Covering Dynamic Demand with Multi-Resource Heterogeneous Teams

Mela Coffey and Alyssa Pierson

***Abstract*— In this work, we consider a team of heterogeneous robots equipped with various types and quantities of resources, and tasked with supplying these resources to multiple dynamic demand locations. We present an adaptive control policy that enables robots to serve a dynamic demand: we allow demand to deplete as robots supply resources, and we allow demand injection and movement of demand locations. We show that the demand is input-to-state stable (ISS) under our proposed resource dynamics, and thus the robots can drive the demand to a steady state. Finally, we present simulations and hardware experiments to demonstrate our approach, and demonstrate the benefits of coverage over a persistent monitoring approach.**

I. Introduction

When natural disasters occur, failure of traditional infrastructure may limit the ability for first responders to assess damage, distribute supplies, and help those in need. Further, traversal of the terrain may be dangerous to those first responders. Robotic delivery systems may provide a solution to help distribute supplies, however, they must be capable of dispersing supplies to evolving, dynamic demand of the different resource types. We envision a team of supply delivery robots capable of carrying different types and quantities of supplies, delivering them to desired locations within the environment. The distributed team adapts to changes in the demand, providing spatial coverage.

Here, we explore the general spatial supply-and-demand problem. In our prior work [1], we solved the problem of supplying multiple types and quantities of resources to various areas of demand. We proposed a Voronoi-based coverage control approach to deploy robots in a continuous, distributed fashion. This prior work only considered a static demand focusing on the initial deployment of a team. Within this paper, we consider dynamic demand that can move and increase over time, as well as depleting supplies on the robots. These methods allow for a longer deployment in evolving environments.

By considering that the demand of resources evolves over time, we extend the capabilities of the system. Consider again the example of a team of delivery robots assisting human first responders after a natural disaster. The different resources needed could include medical supplies, batteries, food, communication devices, or tools. Some robots within the team may be very good at delivering one particular resource, while other robots may carry multiple resource types. The need for these resources across some environment evolves as time passes and more information is known to the first responders. Allowing the resource demand to move and grow over time models the changing delivery locations, while injecting demand into the system models the need for continued or additional resources. As robots provide coverage, their supplies deplete and the demand reduces, such as delivering single-use medical supplies.

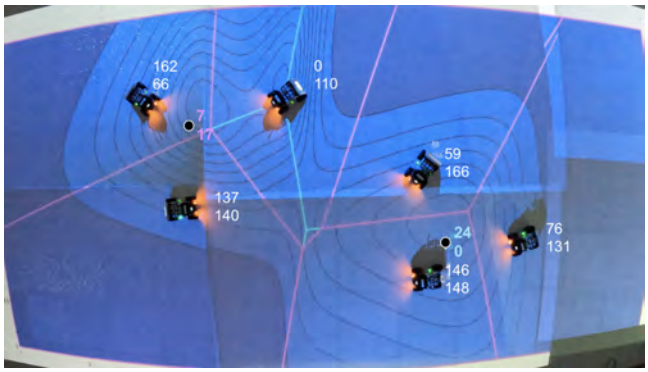


Fig. 1. Our experiments demonstrate a team of 6 robots supplying 2 different types of resources to 2 dynamic locations (○). Quantities of each resource carried/demanded are printed to the right of the robot/○. The contours indicate density in the space, and solid lines represent Voronoi boundaries.

More generally, the spatial supply-and-demand problem pertains to task allocation, drone delivery, persistent monitoring, and job assignment for heterogeneous robot teams. We model the demand of resources as continuous multi-peak distributions over the environment, akin to the probability that resource is needed at a particular location. This paper focuses on the robot team's response and coverage to a dynamic demand. The main contributions of this paper are:

- Adaptive coverage control for deploying a heterogeneous multi-robot team to multiple, moving demand locations with depleting and injected demand;
- Analyses to prove minimization of the locational cost and input-to-state stability of the demand;
- Demonstrate the benefits of coverage control compared to persistent monitoring through simulations; and
- Demonstrate real-time performance through hardware-in-the-loop experiments.

Related Work

Prior work has indeed explored coverage control of heterogeneous teams. For example, [2], [3], [4] explored coverage control in relation to task allocation, specifically the problem of deploying robots with different sensors to cover regions

Mela Coffey and Alyssa Pierson are with the Department of Mechanical Engineering at Boston University, Boston, MA 02115, USA. mcoffey@bu.edu, pierson@bu.edu

This work supported in part by NSF CAREER Award 2235622 and a Boston University startup grant. Their support is gratefully acknowledged.

requiring those sensors. This approach is equivalent to allocating robots with different types of resources to demand. Other approaches introduce heterogeneity by using weighted Voronoi diagrams, where the size of a robot's Voronoi cell depends on some factor, such as sensor quality [5] or robot speed [6]. This approach is equivalent to considering multiple resources of one type. We bridge the gap between these approaches in our previous work [1].

In relation to the general spatial supply-and-demand problem, several studies have explored task allocation and coordination techniques in multi-robot systems. For example, [7] focuses on multi-robot task allocation under task uncertainty and temporal constraints. The authors of [8] propose a framework to enable adaptive dynamic task allocation when robot capabilities are unknown. In [9], authors propose a receding horizon framework to dynamically assign robots to demand regions. The works [10], [11] use a Markov decision process (MDP) approach to dynamically allocate resources in uncertain environments. While these works tackle the task/resource allocation problem, they do not consider heterogeneity within the team.

The supply-and-demand problem can also be tackled by monitoring, exploration, routing, and scheduling. For example [12] considers tasks with heterogeneous requirements, and uses a combination of the traveling salesman and knapsack problems. The authors of [13] consider a heterogeneous team of drones deployed from a moving truck to complete tasks, implementing the vehicle routing problem for drone routes and the traveling salesman problem for scheduling. In [14], authors focus on routing for heterogeneous multi-robot task allocation in smart warehouses. These approaches, while they account for heterogeneity, require that the problems be solved prior to deployment, and thus cannot account for changes in demand or the environment.

Thus far, researchers have approached the multi-robot resource allocation problem as task allocation without considering resource capacity or heterogeneity, or as monitoring and scheduling without the ability to adapt to changes in the system. Contrary to prior work, our approach enables a heterogeneous multi-robot team, with different types and quantities of resources, to perpetually serve a dynamic demand. By utilizing Voronoi-based coverage control, we can guarantee locational optimality while naturally adapting to changes in the system and environment in a distributed fashion. We enable robots to adapt to not only the movement of the demand, but also changes in the resources of the system, including transfer of resources from supply to demand, injection of demand, and moving demand.

The remainder of this paper is organized as follows: Section II details the problem formulation, with a brief review of our prior work and a definition of the resource dynamics. In Section III, we update our previous control policy and prove that our robots meet demand. In Section IV, we demonstrate the performance of our approach through simulations and experiments. Finally, we provide conclusions in Section V.

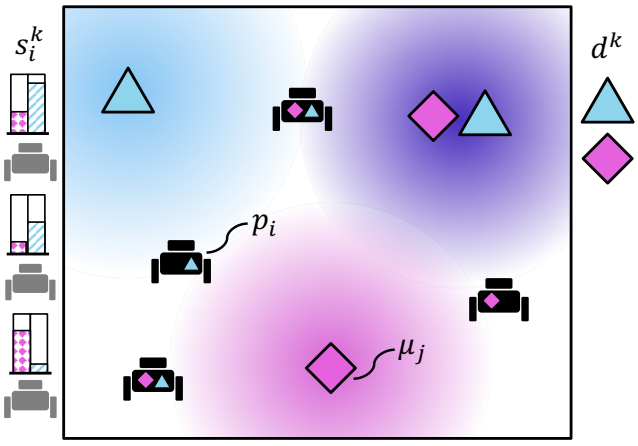


Fig. 2. Our goal is to assign robots with varying supplies to meet varying demand across the environment. In this example, we have three demand locations μ_j, each requiring one or both of the resources denoted by the $\triangle$ and $\diamond$. Each robot at location p_i carries different amounts of resources, and some robots may specialize in one resource type.

II. Problem Formulation

In this section, we review key technical concepts from our prior work detailed in [1], then introduce the resource dynamics that represent evolving demand. Consider a team of $i = \{1, ..., N\}$ robots, $i \in \mathcal{R}$, in a bounded, convex environment $Q \subset \mathbb{R}^2$, with points in Q denoted q, and robot positions denoted $p_i \in Q$. We denote each type of resource with the index $k \in \mathcal{S}$. Demand for the resources occurs at $j = \{1, ..., M\}$ different locations through the environment, with the set of all demand locations denoted $\mathcal{D}$. We let $\phi_j(\cdot)$ represent the density function associated with demand j, centered around a peak $\mu_j \in Q$. We write $d_j^k \in \mathbb{R}_+$ as the demand of resource k that is required at location j. Each robot i contains a supply of resources, with $s_i^k \in \mathbb{R}_+$ denoting the supply of resource k carried by robot i. Our goal is to find the control policy for the team to deliver supplies to the demand within the environment. Figure 2 illustrates this deployment problem of our robots with different resource supplies to the varying demand locations.

A. Heterogeneous Coverage for Resource Allocation

In [1], we enabled coverage control for multi-resource allocation. To allow robots to distinguish between the different demands, we define M different Voronoi partitions. We assign robot i to partition j if robot i can supply resources to demand j. Formally, we define the set of robots assigned to partition j as

$$\mathcal{P}^j = \{i \mid \exists k : d_j^k > 0 \wedge s_i^k > 0\}. \tag{1}$$

Then, the Voronoi cell V_i^j for robot $i \in \mathcal{P}^j$ is given by

$$V_i^j = \{q \in Q \mid \|q - p_i\| \leq \|q - p_l\|, \forall l \in \mathcal{P}^j, \forall l \neq i\}. \tag{2}$$

We can also compute the mass and centroid [3], [4], respectively, of each cell V_i^j by

$$m_i^j = \int_{V_i^j} \phi_j(q,p,t)dq$$
$$c_i^j = \frac{\int_{V_i^j} q\phi_j(q,p,t)dq}{\int_{V_i^j} \phi_j(q,p,t)dq} = \frac{1}{m_i^j}\int_{V_i^j} q\phi_j(q,p,t)dq,$$

where $\phi_j(q,p,t)$ is our proposed time-varying, position-varying density function associated with demand j. We define the following density function [1] to drive robots to demand j based on their resource capacities s_i^k with respect to the demand d_j^k:

$$\phi_j(q,p,t) = \frac{D_j}{S_j},$$

where

$$D_j = \sum_{k\in\mathcal{S}} d_j^k \exp\left[-\frac{1}{\sigma}(q-\mu_j)^\mathsf{T}\Sigma^{-1}(q-\mu_j)\right],$$

$$S_j = \sum_{l\in\mathcal{P}^j}\sum_{k\in\mathcal{S}, d_j^k\neq 0} s_l^k \exp\left[-\frac{1}{\sigma}(q-p_l)^\mathsf{T}\Sigma^{-1}(q-p_l)\right],$$

σ is a positive constant, and Σ is a constant covariant matrix. Figure 2 illustrates the peaks D_j that contribute to this density function.

While traditional Voronoi-based coverage control [15] utilizes Lloyd's algorithm [16] to move robots to their centroids, we implement a minimum-energy, constraint-driven control policy [17], [18] to account for our time-varying density function. We first define the total cost [1] as

$$J(p,t) = \sum_{i\in\mathcal{R}}\sum_{j\in\mathcal{D}} J_i^j(p,t) = \sum_{i\in\mathcal{R}}\sum_{j\in\mathcal{D}} \frac{1}{2} m_i^j \|p_i - c_i^j\|^2. \quad (3)$$

We let each robot i obey the single integrator dynamics

$$\dot{p}_i = u_i, \quad (4)$$

where u_i is computed by solving [1]

$$\begin{aligned}\min_{u_i} \quad & \|u_i\|^2 \\ \text{s.t.} \quad & \sum_{j\in\mathcal{D}} (-m_i^j(p_i-c_i^j)^\mathsf{T}(I - \frac{\partial c_i^j}{\partial p_i})u_i \\ & - \frac{1}{2}\|p_i - c_i^j\|^2 \frac{\partial m_i^j}{\partial p_i}u_i) \geq \sum_{j\in\mathcal{D}}(-\alpha(-J_i^j(p,t)) \\ & - m_i^j(p_i-c_i^j)^\mathsf{T}\frac{\partial c_i^j}{\partial t} + \frac{1}{2}\frac{\partial m_i^j}{\partial t}\|p_i - c_i^j\|^2),\end{aligned} \quad (5)$$

at each point in time, where I is the identity matrix, and α is an extended class $\mathcal{K}$ function.

B. Resource Dynamics

Our prior approach [1] deploys robots with static s_i^k and d_j^k to static demand locations μ_j using a coverage control approach. We now wish to enable dynamic demand that can change location, such that we allow demand to move $\mu_j = \mu_j(t)$, and we define the depletion dynamics as they relate to the changes in supply and demand. First, we define a few variables. Recall that $\dot{d}_j^k$ is the rate of change of resources of type k required (demanded) at location j. Note that $\dot{d}_j^k$ must depend on 1) the rate at which robots are supplying resources of type k to location j and 2) the injected demand of type k to location j. Therefore, we can define the demand dynamics $\dot{d}_j^k$ as

$$\dot{d}_j^k = \zeta_j^k(t) - \sum_{i\in\mathcal{P}^j} \dot{s}_{ij}^k(d_j^k, \mu_j), \quad (6)$$

where $\zeta_j^k(t) \geq 0$ is the injected demand for resources of type k at location j at time t, and $\dot{s}_{ij}^k$ is the rate at which robot i is supplying resources of type k to demand j. Here, $\dot{s}_{ij}^k$ is a function of d_j^k and μ_j.

We then model the supply dynamics $\dot{s}_i^k$ as the rate at which robot i's resources of type k depletes:

$$\dot{s}_i^k = -\sum_{j|i\in\mathcal{P}^j} \dot{s}_{ij}^k(d_j^k, \mu_j). \quad (7)$$

Note that $\dot{s}_{ij}^k$ is a function that we design. In summary, $\dot{s}_i^k$ is the rate at which robot i supplies resources of type k to the demand(s) j such that robot i belongs to partition(s) $\mathcal{P}^j$. Indeed, this implies that one robot may serve multiple demands at one time. Such a feature may be desirable, for example, when two demands μ_1 and μ_2 are close together, and robot i can supply resources to both demands by situating itself in between them. However, when this is not the case, and perhaps p_i is closer to μ_1 than μ_2, then one would expect $\dot{s}_{i1}^k >> \dot{s}_{i2}^k$. One can design $\dot{s}_{ij}^k$ such that, in this example, $\dot{s}_{i2}^k$ would be negligible compared to $\dot{s}_{i1}^k$. We propose such a formulation in Section III-B.

III. Serving Dynamic Demand

We consider a dynamic demand in the following way: 1) demand depletes as robots supply resources to that demand, 2) demand can be injected, in that d_j^k may increase at any time, and 3) demand locations μ_j can move, i.e. $\mu_j = \mu_j(t)$. In this section, we first show in Lemma 1 that with such a dynamic demand, our robots will still be able to supply resources to demand locations, while accounting for multiple resource types and capacities. Then, in Theorem 1, we propose supply-and-demand dynamics, showing that, under certain conditions, robots can drive the demand to a steady state across the environment.

A. Deploying Robots to Heterogeneous Dynamic Demand

In our first work [1], we considered only static s_i^k and d_j^k. When we introduce depletion dynamics, both s_i^k, d_j^k, and μ_j change with time. Therefore, we wish to show that our prior results [1, Lemma 1] and [1, Proposition 1] still

hold. In other words, as shown in Lemma 1, the policy we proposed in [1] still minimizes the locational cost and drives the robots to critical points of their individual costs J_i. We then leverage these properties to demonstrate we can meet the dynamic demand in Section III-B.

Lemma 1: Consider a group of N robots obeying single integrator dynamics (4), with u_i given by (5) for all i and the total locational cost J of the team defined by (3). Let the supply and demand dynamics be defined by $\dot{s}_i^k$ and $\dot{d}_j^k$, respectively, and let the demand locations have dynamics $\dot{\mu}_j$. Then 1) the robots minimize J in a decentralized fashion, and 2) the robots drive to a critical point of the cost J.

Proof: In [1, Lemma 1], we showed that our cost function minimizes the cost J in (3) with static s_i^k, d_j^k, and μ_j by allowing each robot to solve

$$\begin{aligned} \min_{u_i,\delta_i} \quad & \|u_i\|^2 + |\delta_i|^2 \\ \text{s.t.} \quad & -\frac{\partial J_i}{\partial p_i}u_i \geq -\alpha(-J_i(p)) + \frac{\partial J_i}{\partial t} - \delta_i. \end{aligned}$$

Introducing the dynamics $\dot{d}_j^k$, $\dot{s}_i^k$, and $\dot{\mu}_j$ affects $\frac{\partial J_i}{\partial t}$. However, since J is not a direct function of d_j^k, s_i^k, or μ_j, the value of $\frac{\partial J_i}{\partial t}$ remains to be [1]

$$\frac{\partial J_i^j}{\partial t} = \sum_{i\in\mathcal{R}}\sum_{j\in\mathcal{D}} \frac{1}{2}\frac{\partial m_i^j}{\partial t}\|p_i - c_i^j\|^2 + m_i^j(p_i - c_i^j)^\mathsf{T}(\dot{p}_i - \frac{\partial c_i^j}{\partial t}).$$

Therefore, since introducing the dynamics $\dot{d}_j^k$, $\dot{s}_i^k$, and $\dot{\mu}_j$ does not change the original control policy (5), then by [18], our control policy (5) still minimizes the total cost J, proving [1, Lemma 1].

Similarly, since [1, Proposition 1] relies on the fact that $\dot{J}_i(p_i,t) \leq \alpha(J_i(p_i,t))$, and $\dot{J}_i(p_i,t)$ is not a direct function of d_j^k, s_i^k, or μ_j, then [1, Proposition 1] holds, and the robots drive to a critical point of the cost J. ∎

Lemma 1 shows that, with the introduction of depletion dynamics and moving demand locations, the properties of our control policy proposed in [1] still hold. Namely, we minimize the locational cost, and the robots are driven to critical points of their individual locational cost. However, the policy proposed in [1] does not guarantee that we meet a dynamic demand.

In order to meet dynamic demand, we need to update our control policy to account for $\dot{d}_j^k$, $\dot{s}_i^k$, and $\dot{\mu}_j$. As stated in the proof of Lemma 1, these terms only appear in $\frac{\partial \phi_j}{\partial t}$. Therefore, to guarantee that our control policy proposed in [1] meets dynamic demand, we need to update $\frac{\partial \phi_j}{\partial t}$ as follows:

$$\begin{aligned} \frac{\partial \phi_j}{\partial t} =& \frac{D_j}{S_j}\left(\frac{2}{\sigma}(q-\mu_j)\Sigma^{-1}\dot{\mu}_j + \frac{D_j}{\sum_{k\in\mathcal{S}} d_j^k}\right)\sum_{k\in\mathcal{S}}\dot{d}_j^k \\ & - \frac{D_j}{(S_j)^2}\sum_{l\in\mathcal{P}^j}\Big(\frac{2\sum_{k\in\mathcal{S}} s_l^k}{\sigma}(q-p_l)^\mathsf{T}\Sigma^{-1}\dot{p}_l \\ & + \sum_{k\in\mathcal{S}}\dot{s}_l^k\Big)\sum_{k\in\mathcal{S}} s_l^k \exp\left[-\frac{1}{\sigma}(q-p_l)^\mathsf{T}\Sigma^{-1}(q-p_l)\right]. \end{aligned}$$

Then, $\frac{\partial \phi_j}{\partial t}$ is used to compute $\frac{\partial m_i^j}{\partial t}$ and $\frac{\partial c_i^j}{\partial t}$, which are required for the optimal input (5).

B. Meeting Dynamic Demand

The prior section outlines how our control policy (5) accounts for dynamic resource demand, but does not guarantee the demand is met at all locations within the environment over time. We are now ready to introduce our model of resource depletion and injection within the system. From this model, we derive the formulation that allows the robots to meet the resource demand.

Recall the demand dynamics (6) and supply dynamics (7) are both functions of $\dot{s}_{ij}^k$, the rate at which robot i supplies resource type k to demand j. Recall also that $\dot{s}_{ij}^k$ is designed based on user preferences. While design choices might vary between systems, we propose a formulation that allows users to design the resource depletion rates based on the relative locations of the robots. Our chosen formulation allows us to prove that the system is input-to-state stable (Theorem 1), implying the robots can perpetually service the demand. We propose the following resource supply depletion, $\dot{s}_{ij}^k$:

$$\dot{s}_{ij}^k = \frac{m_{ij}^k}{\kappa\|p_i - c_i^j\|^2 + \epsilon}, \tag{8}$$

where κ and ϵ are positive constants, c_i^j is the centroid of the robot cell V_i^j, and m_{ij}^k is the mass corresponding to a particular resource within V_i^j,

$$m_{ij}^k = \int_{V_i^j} \frac{d_j^k \exp\left[-\frac{1}{\sigma}(q-\mu_j)^\mathsf{T}\Sigma^{-1}(q-\mu_j)\right]}{\sum_{l\in\mathcal{P}^j} s_l^k \exp\left[-\frac{1}{\sigma}(q-p_l)^\mathsf{T}\Sigma^{-1}(q-p_l)\right]}dq.$$

Intuitively, robot i is responsible for supplying resources to the demand in its cell V_i^j – if robot i belongs to more than one $\mathcal{P}^j$, then robot i serves multiple demands. With this formulation, the user can adjust κ accordingly. Effectively, the user can design the depletion/flow rate themselves based on the robot proximity to their centroids. If they want robots to serve demand on their way to demand, they can have a low κ. Alternatively, if they want robots to only serve demand when they reach, or get close to, their centroids, then they can set κ to be large. They can also have a time-varying κ, but we consider a static κ within this work.

Note that although robots with smaller cells will more easily serve demands, as they will be closer to each of its centroids, (8) still achieves the desired depletion effects. For example, if p_i is far away from demand peak μ_1, and not close to c_i^1, then m_{ij}^k will be very small, even though the denominator of (8) may be relatively small. Therefore, $\dot{s}_{ij}^k$ should still be small. If the robot p_i is in between multiple peaks μ_j's, but closer to c_i^1, then robot i may serve both demands, but it will serve demand $j=1$ faster.

Before we can claim that the robots will drive the demand to a steady state, we first need to make the assumption that the demand μ_j cannot move faster than the robots. Assumption 1 formalizes this requirement.

Assumption 1: The peak of the demand location cannot move faster than the robots, i.e., $\dot{p}_i \geq \dot{\mu}_j,\ \forall\ i \in \{1,...,N\},\ \forall\ j \in \{1,...,M\}$.

With demand dynamics (6), supply dynamics (8), and Assumption 1, we can now show that the robots will drive the total demand to a steady state, as formalized in Theorem 1. Our analysis focuses on the stability under injected demand ζ_j^k, which is the demand added over time to the system. In fact, treating ζ_j^k as the input to the system $\dot{d}_j^k$, we show that the system (6) is input-to-state stable (ISS). In other words, we show that there exists a class $\mathcal{KL}$ function β and a class $\mathcal{K}$ function γ such that for $t \geq t_0$

$$d_j^k(t) \leq \beta(d_j^k(t_0), t - t_0) + \gamma\left(\sup_{t_0 \leq \tau \leq t} |\zeta_j^k(\tau)|\right),$$

for bounded input $\zeta_j^k(t)$. Thus, d_j^k will be bounded by γ for bounded input ζ_j^k.

Theorem 1: Consider a team of N robots with supply dynamics (7) serving M locations with demand dynamics (6), with $\dot{s}_{ij}^k$ given by (8). Then, the origin $d_j^k = 0$ is ISS with

$$\gamma(r) = \frac{r}{\theta \sum_{i \in \mathcal{P}^j} \frac{\mathcal{M}_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon}}, \tag{9}$$

where $0 < \theta < 1$ and

$$\mathcal{M}_{ij}^k = \int_{V_i^j} \frac{\exp\left[-\frac{1}{\sigma}(q - \mu_j)^\mathsf{T} \Sigma^{-1} (q - \mu_j)\right]}{\sum_{l \in \mathcal{P}^j} s_l^k \exp\left[-\frac{1}{\sigma}(q - p_l)^\mathsf{T} \Sigma^{-1} (q - p_l)\right]} dq.$$

Proof: We begin by noting that when the input $\zeta_j^k = 0$, the system (6) has a globally asymptotically stable equilibrium point at the origin $d_j^k = 0$. Let $V(d_j^k) = \frac{1}{2}(d_j^k)^2$ be a Lyapunov function candidate. The time derivative of V along the trajectories of the system is

$$\begin{aligned}\dot{V}(d_j^k) =& d_j^k \zeta_j^k(t) - d_j^k \sum_{i \in \mathcal{P}^j} \dot{s}_{ij}^k \\ =& d_j^k \zeta_j^k(t) - d_j^k \sum_{i \in \mathcal{P}^j} \frac{m_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon}.\end{aligned}$$

Note d_j^k is constant across m_{ij}^k, therefore

$$\dot{V}(d_j^k) = d_j^k \zeta_j^k(t) - (d_j^k)^2 \sum_{i \in \mathcal{P}^j} \frac{\mathcal{M}_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon}.$$

Then, we can proceed as follows:

$$\begin{aligned}\dot{V}(d_j^k) =& d_j^k \zeta_j^k(t) - (1 - \theta)(d_j^k)^2 \sum_{i \in \mathcal{P}^j} \frac{\mathcal{M}_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon} \\ & - \theta (d_j^k)^2 \sum_{i \in \mathcal{P}^j} \frac{\mathcal{M}_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon} \\ \leq & - (1 - \theta)(d_j^k)^2 \sum_{i \in \mathcal{P}^j} \frac{\mathcal{M}_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon} \\ & \forall d_j^k \geq \rho(\zeta_j^k),\end{aligned} \tag{10}$$

where $0 < \theta < 1$ and

$$\rho(\zeta_j^k) = \frac{\zeta_j^k(t)}{\theta \sum_{i \in \mathcal{P}^j} \frac{\mathcal{M}_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon}}. \tag{11}$$

Since $d_j^k \geq 0 \ \forall \ t \geq 0$, $\rho'(\zeta_j^k) > 0 \ \forall t \geq 0$, and thus $\rho(\zeta_j^k)$ is a class $\mathcal{K}$ function. Therefore, by Theorem 4.19 in [19], the system (6) is ISS, with $\gamma(r)$ given by (9), which completes the proof. ∎

Theorem 1 shows that when the injected demand ζ_j^k is bounded, the demand d_j^k will be bounded by γ, thus the demand d_j^k will reach a steady state. Note also that, although we provide a proof specifically for our proposed dynamics in (8), a similar proof can be carried out for alternative formulations for $\dot{s}_{ij}^k$. We can go beyond Theorem 1 to define the limits of ζ_j^k to maintain input-to-state stability, formalized in Corollary 1.

Corollary 1: With the same assumptions as Theorem 1, the origin $d_j^k = 0$ is ISS when

$$\zeta_j^k(t) \leq \theta \sum_{i \in \mathcal{P}^j} \frac{m_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon}. \tag{12}$$

Proof: The proof follows from the inequality (10). Multiplying both sides by the denominator in (11), we attain

$$\zeta_j^k(t) \leq d_j^k \theta \sum_{i \in \mathcal{P}^j} \frac{\mathcal{M}_{ij}^k}{\kappa \|p_i - c_i^j\|^2 + \epsilon}. \tag{13}$$

Since $d_j^k \sum_{i \in \mathcal{P}^j} \mathcal{M}_{ij}^k \equiv m_{ij}^k$, then (13) simplifies to (12) thus completing the proof. ∎

By Corollary 1, we can conclude that when the input ζ_j^k is bounded as in (12), the demand of a resource d_j^k at a peak reaches a steady state. In other words, when the injected demand is bounded, the robots keep the demand within a threshold defined by γ (9).

Recall that our supply and demand dynamics are functions of robot positions p_i and demand locations μ_j; specifically, the rate at which the robots serve demand increases as the robot positions near the demand locations. Our results demonstrate that, with our proposed formulation (8), the robots can deliver supplies that meet the resource demand over various locations of the environment and over time given bounded input (12).

C. Algorithm Overview

Algorithm 1 summarizes how the supply robots deploy to meet the dynamic resource demand. We first retrieve robot positions, supply quantities, demand quantities, demand locations, and demand injection as inputs. We then deploy resource robots as follows until we meet the demand. At each iteration, we obtain the demand location dynamics and demand injection, then assign robots to partitions and update each Voronoi tessellation. From here, the robots can compute their control inputs u_i. Note that when computing the control input using the optimization problem 5, it is possible for the robots to travel outside of environment Q due to a lack of constraints. We therefore check (Line 8) that if the input u_i causes the robot to travel outside the convex environment Q, the corresponding x-/y- component of u_i will be set to 0. In our simulations, we update the robot positions as in Line 7. In experiments, however, robots simply execute u_i. Then, we update s_i^k, d_j^k, and μ_j, repeating the process until the demand is met.

Algorithm 1 Coverage of Dynamic Demand

Input $t = t_0$, $p_i(t_0)$, $s_i^k(t_0)$, $d_j^k(t_0)$, $\mu_j(t_0)$, $\zeta_j^k(t_0)$, $\forall i, j, k$
while $d_j^k(t) > 0 \; \forall j, k$ **do**
  Input $\dot{\mu}_j(t)$ and $\zeta_j^k(t)$
  Assign robots to partitions $\mathcal{P}^j$ (1)
  Update tessellation V_i^j (2)
  Compute control input u_i (5) $\forall i \in \mathcal{R}$
  Update robot position:
  $p_i(t + \Delta t) \leftarrow p_i(t) + u_i(t)\Delta t$
  if $p_i(t + \Delta t) \notin Q$ **then**
    Set the respective x/y component to zero: $u_i \leftarrow 0$
    Recompute positions: $p_i(t+\Delta t) \leftarrow p_i(t)+u_i(t)\Delta t$
  end if
  Update resources:
  $s_i^k(t + \Delta t) \leftarrow s_i^k(t) + \dot{s}_i^k(t)\Delta t$
  $d_j^k(t + \Delta t) \leftarrow d_j^k(t) + \dot{d}_j^k(t)\Delta t$
  Update demand locations:
  $\mu_j(t + \Delta t) \leftarrow \mu_j(t) + \dot{\mu}_j(t)\Delta t$
  $t \leftarrow t + \Delta t$
end while

IV. Validation

Here, we present a series of simulations and experiments to evaluate our proposed approach for serving dynamic demand. Recall the formulation for $\dot{s}_{ij}^k$ (8). As robots approach their respective centroids, the value of $\dot{s}_{ij}^k$ becomes exponentially large. In practice, robots releasing resources at such a high rate is unrealistic. Therefore, in our simulations and experiments, we place a cap on $\dot{s}_{ij}^k$ such that

$$\dot{s}_{ij}^k = \min\left\{\dot{s}_{ij}^k, \dot{s}_{ij,\max}^k\right\},$$

where $\dot{s}_{ij,\max}^k$ is the maximum rate at which robot i can supply resources of type k to demand j. Additionally, to effectively evaluate the ability of robots to perpetually serve demand, we initialize the robot team with enough resources to be able to serve demand until Algorithm 1 is terminated.

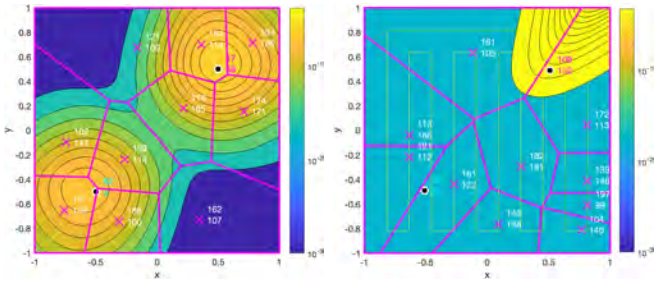
To evaluate the performance of our robot teams, we define the Demand Index (DI) as the percentage of the initial demand:

$$\mathrm{DI}(t) = \frac{\sum_{j\in\mathcal{D}}\sum_{k\in\mathcal{S}} d_j^k(t)}{\sum_{j\in\mathcal{D}}\sum_{k\in\mathcal{S}} d_j^k(t_0)}.$$

A lower DI implies a greater performance, and a DI less than one implies that the team has reduced the overall demand. We utilize this performance metric to evaluate the team in both simulations and experiments. We also introduce the performance metric $\int_{t_0}^{t_f} \mathrm{DI}(t)dt$ as an indication of the speed at which the robots serve demand from time t_0 to the time at which the algorithm is terminated t_f.

A. Simulations

In this section, we demonstrate our approach for serving a dynamic demand and compare our results with a simple persistent monitoring approach to further motivate a coverage control approach to resource allocation. Specifically, we compare our approach with a lawnmower algorithm, giving the robots a predefined trajectory to follow at maximum velocity. Figure 3(a) shows an example of our coverage approach in simulation, and Figure 3(b) shows an example of the lawnmower approach, with the lawnmower trajectory indicated by yellow lines. We study two scenarios, considering two resource types ($k = \{1, 2\}$) with the following randomized team parameters: $N \in [6, 10]$, $s_i^k \in [100, 200]$, and $p_i \in Q$. We initialized the demand locations at $\mu_1 = (-0.5, -0.5)$ and $\mu_2 = (0.5, 0.5)$, and demand quantities at $d_1^k = \{100, 0\}$ and $d_2^k = \{50, 100\}$. We ran 50 trials of each scenario. In each trial, both the demand locations and demand quantities are static for time $t \in [t_0, t_1)$. For time $t \geq t_1$, the demand moves in a counterclockwise circular path at a constant speed about the origin. In our simulations, we let $t_1 = 22.5$s. We ensure that $\dot{p}_{i,\max} > \dot{\mu}_j$ to satisfy Assumption 1.

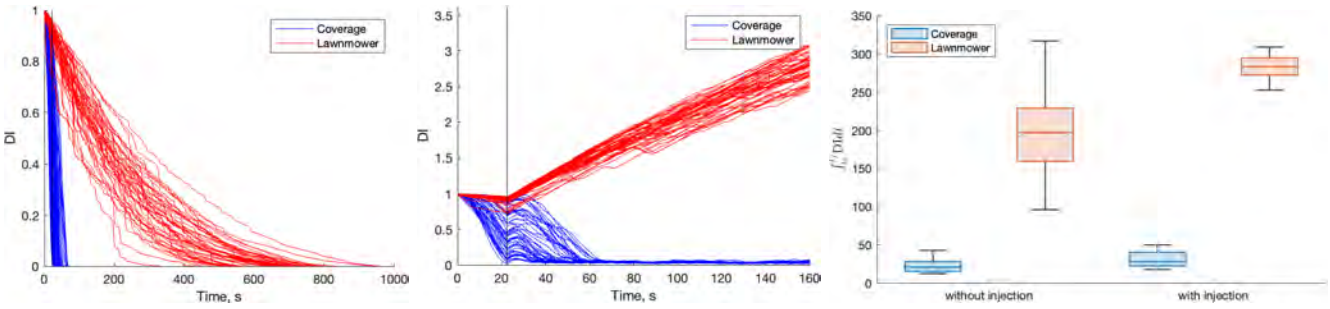


(a) Coverage. (b) Lawnmower.

Fig. 3. Screenshots of example randomized simulations. Both subfigures show configurations from the same initial conditions. In (a), we show our approach. Robots are represented by $\times$'s, and the demand locations μ_j are represented by black $\circ$'s. Magenta lines signify Voronoi boundaries, and the contour lines represent the total density. In (b), robots are following a lawnmower algorithm. Robots follow the solid yellow lines in a clockwise direction. Although the Voronoi diagram and density does not play into the robot controls in the lawnmower algorithm, we still show them as they are required to compute resource dynamics.

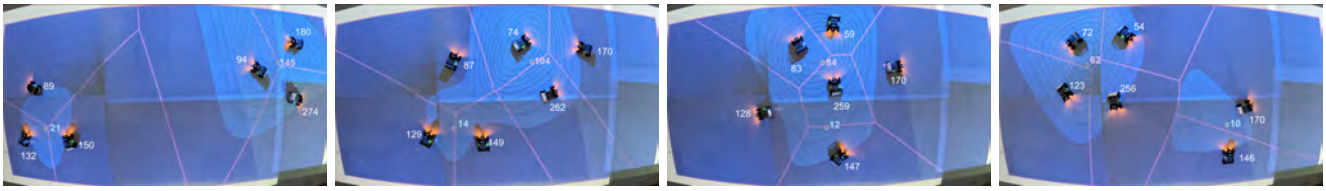
1) Moving demand without injection: In the first scenario, we consider the case where $\zeta_j^k = 0$ for the entire trial, i.e. we allow the demand to move, but we do not inject demand. We allow simulations to run until all demand is completely depleted, i.e. $\sum_{j\in\mathcal{D}}\sum_{k\in\mathcal{S}} d_j^k(t) \equiv 0$. Results are shown in Figure 4(a). Since there is no injected demand, the team is always able to drive the demand to zero. We observe from Figures 4(a) and 4(c) that although both approaches are able to meet demand, our approach depletes the demand at a much faster rate than the lawnmower algorithm.

2) Moving demand with injection: In this scenario, we increase the complexity of the task even more by not only moving demand but also injecting demand into the system at time $t \geq t_1$ ($\dot{\mu}_j = 0$ and $\zeta_j^k = 0$ for $t_0 \leq t < t_1$, and $\dot{\mu}_j \neq 0$ and $\zeta_j^k > 0$ for $t \geq t_1$). Specifically, at time $t \geq t_1$, we inject demand as follows: $\dot{d}_1^k(t) = (8\sin(2t), 10\sin(3t))$, $\dot{d}_2^k(t) = (3, 2)$. Results are shown in Figures 4(b) and 4(c). Since the lawnmower algorithm is not able to drive the demand to zero once $\zeta_j^k > 0$, we terminate simulations at time $t = 160$s. We can see from these results that, in all 50 trials, our approach is not only able to maintain



(a) Without injected demand. (b) With injected demand. (c) Boxplots.

Fig. 4. Simulation results. (a) and (b) show the DI over time for 50 trials of each case study, comparing our approach (blue) and a lawnmower algorithm (red). The black vertical line at time $t = 22.5$s represents the time at which demand changes. In (a), demand locations move but demand is not injected. In (b), demand locations move and demand is injected into the system. (c) shows boxplots of the integral of DI over the entire trial, with 50 trials making up each box. In the left two boxes, demand locations move but demand is not injected. In the right two boxes, demand locations move, and we inject demand into the system.



(a) $t = 63$s (b) $t = 123$s (c) $t = 151$s (d) $t = 197$s

Fig. 5. Time series of Experiment 1. Here we consider six robots, each equipped with one type of resource, serving two demand locations indicated by black circles. The amount of resources that each robot carries is printed to the right of each robot, and the amount of resources demanded at each demand location is printed to the right of each demand location. Solid pink lines indicate the Voronoi boundaries, and the contour lines indicate the density across the environment, which peaks around the demand locations. As the robots serve demand over time, the supply and demand quantities both decrease as the robots transfer their resources to the demand.

a DI < 1, but also drive the demand to a steady state, even when $\zeta_j^k > 0$. At $t = t_1$, we see a slight increase in the DI before the robots adjust, and the DI decreases to steady state. On the other hand, a persistent monitoring approach does not provide adequate service when demand is injected into the system, as the lawnmower algorithm cannot adequately follow the dynamic demand. These results further demonstrate the benefits of implementing a coverage control approach, as coverage control enables us to continuously adapt to spatial changes in demand. Further, we demonstrate that our proposed demand dynamics (8) ensure that robots meet demand spatially.

B. Experiments

In addition to simulations, we conducted hardware-in-the-loop experiments to demonstrate our approach in real time. We implement Algorithm 1 on a team of six AgileX LIMO[1] ground robots each equipped with the NVIDIA Jetson Nano[2]. We compute control inputs u_i on a desktop computer (8-core, 32GB RAM, Windows 10) and send velocity commands to robots over Wi-Fi using Robot Operating System (ROS). Robot poses are retrieved from an Optitrack Motion Capture[3] system. We convert u_i to linear and angular velocity commands [20], respectively, by

$$v_i = k_v \begin{bmatrix} \cos\theta_i & \sin\theta_i \end{bmatrix} u_i,$$
$$\omega_i = k_\omega \arctan\left(\frac{\begin{bmatrix} -\sin\theta_i & \cos\theta_i \end{bmatrix} u_i}{\begin{bmatrix} \cos\theta_i & \sin\theta_i \end{bmatrix} u_i} \right),$$

where θ_i is the robot heading angle with respect to the global frame, and k_v and k_ω are positive gains.

We present two experiments, with a time series of Experiment 1 shown in Figure 5 and a screenshot of Experiment 2 in Figure 1 (for full experiments, see supplemental video). In both cases, demand locations are initialized to $\mu_1 = (2.0, 1.0)$ and $\mu_2 = (7.0, 2.0)$ with no demand injection. At time $t = t_1$, demand is injected as specified below, and demand moves at a constant speed in a straight line between the positions $x = 2.0$ and $x = 7.0$, with y-positions remaining constant. Once demand reaches either x-position, the demand moves toward the other x-position. We ran experiments until the demand appeared to reach a steady state. Results for the team performance over time is shown in Figure 6, where we can see that the team is able to serve the demand.

1) Experiment 1: In the first experiment, we consider only one resource type ($k = \{1\}$) with the two demand locations. Here, we initialize demand at $d_1^k = \{50\}$ and $d_2^k = \{200\}$, and robot supplies at $s_i^k = \{100, 100, 200, 150, 300, 150\}$. At time $t = 57$s, the demand locations begin to move as mentioned above, and we inject demand at rates $\dot{d}_1^1 =$

[1]https://global.agilex.ai/products/limo
[2]https://developer.nvidia.com/embedded/jetson-nano-developer-kit
[3]https://optitrack.com/

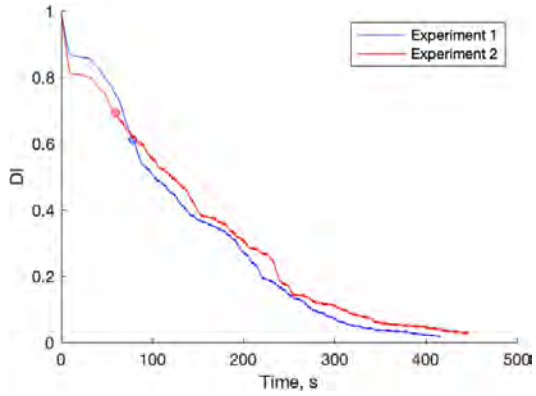


Fig. 6. DI over time for the two experimental trials. The circles on each line represent the time at which demand locations started moving, and demand was injected into the system.

$4\sin(2t)$ and $\dot{d}_2^1 = 8\cos(3t)$.

Figure 5 shows a time series of this experiment, with μ_1 and μ_2 starting in the bottom left and top right corners of the environment, respectively. Over time, we see the demand move from their initial positions to opposite sides of the space: μ_1 moves to the bottom right corner, and μ_2 moves to the top left corner. As the demand locations move and vertically align with one another, we see the robots rearrange themselves to better serve the dynamic demand. Specifically, in Figure 5(a), the robot with 89 resources serves μ_1, but when the two demand peaks cross paths (Figures 5(b)-5(c)), this robot moves to serve μ_2 which demands more resources.

2) Experiment 2: In the second experiment, we account for two resource types ($k = \{1, 2\}$) with the same two demand locations. We initialize demand to be $d_1^k = \{200, 0\}$ and $d_2^k = \{100, 200\}$ and robot resources to be $s_i^1 = \{150, 200, 100, 200, 0, 200\}$ and $s_i^2 = \{100, 100, 200, 150, 300, 150\}$. At time $t = 60$s, the demand locations move with the same dynamics as that of Experiment 1, and we inject demand at rates $\dot{d}_1^k = \{4\sin(2t), 8\sin(3t)\}$ and $\dot{d}_2^k = \{8\cos(3t), 4\cos(2t)\}$. We observe similar results to that of Experiment 1, with the robots taking a slightly longer time to serve demand given the greater number of resources demanded.

V. Conclusions

In this paper, we consider a heterogeneous multi-robot team, equipped with different types and quantities of resources, and tasked with supplying these resources to various demand locations. We build upon our previous work [1] to enable adaptation to changes in demand. Specifically, we allow resources to deplete from both robots and from demand. We also allow demand locations to move, and for demand to be injected into the system. We use a Voronoi-based coverage control approach to enable robots to serve demand perpetually and in a continuous, distributed fashion. Simulations and experiments demonstrate the effectiveness of our approach, particularly compared to a persistent monitoring approach.

References

[1] M. Coffey and A. Pierson, "Heterogeneous Coverage and Multi-Resource Allocation in Supply-Constrained Teams," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[2] M. Santos, Y. Diaz-Mercado, and M. Egerstedt, "Coverage Control for Multirobot Teams With Heterogeneous Sensing Capabilities," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 919–925, 2018.

[3] M. Santos and M. Egerstedt, "Coverage Control for Multi-Robot Teams with Heterogeneous Sensing Capabilities Using Limited Communications," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 5313–5319.

[4] A. Sadeghi and S. L. Smith, "Coverage Control for Multiple Event Types with Heterogeneous Robots," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 3377–3383.

[5] A. Pierson, L. C. Figueiredo, L. C. A. Pimenta, and M. Schwager, "Adapting to sensing and actuation variations in multi-robot coverage," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 337–354, 2017.

[6] S. Kim, M. Santos, L. Guerrero-Bonilla, A. Yezzi, and M. Egerstedt, "Coverage Control of Mobile Robots With Different Maximum Speeds for Time-Sensitive Applications," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3001–3007, 2022.

[7] S. Choudhury, J. K. Gupta, M. J. Kochenderfer, D. Sadigh, and J. Bohg, "Dynamic multi-robot task allocation under uncertainty and temporal constraints," *Autonomous Robots*, vol. 46, no. 1, pp. 231–247, 1 2022.

[8] Z. Li, A. Vatankhah Barenji, J. Jiang, R. Zhong, and G. Xu, "A mechanism for scheduling multi robot intelligent warehouse system face with dynamic demand," *Journal of Intelligent Manufacturing*, vol. 31, 6 2020.

[9] J. A. Shaffer, E. Carrillo, and H. Xu, "Hierarchal Application of Receding Horizon Synthesis and Dynamic Allocation for UAVs Fighting Fires," *IEEE Access*, vol. 6, pp. 78 868–78 880, 2018.

[10] T. Diao, S. Singla, A. Mukhopadhyay, A. Eldawy, R. Shachter, and M. Kochenderfer, "Uncertainty Aware Wildfire Management," 11 2020.

[11] J. D. Griffith, M. J. Kochenderfer, R. J. Moss, V. V. Mišić, V. Gupta, and D. Bertsimas, "Automated Dynamic Resource Allocation for Wildfire Suppression," *Lincoln Laboratory Journal*, vol. 22, no. 2, pp. 38–59, 2017.

[12] T. Sakamoto, S. Bonardi, and T. Kubota, "A Routing Framework for Heterogeneous Multi-Robot Teams in Exploration Tasks," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6662–6669, 2020.

[13] S. Narayan, J. Paulos, S. W. Chen, S. Manjanna, and V. Kumar, "Combined Routing and Scheduling of Heterogeneous Transport and Service Agents," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 1466–1472.

[14] G. S. Oliveira, J. Röning, P. D. M. Plentz, and J. T. Carvalho, "Efficient Task Allocation in Smart Warehouses with Multi-delivery Stations and Heterogeneous Robots," 2 2022.

[15] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[16] S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[17] G. Notomista and M. Egerstedt, "Constraint-Driven Coordinated Control of Multi-Robot Systems," in *2019 American Control Conference (ACC)*, 2019, pp. 1990–1996.

[18] M. Santos, S. Mayya, G. Notomista, and M. Egerstedt, "Decentralized Minimum-Energy Coverage Control for Time-Varying Density Functions," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, 2019, pp. 155–161.

[19] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ: Prentice Hall, 2002.

[20] Y. Diaz-Mercado, S. G. Lee, and M. Egerstedt, "Human-swarm interactions via coverage of time-varying densities," in *Trends in Control and Decision-Making for Human-Robot Collaboration Systems*. Springer International Publishing, 1 2017, pp. 357–385.