The Impact of Diversity on Optimal Control Policies for Heterogeneous Robot Swarms

Amanda Prorok, M. Ani Hsieh, and Vijay Kumar

***Abstract*—We consider the problem of distributing a large group of heterogeneous robots among a set of tasks that require specialized capabilities in order to be completed. We model the system of heterogeneous robots as a community of species, in which each species (robot type) is defined by the traits (capabilities) that it owns. In order to solve the distribution problem, we develop centralized as well as decentralized methods to efficiently control the heterogeneous swarm of robots. Our methods assume knowledge of the underlying task topology and are based on a continuous model of the system that defines transition rates to and from tasks, for each robot species. Our optimization of the transition rates is fully scalable with respect to the number of robots, number of species, and number of traits. Building on this result, we propose a real-time optimization method that enables an online adaptation of transition rates as a function of the state of the current robot distribution. We also show how the robot distribution can be approximated based on local information only, consequently enabling the development of a decentralized controller. We evaluate our methods by means of microscopic simulations and show how the performance of the latter is well predicted by the macroscopic equations. Importantly, our framework also includes a diversity metric that enables an evaluation of the impact of swarm heterogeneity on performance. The metric defines the notion of *minspecies*, i.e., the minimum set of species that are required to achieve a given goal. We show that two distinct goal functions lead to two specializations of minspecies, which we term as *eigenspecies* and *coverspecies*. Quantitative results show the relation between diversity and performance.**

***Index Terms*—Heterogeneous multirobot systems, stochastic systems, swarm robotics, task allocation.**

I. Introduction

TECHNOLOGICAL advances in embedded systems, such as component miniaturization and improved efficiency of sensors and actuators, are enabling the deployment of very large-scale robot systems, i.e., robot swarms [1], [2]. However, as we aspire to solve increasingly complex problems, it becomes ever more difficult to embed all necessary capabilities into one single robot type. Our premise is that one single type of robot cannot cater to all aspects of the task at hand, because at the individual level, it is governed by design rules that limit the scope of its capabilities. For example, a larger robot may be able to carry more powerful sensors, but may be less agile than its smaller counterpart. Or, we could consider the limited payload of aerial robots: if a given task requires rich sensory feedback, multiple heterogeneous aerial robots can complement each other by carrying distinct sensors. Instances of information gathering lend themselves naturally to this problem formulation, with applications to search, surveillance, environmental monitoring, and situational awareness [3]–[5].

As we allocate distinct capabilities among robot team members, we imply a certain degree of specialization among individuals [6]–[8]. During this process, heterogeneity becomes a design feature. The question is, then, how to best design such systems so that the resulting performance is optimized [9]. However, since there has been very little work on quantitative measures of diversity in multirobot systems, we still lack the analytical tools to understand the implications.

In this paper, we contribute toward a general understanding of heterogeneity by proposing a measure that quantifies the diversity of a swarm of robots that is tasked to complete a goal. Our hypothesis is that the diversity measure must be tied to the underlying goal function for it to be meaningful. In particular, we show how for two different goals, we need two different diversity measures. Toward this end, we consider a concrete application with the objective of distributing a swarm of heterogeneous robots as efficiently as possible among tasks that require specialized competences. Our methodology enables the formulation of control policies that take the heterogeneity of the swarm into account explicitly, and that are capable of adapting to changes online.

A. Example of the Redistribution Problem

Fig. 1 shows a system comprising ten tasks that can be serviced by means of four distinct traits. The initial trait distribution is shown at t_0, and subsequent desired trait distributions are shown at t_1, t_2, and t_3. Fig. 2 shows how the distribution of the traits evolves over time. This sequence is an example of how a heterogeneous robot system can be controlled to complete a global goal composed of several subtasks that require a specific set of capabilities in specific amounts. Also, the example shows how the solution to the redistribution problem can incorporate

Manuscript received February 10, 2016; revised June 29, 2016 and September 12, 2016; accepted October 19, 2016. Date of publication January 16, 2017; date of current version April 3, 2017. This paper was recommended for publication by Associate Editor L. Pallottino and Editor T. Murphey upon evaluation of the reviewers' comments. This work was supported in part by the ONR under Grant N00014-15-1-2115, Grant N00014-14-1-0510, and Grant N00014-13-1-0731; in part by ARL under Grant W911NF-08-2-0004; in part by the NSF under Grant IIS-1426840; and in part by TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

A. Prorok and V. Kumar are with University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: prorok@seas.upenn.edu; kumar@seas.upenn.edu).

M. A. Hsieh is with Drexel University, Philadelphia, PA 19104 USA (e-mail: mhsieh1@drexel.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2016.2631593

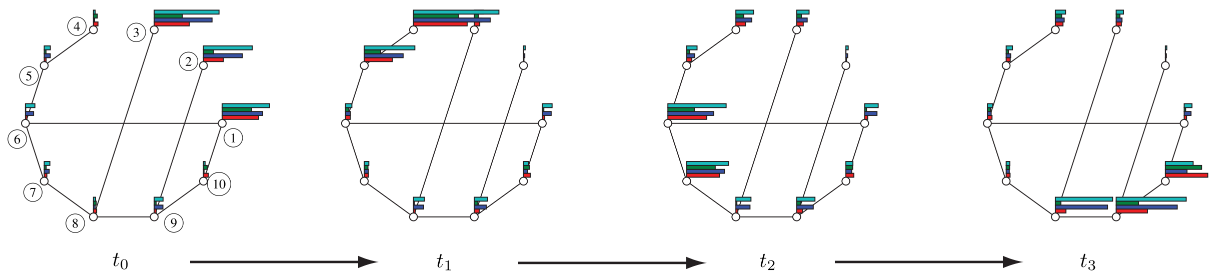


Fig. 1. Four configurations of a system with ten tasks (nodes) and four traits. The trait abundance per task is represented by a bar plot. The edges of this strongly connected graph represent the possibility of switching between a pair of tasks. The system's initial distribution is shown at t_0, with subsequent desired target distributions at $t_{[1,2,3]}$.

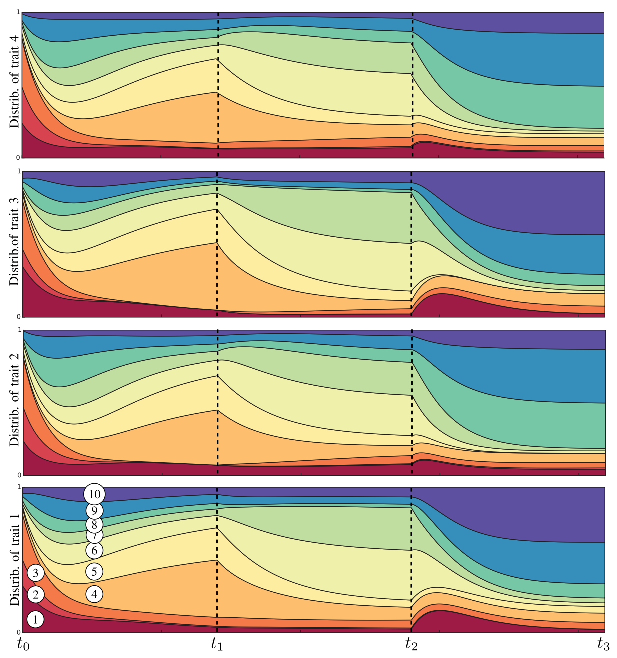


Fig. 2. Evolution over time of the trait distribution as specified by the distributions shown in Fig. 1. Each subplot represents one trait, indicating the distribution of that trait over the set of tasks (for each subplot, task 1 is shown at the bottom and task 5 at the top). The system's initial distribution is shown at t_0, with subsequent desired target distributions reached at $t_{[1,2,3]}$.

temporal constraints or precedence constraints: at an operator level, we can define arbitrary rules that govern transitions from one desired trait distribution to another as a function of the system's performance. As an example application, we could consider a spatially distributed information gathering scenario, with tasks represented by geographically anchored sites: if enough data have been gathered at one site, a central operator can reconfigure the subsequent desired trait distribution so that robots distribute to sites that have not yet been sufficiently accounted for. Which robots are deployed to which sites will depend on their capabilities and how these capabilities meet the needs that are anticipated at the sites. We note that the communication infrastructure required for such an approach is asymmetric: a centralized operator gathers abstract state information about the robot swarm, and relays control inputs back to the robots. This approach ensures algorithmic invariance as we scale the system [10].

B. Background

Given a set of tasks, and knowledge about the task requirements, our problem considers which robots should be allocated to which tasks. This problem is an instance of the *MT-MR-TA: multitask robots, multirobot tasks* problem [11], and can be reformulated as a set-covering problem that stems from combinatorial optimization. This problem is strongly NP-hard [12]. A greedy algorithm to solve this problem was proposed in [13], and later adapted for use in distributed multiagent systems [14]. In the latter approach, the robots must compute all possible "coalitions" (groupings to solve a specific task), and agree on the best ones. Hence, this algorithm is best applied when the space of possible coalitions is naturally limited. Market-based approaches have also been considered to solve task allocation problems [15]. However, such approaches rely on bidding mechanisms that make extensive use of communication, and hence, scale poorly as the number of robots and tasks increases (not to mention that they do not address the problem of controller synthesis). In particular, for systems that are required to adapt to changing task requirements online, we need to consider algorithms that are efficient and that run on low-cost, resource-constrained mobile platforms in real time. Hence, we consider a strategy that is scalable in the number of robots and their capabilities, and is robust to changes in the robot population [16], [17]. An important property of this strategy is its inherently decentralized architecture, with robots modeled to switch stochastically between tasks (behaviors). The key difference between our work and previous work is that we formulate our desired state as a distribution of traits among tasks, instead of specifying the desired state as a direct measure of the robot distribution. In other words, our framework allows a user to specify how much of a given capability is needed for a given task, irrespective of which robot type satisfies that need. As a consequence, we do not employ optimization methods that utilize final robot distributions in their formulations (which is the case in previous works [16] and [18]). Also, our work lies in contrast to the methods proposed in [19], which tackle the *single-task robots, multirobot tasks* problem. Our methods explicitly optimize the distribution of traits, and implicitly solve the combinatorial problem of distributing the right number of robots of a given type to the right tasks.

C. Contributions

In [20], we first presented a method that distributes a swarm of heterogeneous robots among a set of tasks that require specialized capabilities in order to be completed. Since our method is based on the derivation of an analytical gradient, it is very efficient with respect to state-of-the-art methods. The work in [21] builds on this result, proposing a real-time optimization method that enables an online adaptation of transition rates. Finally, in [22], we first propose a diversity metric, and show how the performance of the system relates to this measure. Apart from consolidating these previous publications, and assembling all major contributions into one coherent paper, the present paper provides the following contributions.

1) In our previous work, we considered a single goal, which required an *exact* match of the final trait distribution to the desired trait distribution. Here, we also consider the formulation of a goal that allows for a *minimum* match (and does not penalize superfluous traits). By considering this additional goal function and extending the scope of our work, this paper presents a unified framework that allows us to simultaneously solve the allocation problem as well as the controller synthesis problem for heterogeneous swarms.
2) We generalize the definition of our diversity metric to include multiple possible underlying goal functions. We show that our previous definition of *eigenspecies* is a subclass of the general class of *minspecies*. Furthermore, we show that our second goal function leads to an additional subclass of minspecies, and we refer to this new subclass as the *coverspecies*. We analyze the performance of our system as a function of dedicated diversity measures, one that is based on eigenspecies, and the other that is based on coverspecies.
3) We provide a decentralized implementation of our online performance optimization algorithm for robot swarms that compute the control policy locally. Simulation results show that our algorithm exhibits a graceful performance degradation in the case of increasing communication constraints.
4) We reformulate our optimization problem as a constrained optimization problem that guarantees a minimum acceptable level of performance.
5) We provide updated and new experimental results that support all novel contributions.

II. Problem Formulation

Heterogeneity and diversity are core concepts of this paper. To develop our formalism, we borrow terminology from the biodiversity literature [23]. We define our robot system as a *community* of robots. Each robot belongs to a *species*, defining the unique set of *traits* that encodes the robots' capabilities. In this paper, we consider binary trait instantiations (e.g., the absence or presence of some sensor/actuator). In practice, continuous abilities and constraints (e.g., maximum velocity, sensing range, and computing power) can be quantized and one-hot encoded. Furthermore, we assume that the tasks have been encoded through such binary characteristics that represent the skill sets critical to task completion.

A. Notation

We consider a community of S robot species, with a total number of robots N, and $N^{(s)}$ robots per species such that $\sum_{s=1}^{S} N^{(s)} = N$. The community is defined by a set of U traits, and each robot species owns a subset of these traits. A species is defined by a binary vector $\mathbf{q}^{(s)} = [q_1^{(s)}, q_2^{(s)}, \ldots, q_U^{(s)}]$. We can then define a $S \times U$ matrix $\mathbf{Q}$, with rows $\mathbf{q}^{(s)}$ as

$$\mathbf{Q}_{su} = \begin{cases} 0, \text{if species } s \text{ does not have trait } u \\ 1\ , \text{if species } s \text{ has trait } u \end{cases}.$$

We model the interconnection topology of the M tasks via a strongly connected directed graph, $\mathfrak{G} = (\mathcal{E}, \mathcal{V})$ where the set of vertices $\mathcal{V}$ represents tasks $\{1, \ldots, M\}$ and the set of edges $\mathcal{E}$ represents the ordered pairs (i, j), such that $(i, j) \in \mathcal{V} \times \mathcal{V}$, and i and j are adjacent. Edges denote the possibility for robots to switch between two adjacent tasks. We assign every edge in $\mathcal{E}$ a transition rate, $k_{ij}^{(s)} > 0$, where $k_{ij}^{(s)}$ defines the average frequency with which one robot of species s at task i switches to task j. Here, $k_{ij}^{(s)}$ is a stochastic transition rule. We impose a limitation on the maximum rate of each edge with $k_{ij}^{(s)} < k_{ij,\max}^{(s)}$.

The distribution of the robots belonging to a species s at time t is described by a vector $\mathbf{x}^{(s)}(t) = [x_1^{(s)}(t), \ldots, x_M^{(s)}(t)]^\top$, and is summarized in a $M \times S$ matrix $\mathbf{X}(t)$, which we refer to as abstract state information. Then, if $\mathbf{q}^{(s)}$ are the rows of $\mathbf{Q}$, we have the $M \times U$ matrix $\mathbf{Y}$ that describes the distribution of traits on tasks. For time t this relationship is given by

$$\mathbf{Y}(t) = \mathbf{X}(t) \cdot \mathbf{Q}. \tag{1}$$

B. Assumptions

We make the following assumptions.

1) The robots have the knowledge of the graph topology: they are either informed *a priori* how tasks are interconnected, or they are capable of estimating the graph topology during system initialization.
2) The robots have the knowledge of their current task allocation: they have perfect knowledge of their current location within the graph and can potentially broadcast this information to neighboring robots (i.e., robots completing nearby tasks).
3) The robots have the knowledge of the target trait distribution: at least one robot is connected to a remote operator who sends the new goal, and it is assumed that all robots achieve consensus on this common goal (they know the total number of robots per species).

As a specific example, we could consider a set of tasks that are physically distributed at distinct sites. In our formalism, the sites are represented by graph vertices. Hence, the robots must obtain knowledge of the paths that allow them to reach these sites (assumption 1). This can be done by exploiting maps that are built *a priori* and provided to all robots. The transition rate represents the rate with which a specific path is chosen. We assume that the maximum rate of each edge $k_{ij,\max}^{(s)}$ can be determined by applying system identification methods on the actual system. This value depends on observable factors, such as typical road congestion or the condition of the terrain. The robots have noise-free navigation capabilities (assumption 2). Finally, a centralized monitoring system can determine when the tasks have been sufficiently accounted for, and broadcasts a new desired trait distribution (assumption 3), triggering the robots to reconfigure their distribution.

C. System

The initial state of the system is described by $\mathbf{X}(0)$, and hence, the initial distribution of traits at the tasks is described by $\mathbf{Y}(0)$. The time evolution of the number of robots of species s at task i is given by a linear law

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = \sum_{\forall j|(i,j)\in\mathcal{E}} k_{ji}x_i(t) - \sum_{\forall j|(i,j)\in\mathcal{E}} k_{ij}x_i(t) \tag{2}$$

which controls the robots transitioning to and from a task i (with rates k_{ji} and k_{ij}, respectively), and which exhibits an explicit feedback structure. Then, for all species s, our base model is given by

$$\frac{\mathrm{d}\mathbf{x}^{(s)}}{\mathrm{d}t} = \mathbf{K}^{(s)}\mathbf{x}^{(s)} \quad \forall s \in 1,\ldots,S \tag{3}$$

where $\mathbf{K}^{(s)} \in \mathbb{R}^{M\times M}$ is a rate matrix with the properties

$$\mathbf{K}^{(s)\top}\mathbf{1} = \mathbf{0} \tag{4}$$

$$\mathbf{K}^{(s)}_{ij} \geq 0 \quad \forall (i,j)\in\mathcal{E}. \tag{5}$$

These two properties result in the definition

$$\mathbf{K}^{(s)}_{ij} = \begin{cases} k^{(s)}_{ji}, & \text{if } i\neq j, (i,j)\in\mathcal{E} \\ 0, & \text{if } i\neq j, (i,j)\notin\mathcal{E} \\ -\sum_{i=1,(j,i)\in\mathcal{E}}^{M} k^{(s)}_{ij}, & \text{if } i=j \end{cases}.$$

Since the total number of robots and the number of robots per species is conserved, the system in (3) is subject to the constraints

$$\mathbf{X}^\top \cdot \mathbf{1} = [N^{(1)}, N^{(2)}, \ldots, N^{(S)}]^\top \tag{6}$$

$$\text{with } \mathbf{X} \succeq \mathbf{0} \tag{7}$$

where $\succeq$ is an element-wise greater-than-or-equal-to operator.

D. Problem Statement

Our aim is to redeploy the robots of each species, distributed according to $\mathbf{X}(0)$ initially, so that a desired trait distribution $\mathbf{Y}^\star$ is reached. As described in Section 1, we will consider two goals. A goal consists of a set of admissible trait distributions, and is described by a function $\mathcal{G} : \mathbb{N}^{+M\times U} \rightarrow \Omega$, where Ω is the set of sets of matrices of size $M \times U$. The goal function $\mathcal{G}$ takes as input a target trait distribution $\mathbf{Y}^\star$ and returns a set of admissible trait distributions $\mathcal{G}\ (\mathbf{Y}^\star)$.

We study the following two goal functions in detail.

1) $\mathcal{G}_1(\mathbf{Y}^\star) = \{\mathbf{Y}|\mathbf{Y}^\star = \mathbf{Y}\}$: This goal is achieved by a trait distribution that is exactly equal to the target trait distribution. Thus, the robots must organize themselves among tasks such that the exact number of traits is met for each task.
2) $\mathcal{G}_2(\mathbf{Y}^\star) = \{\mathbf{Y}|\mathbf{Y}^\star \preceq \mathbf{Y}\}$: This goal is achieved by trait distributions that are equal to or greater than the target trait distribution. Thus, robots can organize themselves such that there is an excess of traits for any task.

Finally, the problem consists of finding an optimal rate matrix $\mathbf{K}^{(s)\star}$ for each species s so that the goal is reached as quickly as possible

$$\mathbf{K}^{(s)\star}, \tau^\star = \underset{\mathbf{K}^{(s)},\tau}{\text{argmin}}\ \tau \tag{8}$$

$$\text{such that} \quad \mathbf{X}(\tau^\star)\cdot\mathbf{Q} \in \mathcal{G}(\mathbf{Y}^\star). \tag{9}$$

The solution leads to a robot configuration $\mathbf{X}(\tau^\star)$ that satisfies (9), subject to (6) and (7). In other words, by computing optimal rates, we are centrally synthesizing the feedback policy based on the abstract state information $\mathbf{X}(0)$. We will initially assume that this information can be gathered centrally, and that the control input $\mathbf{K}^{(s)\star}$ can be broadcast to the swarm. Later, in Section V, we see how to infer the abstract state information using local estimators, enabling the robots to synthesize the feedback policy in a decentralized manner.

III. Diversity Metric

Since the desired state of our system is solely described through $\mathbf{Y}^\star$, the corresponding final robot distribution $\mathbf{X}(\tau^\star) = \mathbf{X}^\star$ that achieves the goal $\mathcal{G}(\mathbf{Y}^\star)$ is not known *a priori*. In particular, there may be several $\mathbf{X}^\star$ that satisfy (9)—this is true for both goals $\mathcal{G}_1$ and $\mathcal{G}_2$. Hence, we pose the question: *Can we infer properties of the species-trait matrix* $\mathbf{Q}$ *that quantify how easy it is to find a solution* $\mathbf{X}^\star$ *that reaches* $\mathcal{G}(\mathbf{Y}^\star)$*?* In the following, we show how $\mathbf{Q}$ embodies the *diversity* of the robot community, and how we can quantitatively evaluate the diversity to make conclusions about the system's performance.

A. Definitions

Given an unlimited number of robots per species, it may be possible to reach any given goal $\mathcal{G}(\mathbf{Y}^\star)$ with a subset of the original robot species (independent of the target trait distribution $\mathbf{Y}^\star$). We call the species belonging to an inclusion-wise minimal subset the *minspecies*, and we refer to the size of this subset as the *minspecies cardinality* of the system. More formally, we introduce the following terminology:

Definition 1 (Minspecies): In a robot community described by a species-trait matrix $\mathbf{Q}$, a *minspecies* set is a subset of rows of $\mathbf{Q}$ with minimal cardinality, such that the system can still reach the goal $\mathcal{G}(\mathbf{Y}^\star)$. We represent minspecies by a matrix $\hat{\mathbf{Q}}$ containing a subset of the original rows of $\mathbf{Q}$ such that for any $\mathbf{Y}^\star$ there exists at least one robot distribution $\hat{\mathbf{X}}$ for which $\hat{\mathbf{X}}\hat{\mathbf{Q}} \in \mathcal{G}(\mathbf{Y}^\star)$.

Definition 2 (Minspecies cardinality): The *minspecies cardinality* of a robot community is given by the cardinality of the minspecies set. It is a function $\mathcal{D}_{\mathcal{G}} : \{0,1\}^{S\times U} \rightarrow \mathbb{N}^+$ that takes a species-trait matrix $\mathbf{Q}$ as input, and returns the minimum number of rows of $\mathbf{Q}$ that are needed to reach $\mathcal{G}(\mathbf{Y}^\star)$ for any $\mathbf{Y}^\star$.

B. Implementation

In this section, we develop the minspecies cardinality of our two goals $\mathcal{G}_1$ and $\mathcal{G}_2$. In particular, we demonstrate that for both goals, the minspecies cardinality is a meaningful quantitative measure of the constraint in (9).

Proposition 1: The minspecies cardinality with respect to goal $\mathcal{G}_1$ is

$$\mathcal{D}_{\mathcal{G}_1}(\mathbf{Q}) = \operatorname{rank}(\mathbf{Q}). \tag{10}$$

This implementation of the minspecies cardinality is directly related to the concept of algebraic independence, and hence, we use the specialized term *eigenspecies* (as previously introduced in [22]).

Proof: The admissible trait distribution set contains a single target trait distribution $\mathbf{Y}^\star$ and thus, (9) is equivalent to $\mathbf{Y}^\star = \mathbf{X}^\star\mathbf{Q}$. The matrix $\mathbf{Q}^\top$ can be rank factorized into the product of two matrices $\mathbf{A}$ and $\hat{\mathbf{Q}}$ such that $\mathbf{Q}^\top = \hat{\mathbf{Q}}^\top\mathbf{A}^\top$ with $\hat{\mathbf{Q}}$ containing a subset of the rows of $\mathbf{Q}$ [24]. Since $\mathbf{Y}^\star = \mathbf{X}^\star\mathbf{Q} = \mathbf{X}^\star\mathbf{A}\hat{\mathbf{Q}}$, there exists a robot distribution $\hat{\mathbf{X}} = \mathbf{X}^\star\mathbf{A}$ for which $\hat{\mathbf{X}}\hat{\mathbf{Q}} = \mathbf{Y}^\star$. Hence, as $\hat{\mathbf{Q}}$ has minimal size (due to the rank factorization), $\hat{\mathbf{Q}}$ is a minspecies matrix. ■

Indeed, the rank of $\mathbf{Q}$ quantifies the number of noncollinear species in $\mathbf{Q}$ that span the solution space of the equation $\mathbf{X}^\star\mathbf{Q} = \mathbf{Y}^\star$ (with $\mathbf{X}^\star$ unknown):

1) If $\operatorname{rank}(\mathbf{Q}) < S$, the system is underdetermined, and an infinite number of solutions $\mathbf{X}^\star$ will satisfy (9). In other words, at least one species in the system can be replaced by a combination of the other species. As the rank decreases, the *redundancy* of the community increases.
2) If $\operatorname{rank}(\mathbf{Q}) = S$, there is only one solution $\mathbf{X}^\star$ that satisfies (9). In other words, no species in the system can be replaced by a combination of the other species, and all species are fully *complementary*.

As an example, consider matrix

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \mathbf{A}\cdot\hat{\mathbf{Q}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

The rank of $\mathbf{Q}$ is 2, hence, $\mathcal{D}_{\mathcal{G}_1}(\mathbf{Q}) = 2$, which is the number of species in $\hat{\mathbf{Q}}$.

Proposition 2: The minspecies cardinality with respect to goal $\mathcal{G}_2$ is

$$\mathcal{D}_{\mathcal{G}_2}(\mathbf{Q}) = \min \sum_{s=1}^{S} a^{(s)} \tag{11}$$

$$\text{such that} \sum_{s=1}^{S} a^{(s)}\mathbf{q}^{(s)} \succ \mathbf{0} \text{ and } a^{(s)} \in \{0,1\}.$$

This implementation of the minspecies cardinality is directly related to the concept of cover sets, and hence, we use the specialized term *coverspecies*.

Proof: The admissible trait distribution set for $\mathcal{G}_2$ contains all trait distributions that contain at least the specified amount of traits per task. Equation (9) under goal $\mathcal{G}_2$ becomes $\mathbf{Y}^\star \preceq \mathbf{X}^\star\mathbf{Q}$. The matrix $\mathbf{Q}$ can be factorized into a product of two matrices $\mathbf{A}$ and $\hat{\mathbf{Q}}$ such that $\mathbf{Q} \preceq \mathbf{A}\hat{\mathbf{Q}}$. Since $\mathbf{Y}^\star \preceq \mathbf{X}^\star\mathbf{Q} \preceq \mathbf{X}^\star\mathbf{A}\hat{\mathbf{Q}}$ (since $\mathbf{X}^\star \succeq \mathbf{0}$), there exists a distribution $\hat{\mathbf{X}} = \mathbf{X}^\star\mathbf{A}$ for which the goal is reached. If $\hat{\mathbf{Q}}$ has the minimum number of rows possible, $\hat{\mathbf{Q}}$ is a minspecies matrix, and thus, finding $\hat{\mathbf{Q}}$ amounts to finding the minimum cover set required to cover all traits with selected species of $\mathbf{Q}$. ■

We consider the same example as above, which is

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \preceq \mathbf{A}\cdot\hat{\mathbf{Q}} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}.$$

The minimum cover set has size 1, hence, $\mathcal{D}_{\mathcal{G}_2}(\mathbf{Q}) = 1$, which is the number of species in $\hat{\mathbf{Q}}$.

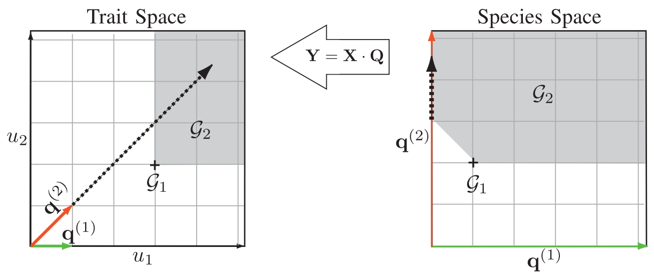


Fig. 3. This basic example illustrates how a goal is achieved by two species $\mathbf{q}^{(1)}$ and $\mathbf{q}^{(2)}$. Species $\mathbf{q}^{(1)}$ only owns trait u_1, and species $\mathbf{q}^{(2)}$ owns both traits u_1 and u_2. The left panel shows how the two species span the trait space. Goal $\mathcal{G}_1$ can only be reached by a combination of both species, whereas goal $\mathcal{G}_2$ can be reached by species $\mathbf{q}^{(2)}$ alone. The same insight can be made by observing the right panel, which shows the goals in species space. We remind the reader that the species-trait matrix $\mathbf{Q}$ is used to map robot species into trait space.

Fig. 3 illustrates a basic example of how goals $\mathcal{G}_1$ and $\mathcal{G}_2$ are achieved by two species, defined as $\mathbf{q}^{(1)} = [1,\ 0]$ and $\mathbf{q}^{(2)} = [1,\ 1]$. The left panel shows how the goals occupy the trait space, and the right panel shows how they occupy the species space. In particular, the panels illustrate how $\mathcal{G}_2$ can be reached by species $\mathbf{q}^{(2)}$ alone (see the dashed arrow). Species $\mathbf{q}^{(1)}$ and $\mathbf{q}^{(2)}$ both belong to the eigenspecies of $\mathcal{G}_1$, whereas $\mathbf{q}^{(2)}$ is the only coverspecies of $\mathcal{G}_2$.

IV. Methodology

In this section, we describe our methodology for obtaining an optimal transition matrix $\mathbf{K}^{(s)\star}$ for each species so that the desired trait distribution is reached as fast as possible. Initially, we assume global knowledge of abstract state information $\mathbf{X}(0)$ (i.e, the initial distribution of the robot swarm among tasks). Later, as we decentralize our method, we show how to build approximations of $\mathbf{X}(t)$, based on real-time local information.

Two general optimization approaches have been considered so far [16]: convex optimization and stochastic optimization. The convex optimization approach requires knowledge of the desired final robot distribution. However, our problem formulation specifies a desired trait distribution $\mathbf{Y}^\star$ without explicit definition of the final robot distribution $\mathbf{X}^\star$. Fully stochastic schemes such as Metropolis optimization have been shown to produce similar results, but they are not computationally efficient, and are ill-suited to real-time applications. In the following, we present a differentiable constrained optimization problem that can be efficiently solved through gradient descent techniques. Our method explicitly minimizes the convergence time of $\mathbf{K}^{(s)}$, unlike the convex optimization methods presented in [16], which approximate $\mathbf{K}^{(s)}$ with a symmetric equivalent (forcing bidirectionally equal transition rates between tasks). Additionally, it is able to find optimal transition rates with only knowledge of $\mathbf{Y}^\star$ and $\mathbf{X}(0)$ (i.e., without knowledge of $\mathbf{X}^\star$).

A. Optimization Problem

We combine the solution of the linear ordinary differential equation, (3), with (1) to obtain the solution

$$\mathbf{Y}(\mathbf{K}^{(1\ldots S)}, t; \mathbf{X_0}) = \sum_{s=1}^{S} e^{\mathbf{K}^{(s)}t}\mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)}. \tag{12}$$

To find the optimal transition rates $\mathbf{K}^{(s)\star}$ for $\mathcal{G}_1$ (details for $\mathcal{G}_2$ are in the appendix), we consider the trait distribution error

$$\mathbf{E_1}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0}) = \left\| \mathbf{Y}^\star - \mathbf{Y}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0}) \right\|_F^2 \tag{13}$$

where τ is the time at which the desired state is reached. The notation $\mathbf{x}_0^{(s)}$ is shorthand for $\mathbf{x}^{(s)}(0)$. The operator $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. An objective function based on $\mathbf{E_1}$ alone will return transition rates that may lead to the desired trait distribution quickly, but there is no guarantee that this state also is a steady state. Additionally, if we compute the transition rates at the outset of the experiment (without refining them online), we may wish to ensure that the state reached at the optimal time $\tau^\star$ remains near constant. Hence, we also consider the difference in robot distribution

$$\mathbf{E_2}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0}) = \sum_{s=1}^{S} \left\| e^{\mathbf{K}^{(s)}\tau}\mathbf{x}_0^{(s)} - e^{\mathbf{K}^{(s)}(\tau+\nu)}\mathbf{x}_0^{(s)} \right\|_2^2 \tag{14}$$

at time τ (when the desired trait distribution should be reached) and time $\tau + \nu$. Enforcing a small value for $\mathbf{E_2}$ allows us to guarantee that the robot distribution remains near constant for arbitrarily long time intervals ν. This is possible because our model in (3) is stable [17], and the difference between the current robot distribution and the one at steady state can only decrease monotonically over time. In other words, the trait distribution corresponding to the steady state of $\mathbf{K}^{(s)\star}$ gets arbitrarily close to the distribution reached at τ as ν increases. We can now formulate our constrained optimization problem as

$$\begin{aligned} &\text{minimize} \quad \tau \\ &\text{such that} \quad \mathbf{E_1}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0}) \le \epsilon_1 \\ &\qquad\qquad\quad \mathbf{E_2}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0}) \le \epsilon_2 \\ &\qquad\qquad\quad k_{ij}^{(s)} \le k_{ij,\max}^{(s)} \\ &\qquad\qquad\quad \tau > 0 \end{aligned} \tag{15}$$

which states that an optimal time $\tau^\star$ is found when the final trait distribution error is smaller than the admissible squared trait error ϵ_1, and when the difference in robot distributions at times $\tau^\star$ and $\tau^\star + \nu$ is smaller than the admissible squared deviation ϵ_2, subject to maximum transition rates $k_{ij,\max}^{(s)}$. The smaller we choose ϵ_1, the closer the trait distribution at time $\tau^\star$ will be to the desired trait distribution. The smaller we choose ϵ_2, the closer the robot distribution at time $\tau^\star$ is to the steady-state distribution of $\mathbf{K}^{(s)\star}$ (and the closer the trait distribution is to the steady-state trait distribution). While the first constraint will decrease $\tau^\star$, the second constraint will tend to increase it.

B. Analytical Gradients

There is no closed-form solution to the optimization problem in (15), hence we resort to numerical techniques. Nevertheless, we can maximize the efficiency of our computations by finding the closed-form expression of the gradient, for each of our constraints. In the following, for better readability, we will omit the explicit notation of the parameters of $\mathbf{E_1}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0})$ and $\mathbf{E_2}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X_0})$, and write $\mathbf{E_1}$ and $\mathbf{E_2}$, respectively. Let us first consider the derivative of $\mathbf{E_1}$. By applying the chain rule, the derivative with respect to the transition matrix $\mathbf{K}^{(s)}$ is

$$\frac{\partial \mathbf{E_1}}{\partial \mathbf{K}^{(s)}} = \frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} \cdot \frac{\partial e^{\mathbf{K}^{(s)}\tau}}{\partial \mathbf{K}^{(s)}\tau} \cdot \frac{\partial \mathbf{K}^{(s)}\tau}{\partial \mathbf{K}^{(s)}}. \tag{16}$$

We first compute the derivative of the cost with respect to the expression $e^{\mathbf{K}^{(s)}\tau}$

$$\frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} = -2\left[\mathbf{Y}^\star - \mathbf{Y}(\mathbf{K}^{(s)}, \tau; \mathbf{X_0})\right] \cdot \left[\mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)}\right]^\top. \tag{17}$$

The derivation of the second element of (16) requires the derivative of the matrix exponential. Computing the derivative of the matrix exponential is not trivial. We adapt the closed-form solution given in [25] to our problem, and write the gradient of our constraint as

$$\frac{\partial \mathbf{E_1}}{\partial \mathbf{K}^{(s)}} = \mathbf{V^{-1}}^\top \left[\mathbf{V}^\top \frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau)\right] \mathbf{V}^\top \tau \tag{18}$$

where $\odot$ is the Hadamard product, $\mathbf{K}^{(s)} = \mathbf{V}\mathbf{D}\mathbf{V}^{-1}$ is the eigendecomposition of $\mathbf{K}^{(s)}$. $\mathbf{V}$ is the $M \times M$ matrix whose jth column is a right eigenvector corresponding to eigenvalue d_i, and $\mathbf{D} = \text{diag}(d_1, \ldots, d_M)$. The matrix $\mathbf{W}(t)$ is composed as follows[1]

$$\mathbf{W}(t) = \begin{cases} (e^{d_i t} - e^{d_j t})/(d_i t - d_j t) & i \neq j \\ e^{d_i t} & i = j \end{cases}.$$

We also need the derivative with respect to parameter τ. This derivative is computed analogously to the derivative with respect to $\mathbf{K}^{(s)}$ [confer (18)]. We have

$$\frac{\partial \mathbf{E_1}}{\partial \tau} = \sum_{s=1}^{S} \mathbf{1}^\top \mathbf{V^{-1}}^\top \mathbf{A}_1 \mathbf{V}^\top \mathbf{K}^{(s)} \mathbf{1} \tag{19}$$

[1]Here, we assume that $\mathbf{K}^{(s)}$ has M distinct eigenvalues. If this is not the case, an analogous decomposition of $\mathbf{K}^{(s)}$ to Jordan canonical form is possible, as elaborated in [25]. We note that for most models of interest, however, this is rarely the case.

and

$$\mathbf{A}_1 = \mathbf{V}^\top \frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau). \tag{20}$$

Now let us consider the derivative of the second constraint in (15). Again, we apply the chain rule to obtain

$$\frac{\partial \mathbf{E_2}}{\partial \mathbf{K}^{(s)}} = \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}\tau}} \frac{\partial e^{\mathbf{K}^{(s)}\tau}}{\partial \mathbf{K}^{(s)}\tau} \frac{\partial \mathbf{K}^{(s)}\tau}{\partial \mathbf{K}^{(s)}} - \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} \frac{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}}{\partial \mathbf{K}^{(s)}(\tau+\nu)} \frac{\partial \mathbf{K}^{(s)}(\tau+\nu)}{\partial \mathbf{K}^{(s)}}. \tag{21}$$

The outer derivative is

$$\frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}\tau}} = \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} = 2\left[e^{\mathbf{K}^{(s)}\tau}\mathbf{x}_0^{(s)} - e^{\mathbf{K}^{(s)}(\tau+\nu)}\mathbf{x}_0^{(s)}\right] \cdot \mathbf{x}_0^{(s)\top}. \tag{22}$$

We apply the same development as in (18) to obtain

$$\frac{\partial \mathbf{E_2}}{\partial \mathbf{K}^{(s)}} = \mathbf{V^{-1}}^\top \left[\mathbf{A}_2\tau - \mathbf{A}_3(\tau+\nu)\right]\mathbf{V}^\top \tag{23}$$

with

$$\mathbf{A}_2 = \mathbf{V}^\top \cdot \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}\tau}} \cdot \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau) \tag{24}$$

and

$$\mathbf{A}_3 = \mathbf{V}^\top \cdot \frac{\partial \mathbf{E_2}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} \cdot \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau+\nu). \tag{25}$$

The derivative with respect to time τ is analogous

$$\frac{\partial \mathbf{E_2}}{\partial \tau} = \sum_{s=1}^{S} \mathbf{1}^\top \mathbf{V^{-1}}^\top \left[\mathbf{A}_2 - \mathbf{A}_3\right]\mathbf{V}^\top \mathbf{K}^{(s)}\mathbf{1}. \tag{26}$$

For all the above, the derivative with respect to the off-diagonal elements ij of the matrix $\mathbf{K}^{(s)}$, with $(i,j) \in \mathcal{E}$, is

$$\frac{\partial \mathbf{E_z}}{\partial k_{ij}^{(s)}} = \left\{\frac{\partial \mathbf{E_z}}{\partial \mathbf{K}^{(s)}}\right\}_{ij} - \left\{\frac{\partial \mathbf{E_z}}{\partial \mathbf{K}^{(s)}}\right\}_{jj} \tag{27}$$

where $\{\cdot\}_{ij}$ denotes the element on row i and column j.

C. Computational Complexity

The overall computational complexity of computing the gradients of both constraints of our optimization problem is $O(S \cdot M^3 + S \cdot M^2 \cdot U)$. The first part of this complexity is dictated by the eigenvalue decomposition, which is known to be $O(M^3)$ for nonsparse matrices [26].[2] We compute this decomposition only once per optimization [see (18), where $\mathbf{K}^{(s)} = \mathbf{VDV}^{-1}$], for each optimization of $\mathbf{K}^{(s)}$. The second part is dictated by the multiplication of the matrices in (18), for which the cost is $O(M^2 \cdot U)$. Globally speaking, the computation grows linearly with the number of species and traits, and it grows slightly slower than the cube of the number of tasks. When studying heterogeneous system, it is indeed a valuable result that the gradient scales at most linearly with the number of traits and species in order to allow for the exploration of a wider range of robot capabilities. Overall, the average time to compute the gradient for a system with $M = 8$, $U = 4$, and $S = 4$ is around 1.35 ms with $\nu = 0$, and 2.2 ms with $\nu > 0$ (the number of parameters to optimize can be as large as 225 in this case, depending on the graph's adjacency matrix). The code was implemented in Python using the NumPy and SciPy libraries, and tested on a 2 GHz Intel Core i7 using a single CPU.

[2]In the special case where all eigenvalues are distinct, the eigenvalue decomposition can be reduced to $O(M^{2.376}\log(M))$ [27].

D. Online Optimization of $\mathbf{K}^{(s)}$

Thanks to the analytical gradients developed above, our optimization problem can be solved efficiently. Building on this result, we implement a continuous, online optimization strategy that allows us to refine the optimal $\mathbf{K}^{(s)\bigstar}$ as a function of the current state of the robot system, in real time. In noisy systems, where the trajectories of individual agents exhibit deviations from predicted macroscopic trajectories, this strategy inevitably leads to an improvement of the convergence time. Furthermore, as seen in our example in Figs. 1 and 2, we may wish to program a sequence of desired trait distributions, with autonomous transition rate updates. The key idea is that by taking the actual robot distribution into account, an online method can recompute updated optimal transition rates. Practically, we initially compute $\mathbf{K}^{(s)\bigstar}$ at time $t = 0$ to control the system over a finite period δ from $t = 0$ to $t = \delta$. After that period (at time $t = \delta$), we optimize a new value of $\mathbf{K}^{(s)\bigstar}$ that controls the system for the next period, as a function of the actual robot distribution that was encountered at time $t = \delta$. This process can be repeated indefinitely. The value δ is called the sampling time. Our online control policy computes the optimal transition rates, and is rewritten as a function of the robot distribution

$$\begin{aligned} \mathbf{K}^{(s)\bigstar}(t), \tau^\bigstar(t) = \; & \underset{\mathbf{K}^{(s)},\tau}{\text{argmin}}\; \tau \\ \text{such that} \quad & \mathbf{E_1}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X}(t_p)) \le \epsilon_1 \\ & \mathbf{E_2}(\mathbf{K}^{(1\ldots S)}, \tau; \mathbf{X}(t_p)) \le \epsilon_2 \\ & k_{ij}^{(s)} \le k_{ij,\max}^{(s)} \\ & \tau > 0, \\ \text{with} \quad & t_p \le t < t_p + \delta \\ & t_p \in k\delta, k \in \mathbb{N} \end{aligned} \tag{28}$$

where t_p is the time at which optimizations happen. For cases where the optimization time becomes large (implying that δ also becomes large), we need to use a strategy that accounts for computation delay, such as those presented in [28]. Also, we note that we can accelerate the computations by setting the initial values of the present sampling window with optimized values of the preceding sampling window (i.e., warm start).

V. Robot Controller

The previous sections describe the methods with which we obtain optimal transition rates $\mathbf{K}^{(s)\bigstar}$. If we assume an architecture such as described in [10], then the optimization is run off-board, centrally, with knowledge of abstract state information $\mathbf{X}(t)$ (i.e, the current distribution of the robot swarm among tasks). In return, the robots only need to receive information on the optimal transition rates for their species, $k_{ij}^{(s)}$. We note that this information is represented by a small number of values (at most M^2 values per species, or a much smaller number if the graph is sparse), and needs to be transmitted to the robots only at the start of each new redistribution.

Conversely, if the robots run the optimization algorithm onboard, they need to estimate the abstract state information $\mathbf{X}(t)$ locally. This knowledge can be obtained through communication with neighboring robots (which broadcast their current task allocation). Due to communication constraints, obtaining an accurate representation of $\mathbf{X}(t)$ is often not possible. Hence, our methods must perform acceptably well, even when these values are approximate. We choose the following approach for estimating $\mathbf{X}(t)$: the robot locally employs a uniform distribution to model the allocation of robots for which it does not have any knowledge. We note that the uniform distribution is merely a placeholder for more sophisticated models. We choose this distribution since it represents the model of highest entropy. Previous work has discussed the violation of the well-mixed assumption in swarm systems [29], [30]. Indeed, any additional information that contributes to a more accurate estimate of $\mathbf{X}(t)$ will allow the performance of the system to approach the optimal solution (for which the exact distribution $\mathbf{X}(t)$ is known). In order to build more accurate models, we may need to take spatial correlations into account. For this, we would first need to devise a model of such higher order moments, and second, populate this model with the information necessary to construct higher order moment estimates. It is particularly challenging to do the latter in a decentralized manner. Previous literature [31]–[33] proposes solutions to these problems. In particular, the solution in [33] proposes a decentralized implementation of ensemble feedback control that relies solely on local interrobot communication. As robots move from one site to another and exchange information with other robots they encounter, each robot can construct its own estimate of the population levels at the various sites. Such an approach could readily be implemented in conjunction with our distribution policy, hence accounting for issues related to nonuniform spatial distributions.

The agent-level control is based on the transition rates $k_{ij}^{(s)}$ encoded by the transition matrix $\mathbf{K}^{(s)}$: A robot of species s at task i transitions to task j according to probability $p_{ij}^{(s)}$, i.e., an element of the matrix $\mathbf{P}^{(s)} = e^{\mathbf{K}^{(s)}\Delta T}$, where ΔT is the duration of one time step. Hence, in order to determine which task the robot must transition to next, it samples a new task with a probability according to $\mathbf{P}^{(s)}$. This is equivalent to sampling from the discrete probability distribution $\mathcal{P}(p_{i1}^{(s)}, \ldots, p_{iM}^{(s)})$, where i represents the current task. This procedure is shown in Algorithm 1. We note that as the robot is transitioning to a new task, it continues the control loop (i.e., sampling new tasks). Although we do not explicitly model transitioning time, the resulting behavior is very close to what is predicted by the macroscopic model in (3), as is shown later in Section VI. Finally, we note that although we detail a probabilistic robot controller, our methods are equally well suited for robots that are deterministically controlled, and whose overall behavior can be captured by stochastic quantities (due to random errors and events).

Algorithm: `Controller`$(s, M, N^{(1),\ldots,(S)}, \mathbf{Q}, \mathbf{Y}^\star, \delta, \Delta T)$.

```
1: $i \leftarrow$ GetInitialTask()
2: $t \leftarrow 0$
3: while 1 do
4:   BroadcastAllocation($< s, i >$)
5:   if modulo($t, \delta$) $= 0$ then
6:     $A \leftarrow$ GetAllocations()
7:     $\mathbf{X}_{local} \leftarrow$ EstimateRobotDistr($A, N^{(1),\ldots,(S)}$, $M$)
8:     $\mathbf{K}^{(s)\star} \leftarrow$ Optimize($\mathbf{X}_{local}, \mathbf{Q}, \mathbf{Y}^\star$)
9:     $\mathbf{P}^{(s)} = e^{\mathbf{K}^{(s)\star}\Delta T}$
10:  end if
11:  $t \leftarrow t + \Delta T$
12:  $m \sim \mathcal{P}(p_{i1}^{(s)}, \ldots, p_{iM}^{(s)})$
13:  if $m \neq i$ then
14:    Switch to task $m$
15:    $i \leftarrow m$
16:  end if
17:  Wait $\Delta T$
18: end while
```

VI. Results

We present results that show the following.

1) Our method successfully achieves the deployment of a heterogeneous system of robots so that a desired trait distribution is reached.
2) Our method extends itself to decentralized architectures (that can approximate certain global quantities locally).
3) We are able to relate the performance to the diversity of the system.

We evaluate these claims over multiple levels of abstraction.

A. Performance Metric

The degree of convergence to $\mathbf{Y}^\star$ is expressed by the fraction of misplaced traits. For our two goals, we formulate this as

$$\mu_{\mathcal{G}_1}(\mathbf{Y}) = \frac{\|\mathbf{Y}^\star - \mathbf{Y}\|_1}{2\|\mathbf{Y}^\star\|_1}, \mu_{\mathcal{G}_2}(\mathbf{Y}) = \frac{\|\max(\mathbf{Y}^\star - \mathbf{Y}, 0)\|_1}{\|\mathbf{Y}^\star\|_1}. \tag{29}$$

Previous work has shown the benefit of validating methods over multiple levels of abstraction (submicroscopic, microscopic, and macroscopic) [29]. In this section, we propose an evaluation of our methods on two levels: macroscopic and microscopic. Indeed, an efficient way for predicting the behavior of a large-scale system of robots is by considering a macroscopic model, which removes the need of simulating each robot individually. This continuous model is derived directly from the ordinary differential equation, (3). In order to validate our methods at a lower level of abstraction, we also implement a discrete microscopic model that emulates the behavior of individual robot controllers. The agent-level control is based on Algorithm 1. Running multiple iterations of the microscopic model enables us to capture the stochasticity resulting from our control system. In the remainder of this paper, we use $\Delta T = 0.04$ s, unless stated otherwise.

B. Example

To illustrate our method in more detail, let us consider the example portrayed earlier, in Fig. 1. The graph is generated randomly according to the Watts–Strogatz model [34] (with a neighboring node degree of $K = 3$, and a rewiring probability of $\gamma = 0.6$; the graph is guaranteed to be connected). The robot community consists of three species and four traits, and is defined as

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix} \quad \text{with} \quad \mathbf{X}^\top \cdot \mathbf{1} = [150, 50, 300]^\top.$$

In this example, $N = 500$ robots transition among $M = 10$ tasks. We sample a random initial robot distribution $\mathbf{X}(t_0)$ with a majority of traits in use at tasks 1, 2, and 3. We specify a randomly generated desired trait distribution, which is visualized in Fig. 1 at t_1. The final robot distribution $\mathbf{X}(t_1)$ then serves as the initial distribution for a subsequent reconfiguration targeting the trait distribution visualized at t_2. As this process is repeated, it demonstrates how our method can be used to redistribute a swarm of robots through time so that changing trait requirements are met. The centralized implementation follows naturally. In case of a decentralized implementation, the quantities $\mathbf{X}(t)$ can be approximated locally (and hence decentralized), whereas the desired trait distributions $\mathbf{Y}^\star(t)$ are defined centrally, and need to be transmitted to all robots *a priori* (before the start of a new reconfiguration).

To illustrate the performance of our method, we implement a kinematic point simulator, emulating a swarm of robots at a microscopic level. The robots move on a two-dimensional plane, which is 3 m in size, where the tasks are represented by spatially anchored sites that have a radius of 0.05 m, and are placed along a circle of radius 1.75 m. We assume that the robots have perfect knowledge of their positions. The paths between the sites are defined according to the adjacency matrix of the graph shown in Fig. 1. The robots travel with an average speed of 0.06 $\mathrm{m/s}$, as they transition from site to site, with a maximum transition rate $k^{(s)}_{ij,\max} = 0.02\ \mathrm{s}^{-1}$. Fig. 4 shows the trail laid by three robots during the period t_0 to t_1 of our example scenario, as they travel from their initial sites to the final site (we show only three out of the total 500 robot trails to avoid cluttering the plot). We note that the motion control used for the robots in this simulator is identical to the one used to control physical robots in a previous experiment [22].

In order to quantify the performance of our system, we perform ten runs of our simulator, and evaluate the ratio of misplaced traits as a function of time. We also evaluate the performance of the system at a macroscopic level. The results

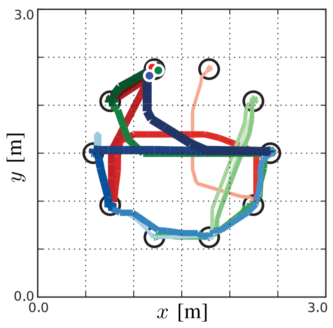


Fig. 4. Trail for selected three robots (out of the total 500 robots), one of each species, for the first 700 s (segment t_0 to t_1) in Fig. 1. The robots start at tasks 1, 2, and 3, respectively, and end at task 4. The earlier the trail, the more transparent the color.

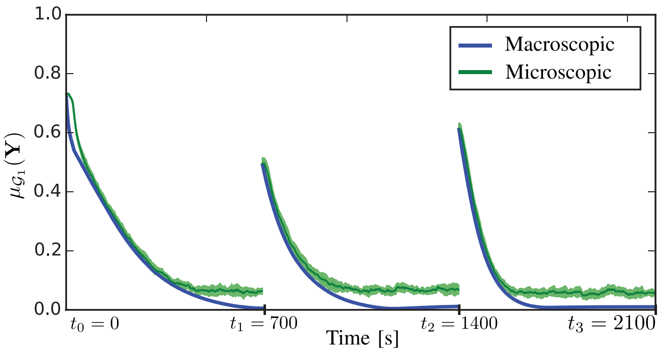


Fig. 5. Ratio of misplaced traits for the redistribution problem shown in Fig. 1. The plot shows the macroscopic model as well as the average over ten iterations of the microscopic model. The shaded area shows the standard deviation.

are shown in Fig. 5. We observe that the trait error decreases exponentially. Initially, the microscopic and macroscopic models show good agreement, but as the system approaches steady state, the stochasticity of the microscopic point simulator forces the error ratio (which counts absolute differences) to be larger than 0. Note that systems with slower dynamics and more robots have a lower noise intensity, and achieve lower average errors at steady state. This experiment shows that our framework is able to cope with spatiality and temporal delays, even though these phenomena are not modeled explicitly in our optimization problem. Throughout the rest of this section, we will focus on the core properties of our method, and hence, we simplify our implementation of the microscopic model by considering instantaneous transitions from one task to the next, within nonspatial configurations.

C. Online Optimization

We evaluate the performance of our online optimization algorithm described in Section IV-D, for both goal functions $\mathcal{G}_1$ and $\mathcal{G}_2$, and compare it to the macroscopic model. Fig. 6 shows the ratio of misplaced traits $\mu(\mathbf{Y})$ over time for a graph with $M = 8$ nodes, for $S = 4$ and $U = 5$, and a total number of robots $N = 1000$. The initial distribution consists of traits randomly allocated to one half of the tasks, and the desired distribution consists of traits randomly allocated to the remaining half of

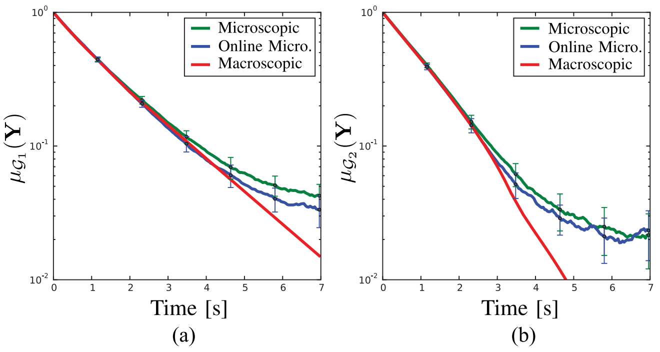


Fig. 6. Ratio of misplaced traits over time, in log scale, for a graph with $M = 8$ nodes. The plot shows the macroscopic model as well as the average over 50 iterations of the microscopic model, with and without online optimization. The error bars show the standard deviation. (a) Goal function $\mathcal{G}_1$, (b) Goal function $\mathcal{G}_2$.

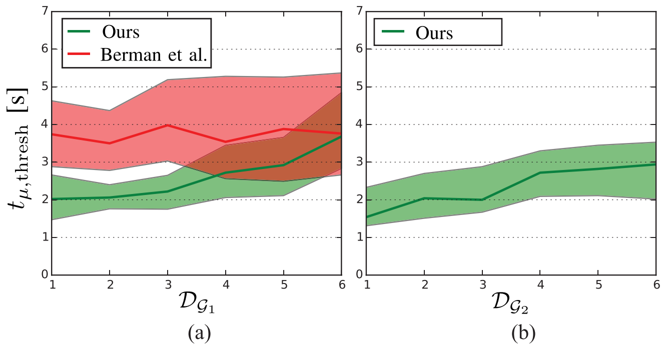


Fig. 7. Plot shows the median convergence time evaluated on the microscopic model, with $t_{\mu_{\text{thresh}}}$ for $\mu_{thresh} = 2.5\%$, as a function of the minspecies cardinality, for 60 random graphs per cardinality value. The system has $M = 10$ tasks and $S = 6$ species. The shaded area shows the 25th and 75th percentiles. (a) Goal function $\mathcal{G}_1$, (b) Goal function $\mathcal{G}_2$.

tasks. Fig. 6(a) considers goal $\mathcal{G}_1$ (matching the desired trait distribution *exactly*) and Fig. 6(b) considers goal $\mathcal{G}_2$ (matching a *minimum* desired amount of traits per task). For both plots, we run 50 iterations of the discrete microscopic model, with and without online optimization. The online optimization method was implemented with $\delta = 20 \cdot \Delta T$. We observe that the trait error decreases exponentially. Since the online optimization method takes the current robot configuration into account, it produces transition rates that lead to lower errors. Finally, we observe that goal $\mathcal{G}_2$ converges faster than $\mathcal{G}_1$, due to the additional degrees of freedom in the system (i.e., traits that can be allocated to any task in the system, as soon as the minimum amount is reached). As previously observed in Fig. 5, we see that here as well, the stochasticity of the microscopic models prevents the error from continuing to decrease exponentially as the system approaches steady state.

D. Impact of Diversity

Our aim is to observe the impact of diversity on system performance. We accomplish this by evaluating the time of convergence to the desired trait distribution as a function of our proposed diversity measure, the minspecies cardinality. We consider a system of $M = 10$ tasks and $S = 6$ species, and generate random species-trait matrices $\mathbf{Q}$ (for both $\mathcal{G}_1$ and $\mathcal{G}_2$) with minspecies cardinality values ranging from 1 to 6. The system is evaluated on 60 graphs, for each minspecies cardinality value, with a random initial robot distribution and a random desired trait distribution per graph. We measure the time $t_{\mu,\text{thresh}}$ at which the system converges to a value $\mu_{\text{thresh}} = 2.5\%$ of misplaced traits, and say that one system converges faster than another if it takes less time for $\mu(\mathbf{Y})$ to decrease to μ_{thresh}. Similar performance metrics have been proposed in [16] and [17].

Fig. 7 shows the results for the goal $\mathcal{G}_1$. Our optimization method is shown in green. We see that as the minspecies cardinality of the system increases, the time to convergence also increases. Indeed, the size of the solution space of (9) decreases as the minspecies cardinality increases. In other words, the more the species are complementary, the harder the system is to optimize. Also, we compare our method to a benchmark convex optimization approach that stems from [16], denoted in the latter work as **[P1]**.[3] We choose this method because it is based on a similar formalism, and because it has roughly the same computational complexity as our method. Importantly, though, its formalism only captures homogenous robot swarms (and is thus artificially bootstrapped to solve our problem). The results of this method are shown in red. We see that the performance does not correlate with the minspecies cardinality. Since the method does not optimize the reconfiguration for desired trait distributions, it is input with a potentially suboptimal final robot distribution (which is exacerbated for low minspecies cardinality). Our method improves upon this state-of-the-art benchmark method by 25% for $\mathcal{D}_{\mathcal{G}_1} = 5$ and by 46% for $\mathcal{D}_{\mathcal{G}_1} = 1$. Fig. 7(b) shows the results for the goal $\mathcal{G}_2$. As before, we see that as the minspecies cardinality of the system increases, the time to convergence also increases. We verify that $\mathcal{D}_{\mathcal{G}_2}$ is a more appropriate measure of diversity than $\mathcal{D}_{\mathcal{G}_1}$ for goal $\mathcal{G}_2$ by computing the Pearson correlation coefficient. Using $\mathcal{D}_{\mathcal{G}1}$ on this data produces a correlation of 0.23 (with p-value $< 10^{-4}$), while $\mathcal{D}_{\mathcal{G}_2}$ produces a correlation of 0.35 (with p-value $< 10^{-4}$), which is a 52% increase over $\mathcal{D}_{\mathcal{G}_1}$. This validates the use of two distinct diversity measures for our two distinct goals.

E. Decentralized Online Performance Optimization

To conclude Section VI, we consider the online optimization of transition rates within a decentralized controller that uses only local communication to obtain state information about the robot swarm. We remind the reader that the robots need the knowledge of abstract state information (i.e., the distribution of the robot swarm among tasks, $\mathbf{X}(t_p)$), at the start of each optimization, see (28). Hence, if we intend the robots to obtain this information

[3]This method implicitly optimizes the convergence time by optimizing the asymptotic convergence rate (of a system of homogenous robots). We adapt the method to our problem: we minimize the second eigenvalue λ_2 of a symmetric matrix $\mathbf{S}^{(s)}$, such that $\lambda_2(\mathbf{S}^{(s)}) \geq \text{Re}(\lambda_2(\mathbf{K}^{(s)}))$. Since this method requires the knowledge of the desired species distribution $\mathbf{X}^\star$, we artificially bootstrap the method by computing a random instantiation of $\mathbf{X}^\star$ that satisfies the desired trait distribution defined by (1). We note that in practical applications, computing a good instantiation of $\mathbf{X}^\star$ is not trivial.

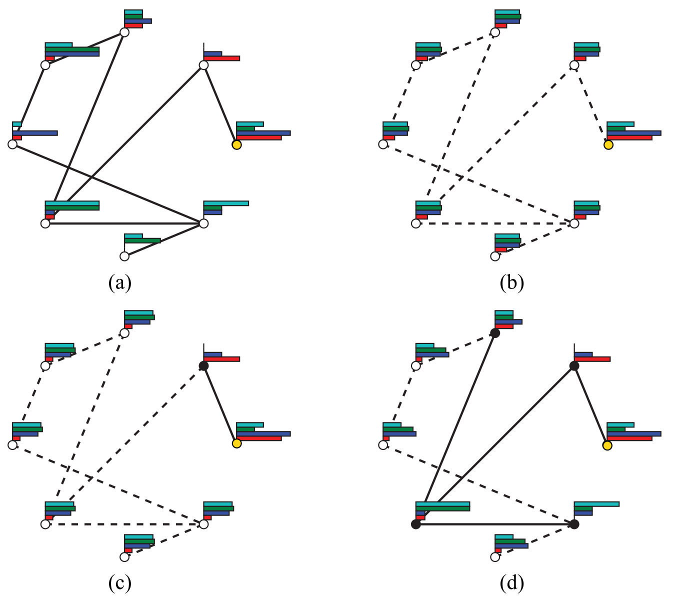


Fig. 8. Communication topology, as perceived by robots located at the task marked in yellow. The robots have knowledge of the trait distribution in their local neighborhood, which is defined by the hop count. We show the topology for 0, 1, and 3-hop neighborhoods, where tasks within the neighborhood are marked in black. Panel (a) shows the actual trait distribution, and panels (b), (c) and (d) show the estimated uniform distribution of remaining traits among all tasks that are outside of the local neighborhood. (a) Full, (b) 0-Hops, (c) 1-Hop, (d) 3-Hops

through local communication channels only, then we need to understand the effects that different communication topologies will have on the performance of the system. Our controller is based on (28), which updates transition rates at time intervals δ. To emulate communication constraints, we consider incremental coverage: at the most restricted level, we assume that robots are only able to communicate with other robots collocated at the same task; this assumption is incrementally relaxed, as we increase the communication neighborhood to include robots at adjacent tasks, within a fixed hop-count relative to the present task. Fig. 8 illustrates this concept for hop counts of 0, 1, and 3. For this particular graph, a hop count of 4 reaches full coverage [shown in Fig. 8(a)].

Fig. 9(a) shows the ratio of misplaced traits as a function of time, for five different hop counts. Fig. 9(b) shows the distribution of the data points of the final time step, with inclusion of all extrema. We observe that, the more we restrict our communication topology, the higher the error at steady state. We note that this performance degradation is graceful—as we reduce the size of our communication neighborhood from 4 hops to 0 hops, the absolute difference to the best case scenario (4 hops) is 0.1%, 0.4%, 1%, and 3%, respectively, and still converges to a modest error of 8% for the most restrictive communication radius. The swarm is able to perform relatively well, even for 0 hops, because robots exploit local knowledge available at their current task node (i.e., a robot knows which robots are collocated at the same task). Then, as robots switch to new tasks, they re-estimate the distribution based on their new local neighborhood (due to the online optimization method). As a result, the trait distribution progresses toward the desired value, albeit with a slower convergence rate.

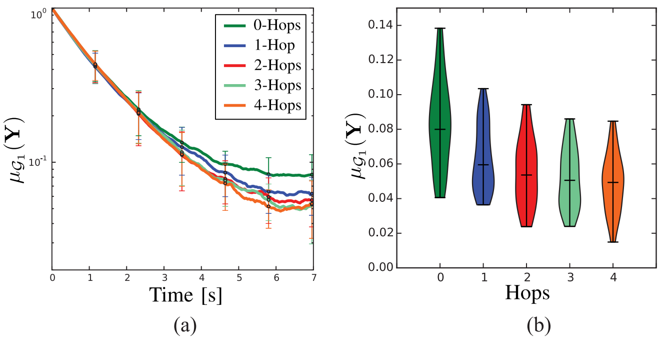


Fig. 9. (a) Ratio of misplaced traits over time, in log scale. The results are averaged over 4 iterations for 8 different graphs with $M = 8$ nodes. The error bars show the standard deviation. (b) Violin plots with marked median values for data in the final time step of plot (a).

VII. Conclusion

Overall, this work shows how global design variables (such as diversity) are incorporated at a high level of abstraction (i.e., macroscopic level), to produce optimal controllers that can also be implemented in decentralized form, and that are able to take realistic constraints into account (such as limited communication). By considering the specific problem of distributing a heterogeneous swarm of robots among a set of tasks with the goal of satisfying a desired distribution of robot capabilities among those tasks, we contribute to the understanding of the effects of *diversity* in heterogeneous swarms. We propose a formulation for heterogeneous robot systems through *species* and *traits*, and show how this formulation is used to achieve an optimal distribution of robots by specifying the desired final trait distribution. Using this formulation, we propose a diversity metric based on *minspecies* that indicates how performance is affected by diversity. We show that the more the robot community is diverse, the harder it is to optimize: by adding redundant (noncomplementary) species, we increase the size of the solution space and facilitate the optimization. In particular, we show that this conclusion is valid for two different goals that require specific implementations of our diversity metric. The latter implementations are based on specializations of the minspecies, and are referred to as *eigenspecies* and *coverspecies*.

Our method consists of a constrained optimization problem, for which we find a computationally efficient solution that is capable of producing fast convergence times, even for large numbers of species and traits. Indeed, our computation is fully scalable with respect to the number of robots, number of species, and number of traits. Building on this result, we propose a real-time optimization method that enables an online adaptation of transition rates as a function of the state of the current robot distribution. We evaluate our methods by means of microscopic simulations, and show how the performance of the latter is well predicted by the macroscopic equations.

Future work will consider the development of methods that automatically generate desired trait distributions as a function of underlying real-world problems. We also intend to develop methods that enable local online estimation of changing desired trait distributions, as a result of dynamic environments and fluctuating needs.

APPENDIX

The optimization in (15) is reformulated for goal $\mathcal{G}_2$ with

$$\mathbf{E_1} = \left\| \max(\mathbf{Y}^\star - \mathbf{Y}, 0) \right\|_F^2 \qquad (30)$$

and the derivative [analogous to (17)] is

$$\frac{\partial \mathbf{E_1}}{\partial e^{\mathbf{K}^{(s)}\tau}} = -2 \left[\max(\mathbf{Y}^\star - \mathbf{Y}, 0)\right] \cdot \left[\mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)}\right]^\top . \qquad (31)$$

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their constructive feedback in improving this paper.

REFERENCES

[1] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm Robotics: A review from the swarm engineering perspective," *Swarm Intell.*, vol. 7, pp. 1–41, 2013.

[2] G. Beni, "From swarm intelligence to swarm robotics," in *Distributed Artificial Intelligence Architecture and Modelling*. Berlin, Germany: Springer, 2005, pp. 1–9.

[3] M. Dorigo *et al.*, "Swarmanoid: A novel concept for the study of heterogeneous robotic swarms," *IEEE Robot. Autom. Mag.*, vol. 20, no. 4, pp. 60–71, Dec. 2013.

[4] M. A. Hsieh *et al.*, "Adaptive teams of autonomous aerial and ground robots for situational awareness," *J. Field Robot.*, vol. 24, pp. 991–1014, 2007.

[5] P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler, "Sensor planning for a symbiotic UAV and UGV system for Precision Agriculture," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 5321–5326.

[6] E. G. Jones, B. Browning, M. B. Dias, B. Argall, and M. Veloso, "Dynamically formed heterogeneous robot teams performing tightly coordinated tasks," in *Proc. Int. Conf. Robot. Autom.*, 2006, pp. 570–575.

[7] T. Balch, "Hierarchic social entropy: An information theoretic measure of robot group diversity," *Auton. Robots*, vol. 8, pp. 209–237, 2000.

[8] W. Abbas and M. Egerstedt, "Characterizing heterogeneity in cooperative networks from a resource distribution view-point," *Commun. Inf. Syst.*, vol. 14, pp. 1–22, 2014.

[9] H. Hamann, G. Valentini, and M. Dorigo, "Population coding: A new design paradigm for embodied distributed systems," in *Proc. Int. Conf. Swarm Intell.*, 2016, pp. 173–184.

[10] N. Michael, J. Fink, S. Loizou, and V. Kumar, "Architecture, abstractions, and algorithms for controlling large teams of robots: Experimental testbed and results," in *Robotics Research; Springer Tracts in Advanced Robotics*. New York, NY, USA: Springer, 2011, pp. 409–419.

[11] B. P. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," *Int. J. Robot. Res.*, vol. 23, no. 9, pp. 939–954, 2004.

[12] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*. Berlin, Germany: Springer, 2000.

[13] V. Chvatal, "A greedy heuristic for the set-covering problem," *Math. Oper. Res.*, vol. 4, no. 3, pp. 233–235, Aug. 1979.

[14] O. Shehory and S. Kraus, "A kernel-oriented model for autonomous-agent coalition-formation in general environments," in *Distributed Artificial Intelligence Architecture and Modelling*. Berlin, Germany: Springer, 2005, pp. 31–45.

[15] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multi-robot coordination: A survey and analysis," *Proc. IEEE*, vol. 94, no. 7, pp. 1257–1270, Jul. 2006.

[16] S. Berman, Á. Halasz, M. A. Hsieh, and V. Kumar, "Optimized stochastic policies for task allocation in swarms of robots," *IEEE Trans. Robot.*, vol. 25, no. 4, pp. 927–937, Aug. 2009.

[17] M. A. Hsieh, Á. Halasz, S. Berman, and V. Kumar, "Biologically inspired redistribution of a swarm of robots among multiple sites," *Swarm Intell.*, vol. 2, no. 2/4, pp. 121–141, 2008.

[18] L. Matthey, S. Berman, and V. Kumar, "Stochastic strategies for a swarm robotic assembly system," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2009, pp. 1953–1958.

[19] G. Valentini, H. Hamann, and M. Dorigo, "Global-to-local design for self-organized task allocation in swarms," IRIDIA, Universite Libre de Bruxelles, Bruxelles, Belgium, Tech. Rep. TR/IRIDIA/2016-002, 2016.

[20] A. Prorok, M. A. Hsieh, and V. Kumar, "Fast redistribution of a swarm of heterogeneous robots," in *Proc. Int. Conf. Bio-Inspired Inf. Commun. Technol.*, 2015, pp. 249–255.

[21] A. Prorok, A. M. Hsieh, and V. Kumar, "Adaptive redistribution of a swarm of heterogeneous robots," *Acta Polytechnica*, vol. 56, no. 1, pp. 67–75, 2016.

[22] A. Prorok, A. M. Hsieh, and V. Kumar, "Formalizing the impact of diversity on performance in a heterogeneous swarm of robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 5364–5371.

[23] O. L. Petchey and K. J. Gaston, "Functional diversity (FD), species richness and community composition," *Ecol. Lett.*, vol. 5, no. 3, pp. 402–411, 2002.

[24] G. W. Stewart, *Matrix Algorithms I. Basic Decompositions*. Philadelphia, PA, USA: SIAM, 1998.

[25] J. D. Kalbfleisch and J. F. Lawless, "The analysis of panel data under a Markov assumption," *J. Amer. Statist. Assoc.*, vol. 80, pp. 863–871, 1985.

[26] J. Demmel, I. Dumitriu, and O. Holtz, "Fast linear algebra is stable," *Numerische Mathematik*, vol. 108, no. 1, pp. 59–91, 2007.

[27] V. Y. Pan and Z. Q. Chen, "The complexity of the matrix eigenproblem," in *Proc. ACM Symp. Theory Comput.*, 1999, pp. 507–516.

[28] M. Milam, R. Franz, J. Hauser, and M. Murray, "Receding horizon control of vectored thrust flight experiment," *IEE Proc. Control Theory Appl.*, vol. 152, no. 3, pp. 340–348, May 2015.

[29] A. Prorok, N. Correll, and A. Martinoli, "Multi-level spatial modeling for stochastic distributed robotic systems," *Int. J. Robot. Res.*, vol. 30, pp. 574–589, Apr. 2011.

[30] G. Valentini and H. Hamann, "Time-variant feedback processes in collective decision-making systems: Influence and effect of dynamic neighborhood sizes," *Swarm Intell.*, vol. 9, no. 2/3, pp. 153–176, Jun. 2015.

[31] T. W. Mather and M. Ani Hsieh, "Macroscopic modeling of stochastic deployment policies with time delays for robot ensembles," *Int. J. Robot. Res.*, vol. 30, no. 5, pp. 590–600, Apr. 2011.

[32] T. W. Mather and M. A. Hsieh, "Distributed robot ensemble control for deployment to multiple sites," in *Proc. Robot. Sci. Syst.*, Los Angeles, CA, USA, Jun. 2011. doi:10.15607/RSS.2011.VII.028

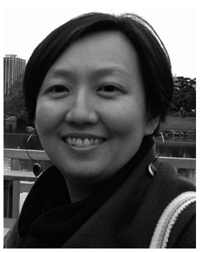
[33] T. W. Mather and M. A. Hsieh, "Synthesis and analysis of distributed ensemble control strategies for allocation to multiple tasks," *Robotica*, vol. 32, no. 2, pp. 177–192, Dec. 2013.

[34] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440–442, 1998.

Amanda Prorok received the Ph.D. degree from Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, where she addressed the topic of localization with ultra-wideband sensing for robotic networks.

She is a Postdoctoral Researcher in the General Robotics, Automation, Sensing and Perception Laboratory, University of Pennsylvania, Philadelphia, PA, USA. Her current research focuses on heterogeneous robot networks.

Dr. Prorok's dissertation was awarded the Asea Brown Boveri (ABB) award for the best thesis at EPFL in the fields of computer sciences, automatics and telecommunications.



M. Ani Hsieh received the B.S. degree in engineering and the B.A. degree in economics from Swarthmore College, Swarthmore, PA, USA, in 1999, and the Ph.D. degree in mechanical engineering from the University of Pennsylvania, Philadelphia, PA, USA, in 2007.

She is an Associate Professor in the Mechanical Engineering & Mechanics Department, Drexel University, Philadelphia, PA, USA. Her research interest focuses on developing a general control and coordination framework for distributed sensing and monitoring of dynamic and uncertain environments by mobile robot teams.

Dr. Hsieh received the 2012 Office of Naval Research (ONR) Young Investigator Award and the 2013 National Science Foundation (NSF) CAREER Award.

Vijay Kumar received the B.Tech. degree from the Indian Institute of Technology, Kanpur, India, and the Ph.D. degree from The Ohio State University, Columbus, OH, USA, in 1987.

He is the Nemirovsky Family Dean of Penn Engineering with appointments in the Departments of Mechanical Engineering and Applied Mechanics, Computer and Information Science, and Electrical and Systems Engineering at the University of Pennsylvania. He has been on the Faculty in the Department of Mechanical Engineering and Applied Mechanics with a secondary appointment in the Department of Computer and Information Science at the University of Pennsylvania since 1987. He was the Deputy Dean for Research in the School of Engineering and Applied Science from 2000 to 2004. He was the Director of the General Robotics, Automation, Sensing and Perception Laboratory, a multidisciplinary robotics and perception laboratory, from 1998 to 2004. He was the Chairman of the Department of Mechanical Engineering and Applied Mechanics from 2005 to 2008. He was the Deputy Dean for Education in the School of Engineering and Applied Science from 2008 to 2012. He then served as the Assistant Director of Robotics and Cyber Physical Systems, White House Office of Science and Technology Policy (2012–2013). His research interests include robotics, specifically multirobot systems, and micro aerial vehicles.

Dr. Kumar is a Fellow of the American Society of Mechanical Engineers (2003), a Fellow of the Institute of Electrical and Electronic Engineers (2005), and a member of the National Academy of Engineering (2013). He has been on the editorial boards of IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, *ASME Journal of Mechanical Design*, *ASME Journal of Mechanisms and Robotics*, and *Springer Tract in Advanced Robotics*. He currently serves as the Editor of *ASME Journal of Mechanisms and Robotics* and as an Advisory Board Member of *AAAS Science Robotics Journal*. He received the 1991 National Science Foundation Presidential Young Investigator award, the 1996 Lindback Award for Distinguished Teaching (University of Pennsylvania), the 1997 Freudenstein Award for significant accomplishments in mechanisms and robotics, the 2012 ASME Mechanisms and Robotics Award, the 2012 IEEE Robotics and Automation Society Distinguished Service Award, the 2012 World Technology Network Award, and the 2014 Engelberger Robotics Award. He has won best paper awards at DARS 2002, ICRA 2004, ICRA 2011, RSS 2011, and RSS 2013, and has advised doctoral students who have won Best Student Paper Awards at ICRA 2008, RSS 2009, and DARS 2010.