be found in any standard text on the topic, for example [2]. A set $\Omega \subset \mathbb{R}^n$ is called convex if, for any two points in Ω, all points along the line segment joining them are also in Ω. An important consequence of the convexity of Ω is that any convex combination of points in Ω is also in Ω. A convex combination of m points $x_i \in \Omega$ is one of the form

$$x = \sum_{i=1}^{m} \alpha_i x_i \quad \text{where} \quad \sum_{i=1}^{m} \alpha_i = 1 \quad \text{and} \quad \alpha_i \geq 0 \quad \forall i.$$

A function $f : \Omega \mapsto \mathbb{R}$ is called convex if

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y) \quad \forall x, y \in \Omega \quad \text{and} \quad \forall \alpha \in [0,1].$$

This is equivalent to saying that the set of all points lying on or above the function $f(x)$ is a convex set (this set is known as the epigraph of $f(x)$). A function is called strictly convex if the '$\leq$' can be replaced with a '$<$' in the above relation. Also, we will use the word minimum to mean minimum or infimum if no minimum exists. We

now state a theorem that follows from Weierstrass' Theorem and some well-known properties of convex functions.

Theorem 2 (Minima of Convex Functions). *For a continuous, convex function* $f : \Omega \mapsto \mathbb{R}$*, where the domain* $\Omega \subset \mathbb{R}^n$ *is convex, if any of the following are true:*

1. Ω *is bounded*
2. *There exists a scalar* γ *such that the level set* $\{x \in \Omega \mid f(x) \leq \gamma\}$ *is nonempty and bounded*
3. f *is such that* $lim_{\|x\| \to \infty} f(x) = \infty$

then the set of global minima of f *is non-empty and convex.*

We will apply this result to our multi-agent scenario. Consider a continuous multi-agent cost function $\mathscr{H} : \mathscr{P}^n \mapsto \mathbb{R}$. As before, an agent i has a state $p_i \in \mathscr{P} \subset \mathbb{R}^d$. It will be more convenient in this section to refer to a configuration of agents as a tuple $(p_1, \ldots, p_n) \in \mathscr{P}^n$, rather than the column vector notation used previously. Let us assume that agents are anonymous with respect to the cost function, by which we mean that the positions of any two agents can be interchanged without affecting the value of the cost function. This is formalized by the following assumption.

Assumption 1 (Anonymity of Agents) *The cost function* $\mathscr{H}$ *is such that*

$$\mathscr{H}(\ldots, p_i, \ldots, p_j, \ldots) = \mathscr{H}(\ldots, p_j, \ldots, p_i, \ldots) \quad \forall i, j \in \{1, \ldots, n\}.$$

Assumption 1 is in keeping with the ethos of complex, multi-agent systems, where the emphasis is on the global patterns that result from the interactions of many identical agents. Furthermore, let us assume that $\mathscr{H}$ and $\mathscr{P}^n$ satisfy at least one of the three properties in Theorem 2. Now we give the main result of this section.

Theorem 3 (Convexity and Consensus). *Under Assumption 1, if the cost function* $\mathscr{H}(p_1, \ldots, p_n)$ *is continuous and convex,* $\mathscr{P}^n$ *is convex, and one of the conditions in Theorem 2 is satisfied, then* $\mathscr{H}(p_1, \ldots, p_n)$ *has a global minimum such that* $p_i = p_j$ $\forall i, j \in \{1, \ldots, n\}$.

Proof: Our argument rests upon Assumption 1 and the fact from Theorem 2 that the set of minima of a convex function $\mathscr{H}$ is a convex set. Let h^* be the set of minima, and let $(\ldots, p_i^*, \ldots, p_j^*, \ldots)$ be an optimal solution in that set. By Assumption 1, $(\ldots, p_j^*, \ldots, p_i^*, \ldots)$ is also an optimal solution for any i and j. Therefore all permutations of components in $(p_1^*, \ldots, p_n^*)$ are optima. Then by convexity of h^*, all convex combinations of points in h^* are in h^*. In particular, the point $(\bar{p}, \ldots, \bar{p})$, where $\bar{p} = 1/n \sum_{i=1}^n p_i$ is an optimal solution (since it is a convex combination of permutations of $(p_1, \ldots, p_n)$). □

We show a geometric schematic of the proof argument in Fig. 4. The proof uses the fact that the convex set of minima must intersect the consensus hyperplane (the hyperplane where $p_i = p_j$ $\forall i, j$) at at least one point. A simple corollary follows.

Corollary 2 (Strict Convexity). *If the conditions of Theorem 3 are met and the cost function* $\mathscr{H}(p_1, \ldots, p_n)$ *is strictly convex, then the minimum is unique and is such that* $p_i = p_j$ $\forall i, j \in \{1, \ldots, n\}$.

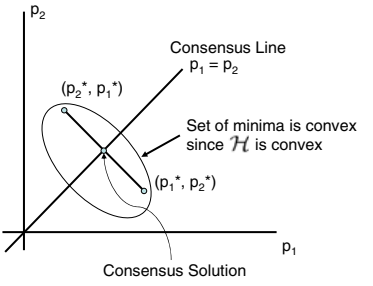


Fig. 4 This schematic shows the geometrical intuition behind the proof of Theorem 3 in a simplified 2D setting. Corollary 2 is proved by noticing that the set of minima is a single point (the consensus solution) if $\mathscr{H}$ is strictly convex.

Proof: A strictly convex function has at most one minimum over a convex domain. □

Remark 3 (Coverage is Nonconvex). Theorem 3 delineates two classes of multi-agent behaviors reminiscent of complexity classes in the theory of computation. One class, which we will call consensus behaviors, can be described as optimizing a convex cost function. The other class, which we will call non-consensus behaviors, is fundamentally different in that it can only be described with nonconvex cost functions. This is important because if we wish to design an optimization to solve a multi-agent problem, and we know that the problem cannot be solved satisfactorily by all the agents taking the same state, then we must use a nonconvex cost function. Likewise if we observe a multi-agent behavior in nature which cannot be described by all agents reaching the same state (the construction of a termite nest, for example), then an optimization-based explanation of this behavior must be nonconvex.

This is directly applicable to coverage problems. Indeed, coverage cannot be achieved with all agents moving to the same place, therefore coverage problems must involve the optimization of a nonconvex cost function. Our parameterized cost function $\mathscr{H}_\alpha$ from (3) is nonconvex for $\alpha < 1$ and is convex for $\alpha \geq 1$ because $\mathscr{H}_\alpha$ inherits the convexity properties of g_α. Correspondingly, we see that the robots become more highly aggregated as α approaches 1, and the robots all move to a common final position (i.e. consensus) when $\alpha \geq 1$. Theorem 3 explains why this is the case, and suggests that the search for a convex cost function for coverage control is futile.

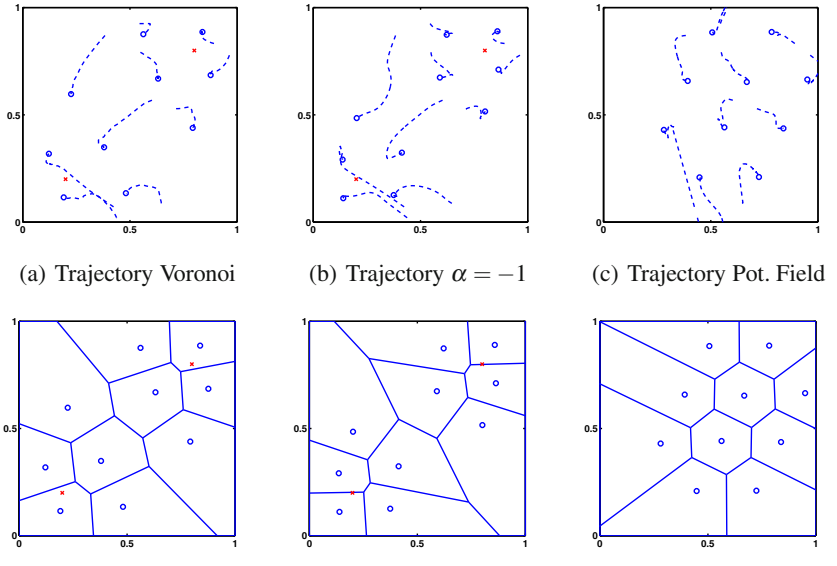
From an algorithmic point of view, however, this is unfortunate. Convex optimization has a powerful and well characterized tool set, but nonconvex optimization requires generally less efficient solution tools with looser guarantees. Distributed coverage controllers that use gradient methods (such as those in this paper) guarantee convergence to local optima, which is all one can expect for a nonconvex optimization. This points toward an open question for future work: are there nonconvex optimization methods not based on gradient descent that can be implemented in a multi-agent setting?

5 Simulation Results

The controller for the three scenarios described in Section 3 were simulated in a Matlab environment. The environment Q was taken to be a unit square, and the function $\phi(q)$ was set to be the sum of two Gaussian functions, one centered at $(.2,.2)$ and the other at $(.8,.8)$, both with variance .2. We expect to see a higher density of robots around areas of large $\phi(q)$. In our case, the robots group around the Gaussian centers.

The results of a simulation with ten robots using the Voronoi based controller, which corresponds to $\alpha \to -\infty$, is shown in Figs. 5(a) and 5(d). Similar plots are shown for the minimum variance controller, with $\alpha = -1$, in Figs. 5(b) and 5(e). Comparison of the two controllers shows that the Voronoi based controller causes the robots to spread out more, while as α increases, the robots group more closely together. When $\alpha > 1$, the cost function becomes convex, and the robots all move to the same position, which corroborates our results relating convexity to consensus (this is not shown in the plots).

The third scenario shown in Figs. 5(c) and 5(f) uses the potential field controller from (8). This controller uses a sum of delta-Dirac functions for $\phi(q)$ rather than a sum of Gaussians, which causes the robots to arrange themselves in the close-packed lattice pattern.



(a) Trajectory Voronoi (b) Trajectory $\alpha = -1$ (c) Trajectory Pot. Field

(d) Final Config. Voronoi (e) Final Config. $\alpha = -1$ (f) Final Config. Pot. Field

Fig. 5 Trajectories and final configurations are shown for ten robots using the gradient control law with the Voronoi controller (5(a), 5(d)), the minimum variance controller (5(b), 5(e)), and a potential field controller (5(c), 5(f)). The Voronoi tessellation is shown for all scenarios for comparison, even though the right two controllers do not use the Voronoi cells for control.

6 Conclusion

In this paper we introduce a unifying optimization framework for multi-robot coverage control that brings together several different existing coverage algorithms. We point out that important properties of the underlying coverage objective are embodied in the way sensor information or actuator capabilities are combined from different robots. We propose a parameterized function to accomplish this combination, where different parameter values are shown to lead to different kinds coverage algorithms. Finally, we prove that for coverage problems the underlying optimization is necessarily nonconvex, making global optimization an unrealistic objective, especially for gradient descent controllers.

Our work invites an immediate extension, which is how to approximate the gradient controller over a communication graph. We outlined two methods for doing this. In the future the stability and robustness properties of these methods should be characterized and other methods should be investigated as well. Also our recognition that coverage problems stem from nonconvex optimizations suggests some new research directions. Gradient descent controllers, which are the most common type in the coverage control literature, can only be expected to find local minima. Therefore it is worthwhile to look for other nonconvex optimization methods that can be implemented in a multi-agent setting. We expect that these open questions will motivate new results for coverage control.

Acknowledgements. This work was done in the Distributed Robotics Laboratory at MIT. This work was supported in part by the MURI SWARMS project grant number W911NF-05-1-0219, NSF grant numbers IIS-0426838, CNS-0520305, CNS-0707601, EFRI-0735953, the MAST project, the ONR MURI SMARTS project, and The Boeing Company.

References

1. Arsie, A., Frazzoli, E.: Efficient routing of multiple vehicles with no explicit communications. International Journal of Robust and Nonlinear Control 18(2), 154–164 (2007)
2. Bertsekas, D., Nedić, A., Ozdaglar, A.E.: Convex Analysis and Optimization. Athena Scientific, Nashua (2003)
3. Bullo, F., Cortés, J., Martínez, S.: Distributed Control of Robotic Networks (2008) (manuscript preprint), Electronically, `http://coordinationbook.info`
4. Butler, Z.J., Rizzi, A.A., Hollis, R.L.: Complete distributed coverage of rectilinear environments. In: Proceedings of the Workshop on the Algorithmic Foundations of Robotics, Hanover, NH (March 2000)
5. Choset, H.: Coverage for robotics—A survey of recent results. Annals of Mathematics and Artificial Intelligence 31, 113–126 (2001)
6. Cortés, J., Martínez, S., Bullo, F.: Spatially-distributed coverage optimization and control with limited-range interactions. ESIAM: Control, Optimisation and Calculus of Variations 11, 691–719 (2005)
7. Cortés, J., Martínez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. IEEE Transactions on Robotics and Automation 20(2), 243–255 (2004)

8. Dimarogonas, D.V., Kyriakopoulos, K.J.: Connectedness preserving distributed swarm aggregation for multiple kinematic robots. IEEE Transactions on Robotics 24(5), 1213–1223 (2008)
9. Gazi, V., Passino, K.M.: A class of repulsion/attraction forces for stable swarm aggregations. International Journal of Control 77(18), 1567–1579 (2004)
10. Hernandez, M.L., Kirubarajan, T., Bar-Shalom, Y.: Multisensor resource deployment using posterior Cramér-Rao bounds. IEEE Transactions on Aerospace and Electronic Systems 40(2), 399–416 (2004)
11. Hirsch, M.W., Smale, S.: Differential Equations, Dynamical Systems, and Linear Algebra. Academic Press, Inc., Orlando (1974)
12. Howard, A., Matarić, M.J., Sukhatme, G.S.: Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In: Proceedings of the 6th International Symposium on Distributed Autonomous Robotic Systems (DARS 2002), Fukuoka, Japan (June 2002)
13. Krause, A., Guestrin, C.: Near-optimal observation selection using submodular functions. In: Proceedings of 22nd Conference on Artificial Intelligence (AAAI), Vancouver, Canada (2007)
14. Latimer IV, D.T., Srinivasa, S., Shue, V.L., Sonneadnd, S., Choset, H., Hurst, A.: Towards sensor based coverage with robot teams. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 1, pp. 961–967 (May 2002)
15. Li, W., Cassandras, C.G.: Distributed cooperative coverage control of sensor networks. In: Proceedings of the IEEE Conference on Decision ans Control and the European Control Conference, Seville, Spain (December 2005)
16. Martínez, S., Cortés, J., Bullo, F.: Motion coordination with distributed information. IEEE Control Systems Magazine 27(4), 75–88 (2007)
17. Pavone, M., Smith, S.L., Bullo, F., Frazzoli, E.: Dynamic multi-vehicle routing with multiple classes of demands. In: Proceedings of American Control Conference, St. Louis, Missouri (June 2009)
18. Pimenta, L.C.A., Kumar, V., Mesquita, R.C., Pereira, G.A.S.: Sensing and coverage for a network of heterogeneous robots. In: Proceedings of the IEEE Conference on Decision and Control, Cancun, Mexico (December 2008)
19. Schwager, M., Julian, B., Rus, D.: Optimal coverage for multiple hovering robots with downward-facing cameras. In: Proceedings of the International Conference on Robotics and Automation (ICRA 2009), Kobe, Japan (May 2009)
20. Schwager, M., McLurkin, J., Slotine, J.J.E., Rus, D.: From theory to practice: Distributed coverage control experiments with groups of robots. In: Proceedings of the International Symposium on Experimental Robotics (ISER 2008), Athens, Greece (July 2008)
21. Schwager, M., Rus, D., Slotine, J.J.: Decentralized, adaptive coverage control for networked robots. International Journal of Robotics Research 28(3), 357–375 (2009)
22. Tanner, H.G., Jadbabaie, A., Pappas, G.J.: Flocking in fixed and switching networks. IEEE Transactions on Automatic Control 52(5), 863–868 (2007)