Acta Polytechnica **56**(1):67–75, 2016
doi:10.14311/APP.2016.56.0067

available online at http://ojs.cvut.cz/ojs/index.php/ap

ADAPTIVE DISTRIBUTION OF A SWARM OF HETEROGENEOUS ROBOTS

Amanda Prorok*, M. Ani Hsieh, Vijay Kumar

General Robotics, Automation, Sensing & Perception (GRASP) Laboratory, University of Pennsylvania, Philadelphia, USA

* corresponding author: prorok@seas.upenn.edu

Abstract. We present a method that distributes a swarm 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, where each species (robot type) is defined by the traits (capabilities) that it owns. Our method is based on a continuous abstraction of the swarm at a macroscopic level as we model robots switching between tasks. We formulate an optimization problem that produces an optimal set of transition rates for each species, so that the desired trait distribution is reached as quickly as possible. Since our method is based on the derivation of an analytical gradient, it is very efficient with respect to state-of-the-art methods. Building on this result, we propose a real-time optimization method that enables an online adaptation of transition rates. Our approach is well-suited for real-time applications that rely on online redistribution of large-scale robotic systems.

Keywords: heterogeneous multi-robot systems; swarm robotics; stochastic systems; task allocation.

1. 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., swarms of robots. However, the smaller we design our platforms, the more stringent the tradeoffs we need to make with respect to endowed capabilities. As a consequence, investigators are composing their robot systems with multiple, heterogeneous types of robots in order to tackle increasingly challenging tasks [1, 2]. Our premise is that, in a swarm of robots, 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.

In this work, our objective is to distribute a swarm of heterogeneous robots as quickly and efficiently as possible among tasks that require specialized competences. This objective is part of a larger vision to develop control and coordination strategies for teams of heterogeneous robots with specific 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, altogether more than a single one could carry on its own. Initially, we consider tasks that are to be performed in parallel, continuously, and independently, yet we develop a framework that can easily be extended to accommodate temporal and precedence constraints. Instances of information gathering lend themselves naturally to our problem formulation, with applications to surveillance, environmental monitoring, and situational awareness [3–5].

Given a set of tasks, and knowledge about what the task requirements are, our problem considers which robots should be allocated to which tasks. This problem is an instance of the *MT-MR-TA: Multi-Task Robots, Multi-Robot Tasks* problem [6], and can be reformulated as a set-covering problem that stems from combinatorial optimization. This concept considers subsets of robots in a multi-robot system, and pairs them optimally (given a cost function) to tasks. This problem is strongly NP-hard [7]. A number of heuristic algorithms have been proposed. However, the running times of these algorithms are functions of the sizes of the feasible subsets (of robots paired to a task), and hence, become very expensive for large robot teams and swarms. Furthermore, the algorithms are penalized when multiple robot-subset-to-task combinations are feasible (this is the case, for example, when robots have overlapping capabilities).

These algorithms are not suitable for large-scale systems, such as robot swarms. In particular, for systems that are required to adapt to changing task requirements online, we need to consider algorithms that are efficient and implementable on mobile platforms that run them in real-time. Our method builds on previous work done in the domain of dynamic redistribution of homogenous robot swarms [8–10]. We consider a strategy that is scalable in the number of robots and their capabilities, and is robust to changes in the robot population. The key property of this strategy is its inherently decentralized architecture, with robots switching between behaviors (tasks) stochastically. This model is inspired by previous work in the swarm-robotic domain that explores self-organized behavior of natural systems [11, 12].

The present work focuses on the optimization of transition rates that enables the robot swarm to converge quickly to a configuration that satisfies a desired trait distribution. The key difference between our work and previous work is that we formulate our desired state as a distribution of traits among sites, 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 the works presented by [8] and [13]). Instead, we 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. Indeed, we will later show that there are cases where multiple final robot distributions satisfy the desired trait distribution. In such cases, state-of-the-art strategies require that we first determine the best final robot distribution (one that can be reached the fastest), and subsequently optimize transition rates using methods such as in [8].

2. Problem Formulation

Heterogeneity and diversity are core concepts of this work. To develop our formalism, we will borrow terminology from biodiversity literature [14, 15]. 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 work, we will consider binary instantiations of traits (corresponding to the presence or absence of a given trait in a species). As an example, one trait might consider the presence/absence of a particular sensor, such as a camera or laser range finder. Another trait might consider the capability of fitting through a passageway with a fixed width. In this work, we assume that the tasks have been encoded through binary characteristics that represent the skill sets critical to task completion.

2.1. 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 vector $\mathbf{q}^{(s)} = [q_1^{(s)}, q_2^{(s)}, ..., q_U^{(s)}]$. We can then define a $S \times U$ matrix $\mathbf{Q}$, with rows $\mathbf{q}^{(s)}$:

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

We model the interconnection topology of the M sites via a directed graph, $\mathcal{G} = (\mathcal{E}, \mathcal{V})$ where the set of vertices, $\mathcal{V}$, represents task sites $\{1, \ldots, M\}$ 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. We assume the graph $\mathcal{G}$ is a strongly connected graph, i.e., a path exists between any pair of vertices (in contrast to a *fully* connected graph, where an edge exists between any pair of vertices). In other words, if the nodes in our graph are physically distributed sites, then, via some road, we can reach any other site. We assign every edge in $\mathcal{E}$ a transition rate, $k_{ij}^{(s)} > 0$, where $k_{ij}^{(s)}$ defines the transition probability per unit time for one robot of species s at site i to switch to site j. Here $k_{ij}^{(s)}$ is a stochastic transition rule. We assume every robot has knowledge of $\mathcal{G}$ as well as all the transition rates of its species $k_{ij}^{(s)}$. We note that this information is represented by a small number of values (at most $M^2 \cdot S$ values, or much less if the graph is sparse), and needs to be transmitted to the robots only once, at the start of a run. We impose a limitation on the maximum rate of each edge with $k_{ij}^{(s)} < k_{ij,\max}^{(s)}$. These values can be determined by applying system identification methods on the actual system. For example, in a system where nodes represent physically distributed sites, the transition rate represents the rate with which a specific path is chosen. This value can depend on observed factors, such as typical road congestion or the condition of the terrain.

The distribution of the robots belonging to a species s at time t is described by a vector $\mathbf{x}^{(s)}(t) = [\mathbf{x}_1^{(s)}(t), ..., \mathbf{x}_M^{(s)}(t)]^\top$. Then, if $\mathbf{x}^{(s)}$ are the columns of $\mathbf{X}(t)$, and $\mathbf{q}^{(s)}$ are the rows of $\mathbf{Q}$, we have the $M \times U$ matrix $\mathbf{Y}$ that describes the distribution of traits on sites. For time t this relationship is given by

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

As we will see in Section 2.3, there may be several robot distributions $\mathbf{X}(t)$ that satisfy this equation for a given $\mathbf{Y}(t)$.

2.2. Problem Statement

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

$$\frac{\mathrm{d}\mathbf{x}_i^{(s)}}{\mathrm{d}t} = \sum_{\forall j | (i,j) \in \mathcal{E}} k_{ji} \mathbf{x}_i^{(s)}(t) - \sum_{\forall j | (i,j) \in \mathcal{E}} k_{ij} \mathbf{x}_i^{(s)}(t). \tag{2}$$

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

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

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

These two properties result in the following definition:

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

Since the total number of robots and the number of robots per species is conserved, the system in Eq. 3 is subject to the constraint

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

The goal is to find an optimal rate matrix $\mathbf{K}^{(s)\bigstar}$ for each species s so that we have

$$\bar{\mathbf{Y}} = \bar{\mathbf{X}} \cdot \mathbf{Q}. \tag{7}$$

In other words, the task is to redeploy the robots of each species configured according to $\mathbf{X}(0)$ initially, so that a desired trait configuration $\bar{\mathbf{Y}}$ is reached. In doing this, we reach a robot configuration $\bar{\mathbf{X}}$ that satisfies Eq. 1, subject to Eq. 6.

2.3. System Properties

Since we describe the desired configuration of our system through $\bar{\mathbf{Y}}$, the final robot distribution $\bar{\mathbf{X}}$ is not known a priori. Given knowledge of $\mathbf{Q}$ we can infer the following properties: If a solution to Eq. 7 subject to Eq. 6 exists, then

(1.) If $\text{rank}(\mathbf{Q}) < S$, the system is underdetermined, and an infinite number of solutions $\bar{\mathbf{X}}$ will satisfy Eq. 7. In other words, at least one species in the system can be replaced by a combination of the other species.

(2.) If $\text{rank}(\mathbf{Q}) = S$, only one solution $\bar{\mathbf{X}}$ exists that satisfies Eq. 7. In other words, no species in the system can be expressed as a combination of the other species.

We note that solving case (1) is relevant when we embed redundancy into the robot system, and case (2) is relevant when we consider fully complementary robot species.

3. Methodology

In this section, we describe our methodology for obtaining an optimal transition matrix $\mathbf{K}^{(s)\bigstar}$ for each species so that the desired trait distribution is reached. Berman et al. [8] present an exposé of optimization methods that can be used to obtain optimal transition rates for a homogenous robot swarm that is required to converge to a desired distribution. Two general approaches are considered: convex optimization and stochastic optimization. The convex optimization approach requires knowledge of the desired final robot distribution. Indeed, our problem formulation specifies a desired trait distribution $\bar{\mathbf{Y}}$ without explicit definition of the final robot distribution $\bar{\mathbf{X}}$. Hence, convex optimization strategies as in [8] are not applicable to our problem, unless $\text{rank}(\mathbf{Q}) = S$, and we can infer $\bar{\mathbf{X}}$. Given this rationale, we choose an optimization approach that is able to find optimal transition rates with knowledge of $\bar{\mathbf{Y}}$ and $\mathbf{X}(0)$, without knowledge of $\bar{\mathbf{X}}$. Although fully stochastic schemes such as Metropolis optimization have been shown to produce similar results [8], they are not computationally efficient, and are ill-suited to real-time applications. In the following, we present a differentiable objective function that can be efficiently minimized through gradient descent techniques. We show that our method has a computational complexity that is well-suited to real-time applications. Additionally, our method explicitly minimizes the convergence time of $\mathbf{K}^{(s)}$, unlike the convex optimization methods presented in [8] which approximate $\mathbf{K}^{(s)}$ with a symmetric equivalent (forcing bidirectionally equal transition rates between sites).

3.1. Design of Optimal Transition Rates

We combine the solution of the linear ordinary differential equation, Eq. 3, and Eq. 7 to obtain the solution for a desired trait distribution

$$\bar{\mathbf{Y}} = \sum_{s=1}^{S} e^{\mathbf{K}^{(s)\bigstar}\tau} \mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)}, \tag{8}$$

where τ is the time at which the desired state is reached. We design our objective function as follows. To find the values of $\mathbf{K}^{(s)\bigstar}$ for all species for the given initial configuration, $\mathbf{X}(0)$, we consider the following optimization

$$\begin{aligned} &\text{minimize } \mathcal{J}^{(1)} = \left\| \bar{\mathbf{Y}} - \sum_{s=1}^{S} e^{\mathbf{K}^{(s)}\tau} \mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)} \right\|_F^2 \\ &\text{such that } k_{ij}^{(s)} < k_{ij,\max}^{(s)}, \end{aligned} \tag{9}$$

which formulates that a minimum cost is found when the final trait distribution corresponds to the desired trait distribution, subject to maximum transition rates $k_{ij,\max}^{(s)}$. The notation $\mathbf{x}_0^{(s)}$ is shorthand for $\mathbf{x}^{(s)}(0)$. The operator $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. There is no closed-form solution to the optimization problem in Eq. 9, but we can use the derivatives of $\mathcal{J}^{(1)}$ with respect to the parameters to perform gradient descent. So that the implementation of the optimization function is efficient, it is important that the function is differentiable and that an analytical gradient can be computed. By applying the chain rule, the derivative of our objective function with respect to the transition matrix $\mathbf{K}^{(s)}$ is

$$\frac{\partial \mathcal{J}^{(1)}}{\partial \mathbf{K}^{(s)}} = \frac{\partial \mathcal{J}^{(1)}}{\partial e^{\mathbf{K}^{(s)}\tau}} \cdot \frac{\partial e^{\mathbf{K}^{(s)}\tau}}{\partial \mathbf{K}^{(s)}\tau} \cdot \frac{\partial \mathbf{K}^{(s)}\tau}{\partial \mathbf{K}^{(s)}}. \tag{10}$$

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

$$\frac{\partial \mathcal{J}^{(1)}}{\partial e^{\mathbf{K}^{(s)}\tau}} = 2\Big[\sum_{r\in S} e^{\mathbf{K}^{(r)}\tau} \mathbf{x}_0^{(r)} \cdot \mathbf{q}^{(r)} - \bar{\mathbf{Y}}\Big] \cdot \big[\mathbf{x}_0^{(s)} \cdot \mathbf{q}^{(s)}\big]^\top. \tag{11}$$

The derivation of the 2nd element of Eq.10 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 [16] to our problem, and write the gradient of our cost function as

$$\frac{\partial \mathcal{J}^{(1)}}{\partial \mathbf{K}^{(s)}} = \mathbf{V}^{-\mathbf{1}^\top}\left[\mathbf{V}^\top \frac{\partial \mathcal{J}^{(1)}}{\partial e^{\mathbf{K}^{(s)}\tau}} \mathbf{V}^{-\mathbf{1}^\top} \odot \mathbf{W}(\tau)\right]\mathbf{V}^\top \tau, \tag{12}$$

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]

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

3.2. Optimization of Convergence Time

The cost function in Eq. 9 does not consider convergence time τ as a variable. By adding a term that penalizes high convergence time values, we can compute transition rates that explicitly optimize convergence time. The modified objective function is

$$\begin{aligned} &\text{minimize } \mathcal{J}^{(2)} = \mathcal{J}^{(1)} + \alpha\tau^2 \\ &\text{such that } k_{ij}^{(s)} < k_{ij,\max}^{(s)} \text{ and } \tau > 0, \end{aligned} \tag{13}$$

and $\alpha > 0$. By increasing α, we increase the importance of the convergence time (by penalizing high values of τ). The derivative with respect to the transition rates is

$$\frac{\partial \mathcal{J}^{(2)}}{\partial \mathbf{K}^{(s)}} = \frac{\partial \mathcal{J}^{(1)}}{\partial \mathbf{K}^{(s)}}. \tag{14}$$

In order to optimize the convergence time, we need the derivative with respect to τ. This derivative is computed analogously to the derivative with respect to $\mathbf{K}^{(s)}$ (confer Eq. 12). We have

$$\frac{\partial \mathcal{J}^{(2)}}{\partial \tau} = \frac{\partial \mathcal{J}^{(1)}}{\partial \tau} + 2\alpha\tau \tag{15}$$

with

$$\frac{\partial \mathcal{J}^{(1)}}{\partial \tau} = \sum_{s=1}^{S} \mathbf{1}^\top \mathbf{V}^{-\mathbf{1}^\top} \mathbf{A}_1 \mathbf{V}^\top \cdot \mathbf{K}^{(s)}\mathbf{1} \tag{16}$$

and

$$\mathbf{A}_1 = \mathbf{V}^\top \frac{\partial \mathcal{J}^{(1)}}{\partial e^{\mathbf{K}^{(s)}\tau}} \mathbf{V}^{-\mathbf{1}^\top} \odot \mathbf{W}(\tau). \tag{17}$$

The optimization of Eq. 13 produces transition rates that lead to the desired trait distribution quickly, but there is no guarantee that this is the steady state of $\mathbf{K}^{(s)}$. 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 $t^\bigstar$ remains near-constant. Hence, we modify our cost function in Eq. 13 as follows.

$$\begin{aligned} &\text{minimize } \mathcal{J}^{(3)} = \mathcal{J}^{(2)} + \beta \sum_{s=1}^{S} \left\| e^{\mathbf{K}^{(s)}\tau}\mathbf{x}_0^{(s)} \right. \\ &\qquad \left. - e^{\mathbf{K}^{(s)}(\tau+\nu)}\mathbf{x}_0^{(s)} \right\|_2^2 \\ &\text{such that } k_{ij}^{(s)} < k_{ij,\max}^{(s)} \text{ and } \tau > 0, \end{aligned} \tag{18}$$

and $\beta > 0$. The additional term in our cost function allows us to ensure that the robot distribution reached by employing $\mathbf{K}^{(s)\bigstar}$ remains near-constant for arbitrarily long time intervals ν. This is possible because our model in Eq. 3 is stable [9], and the difference between the current robot distribution and the one at steady state can only decrease monotonically over time. By increasing the value of β, the difference of the robot distributions at times τ and $\tau + \nu$ is decreased. In other words, as we will see in Section 4, the trait distribution corresponding to the steady state robot distribution of $\mathbf{K}^{(s)}$ gets arbitrarily close to the desired trait distribution $\bar{\mathbf{Y}}$ as β increases (the same is true when we increase ν). Note that when $\alpha = 0$, β should not be infinitely large, as in this case, $\mathbf{K}^{(s)\bigstar} = \mathbf{0}$. However, for all practical purposes β is bounded and $\alpha > 0$.

Let us refer to this additional third term of $\mathcal{J}^{(3)}$ (and second term of Eq. 18) as $\mathcal{J}^{(3,3)}$. Then, the derivative of the new objective function with respect to the transition rates can be expressed as

$$\frac{\partial \mathcal{J}^{(3)}}{\partial \mathbf{K}^{(s)}} = \frac{\partial \mathcal{J}^{(2)}}{\partial \mathbf{K}^{(s)}} + \frac{\partial \mathcal{J}^{(3,3)}}{\partial \mathbf{K}^{(s)}}. \tag{19}$$

Again, we apply the chain rule to obtain

$$\begin{aligned} \frac{\partial \mathcal{J}^{(3,3)}}{\partial \mathbf{K}^{(s)}} &= \frac{\partial \mathcal{J}^{(3,3)}}{\partial e^{\mathbf{K}^{(s)}\tau}} \frac{\partial e^{\mathbf{K}^{(s)}\tau}}{\partial \mathbf{K}^{(s)}\tau} \frac{\partial \mathbf{K}^{(s)}\tau}{\partial \mathbf{K}^{(s)}} \\ &\quad - \frac{\partial \mathcal{J}^{(3,3)}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} \frac{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}}{\partial \mathbf{K}^{(s)}(\tau+\nu)} \frac{\partial \mathbf{K}^{(s)}(\tau+\nu)}{\partial \mathbf{K}^{(s)}}. \end{aligned} \tag{20}$$

The outer derivative is

$$\begin{aligned} \frac{\partial \mathcal{J}^{(3,3)}}{\partial e^{\mathbf{K}^{(s)}\tau}} &= \frac{\partial \mathcal{J}^{(3,3)}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} \\ &= 2\beta\left[e^{\mathbf{K}^{(s)}\tau}\mathbf{x}_0^{(s)} - e^{\mathbf{K}^{(s)}(\tau+\nu)}\mathbf{x}_0^{(s)}\right] \cdot {\mathbf{x}_0^{(s)}}^\top. \end{aligned} \tag{21}$$

We apply the same development as in Eq. 12 to obtain the equation

$$\frac{\partial \mathcal{J}^{(3,3)}}{\partial \mathbf{K}^{(s)}} = \mathbf{V}^{-\mathbf{1}^\top} \mathbf{A}_2 \mathbf{V}^\top \cdot \tau - \mathbf{V}^{-\mathbf{1}^\top} \mathbf{A}_3 \mathbf{V}^\top \cdot (\tau + \nu) \tag{22}$$

with

$$\mathbf{A}_2 = \mathbf{V}^\top \cdot \frac{\partial \mathcal{J}^{(3,3)}}{\partial e^{\mathbf{K}^{(s)}\tau}} \cdot \mathbf{V}^{-\mathbf{1}^\top} \odot \mathbf{W}(\tau) \tag{23}$$

[1] Here, we assume that 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 [16]. We note that for most models of interest, however, this is rarely the case.

and

$$\mathbf{A}_3 = \mathbf{V}^\top \cdot \frac{\partial \mathcal{J}^{(3,3)}}{\partial e^{\mathbf{K}^{(s)}(\tau+\nu)}} \cdot \mathbf{V^{-1}}^\top \odot \mathbf{W}(\tau+\nu). \quad (24)$$

For all above cost functions, $z = 1, 2, 3$, the derivative with respect to the off-diagonal elements ij of the matrix $\mathbf{K}^{(s)}$, with $(i, j) \in \mathcal{E}$, is

$$\frac{\partial \mathcal{J}^{(z)}}{\partial \mathbf{K}^{(s)}_{ij}} = \left\{ \frac{\partial \mathcal{J}^{(z)}}{\partial \mathbf{K}^{(s)}} \right\}_{ij} - \left\{ \frac{\partial \mathcal{J}^{(z)}}{\partial \mathbf{K}^{(s)}} \right\}_{jj}, \quad (25)$$

where $\{\cdot\}_{ij}$ denotes the element on row i and column j. The derivative with respect to time τ is analogous. Finally, we summarize our optimization problem as follows:

$$\mathbf{K}^{(s)\bigstar}, \tau^{\bigstar} = \underset{\mathbf{K}^{(s)},\tau}{\operatorname{argmin}} \mathcal{J}^{(3)}, \quad (26)$$

under the constraints shown in Eq. 18. To solve the system, we implement a basin-hopping optimization algorithm [17], which attempts to find the global minimum of a smooth scalar function. Locally, our basin-hopping algorithm uses a quasi-Newton method (namely, the Broyden-Fletcher-Goldfarb-Shanno algorithm [18] with bound constraints).

3.3. Computational Complexity

The computational complexity of computing the gradient of our objective function 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 non-sparse matrices [19][2]. We compute this decomposition only once per optimization (see Eq. 12, where $\mathbf{K}^{(s)} = \mathbf{V}\mathbf{D}\mathbf{V}^{-1}$), for each optimization of $\mathbf{K}^{(s)}$. The second part is dictated by the multiplication of the matrices in Eq. 12, 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 cubically with respect to the number of tasks. Overall, for the results shown in Section 4, 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.

3.4. Continuous Optimization of $\mathbf{K}^{(s)}$

As shown above, the optimization of the objective function $\mathcal{J}^{(3)}$ is efficient and can be performed in real-time. Building on this result, we can implement a continuous, online optimization strategy that allows us to refine the optimal $\mathbf{K}^{(s)\bigstar}$ as a function of the current state. 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. By taking the actual robot distribution into account, it 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. Formally, we write our control policy as

$$\mathbf{K}^{(s)\bigstar}(t), \tau^{\bigstar}(t) = \underset{\mathbf{K}^{(s)},\tau}{\operatorname{argmin}} \tilde{\mathcal{J}}^{(3)}(\mathbf{X}(t_p)) \quad (27)$$
$$\text{with } t_p \le t < t_p + \delta$$
$$t_p \in k\delta, k \in \mathbb{N},$$

where we rewrite our cost equation as a function of the robot distribution

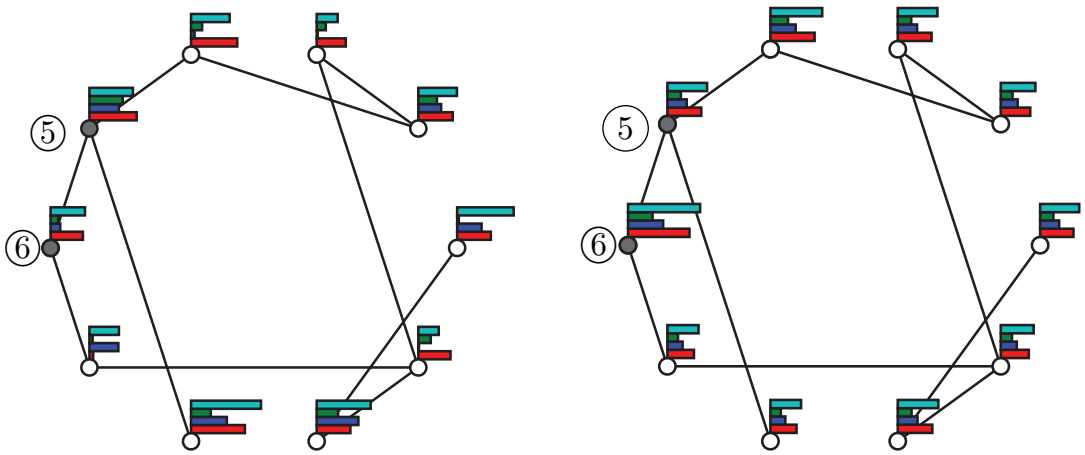
$$\tilde{\mathcal{J}}^{(3)}(\mathbf{X}) = \left\| \bar{\mathbf{Y}} - \sum_{s=1}^{S} e^{\mathbf{K}^{(s)}\tau} \mathbf{x}^{(s)} \cdot \mathbf{q}^{(s)} \right\|_F^2 + \alpha\tau^2 \quad (28)$$
$$+ \beta \sum_{s=1}^{S} \left\| e^{\mathbf{K}^{(s)}\tau} \mathbf{x}^{(s)} - e^{\mathbf{K}^{(s)}(\tau+\nu)} \mathbf{x}^{(s)} \right\|_2^2.$$

Note that $\tilde{\mathcal{J}}^{(3)}(\mathbf{X}(0))$ is equal to $\mathcal{J}^{(3)}$. In practice, if δ is small with respect to the rates at which robots transition between sites (cf. $k_{ij,\max}$), we set $\beta = 0$, since the continuous optimization of $\mathbf{K}^{(s)}$ enforces a current state that is close to the desired state, irrespective of the steady-state distribution. For cases where the optimization time becomes large (implying that δ also becomes large), we either need to set $\beta > 0$, or use a strategy that accounts for computation delay, such as those presented in [21]. 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).

4. Results

Previous work has shown the benefit of validating methods over multiple, complementary levels of abstraction (sub-microscopic, microscopic, and macroscopic) [22]. In the present work, we propose an evaluation of our methods on two levels: microscopic and macroscopic. Indeed, the most efficient way of simulating the swarm of robots is by considering a continuous macroscopic model, derived directly from the ordinary differential equation, Eq. 3. In order to validate the control policy at a lower level of abstraction, we also implement a discrete microscopic model that emulates the behavior of individual robot controllers. This 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 site i transitions to site

[2] In the special case where all eigenvalues are distinct, the eigenvalue decomposition can be reduced to $O(M^{2.376} \log(M))$ [20].



(a) (b)

Figure 1. A strongly connected instance of a graph with 10 nodes representing spatially distributed sites (nodes) and their paths (edges). The system includes 4 traits. The trait abundance at each node is represented by a bar plot. Nodes 5 and 6 are highlighted. (a) Initial configuration (b) Desired final configuration.

j according to probability $p_{ij}^{(s)}$ that is an element of the matrix $\mathbf{P}^{(s)} = e^{\mathbf{K}^{(s)}\Delta T}$, where ΔT is the duration of one time-step. Running multiple iterations of the microscopic model enables us to capture the stochasticity resulting from our control system.

Our performance metric considers the degree of convergence to $\bar{\mathbf{Y}}$, expressed by the fraction of misplaced traits

$$\mu(\mathbf{Y}) = \frac{\|(\mathbf{Y} - \bar{\mathbf{Y}})\|_1}{\|\mathbf{Y}\|_1}. \tag{29}$$

We say that one system converges faster than another if it takes less time for $\mu(\mathbf{Y})$ to decrease to some relative error μ_{thresh}, such as $\mu_{\text{thresh}} = 5\%$. Similar performance metrics have been proposed in [8–10].

We will consider three optimization methods, two of which stem from this paper, and one of which stems from [8]:

Fixed-NC — We consider the time-constant, non-convex optimization problem posed in Eq. 26, with $\alpha = 1$, $\beta = 5$, and $\nu = 2$, producing a fixed $\mathbf{K}^{(s)\bigstar}$ for each species. These values were not tuned in any way to improve performance.

Adaptive-NC — We consider the time-continuous optimization problem in Eq. 27, with $\alpha = 1$, $\beta = 0$, and a sampling time $\delta = 0.08$ s, producing an adaptive control policy $\mathbf{K}^{(s)\bigstar}(t)$ for each species.

Fixed-C — We adapt the convex optimization method presented in [8], denoted in the latter work as **[P1]**. This adapted method optimizes the asymptotic convergence rate (of a system of homogenous robots) by minimizing 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 $\bar{\mathbf{X}}$, we compute a random instantiation of $\bar{\mathbf{X}}$ that satisfies the desired trait distribution defined by Eq. 7. We note that in practical applications, computing a good instantiation of $\bar{\mathbf{X}}$ is not trivial.

For all optimization methods, we set $k_{ij,\max}^{(s)} = 2\,[\text{s}^{-1}]$.

4.1. Example

To illustrate our method, we consider an example of $N = 3893$ robots moving between 10 sites. We sample a random initial robot distribution $\mathbf{X}(0)$, and generate a random, feasible desired trait distribution $\bar{\mathbf{Y}}$. The initial trait distribution is visualized in Fig. 1(a), and the desired trait distribution is visualized in Fig. 1(b). The graph is generated randomly according to the Watts-Strogatz model [23] (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 3 species and 4 traits, and is defined as follows:

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

with

$$\mathbf{X}^\top \cdot \mathbf{1} = [987, 1490, 1416]^\top \tag{30}$$

We solve the system for $\bar{\mathbf{Y}}$ as shown in Fig. 1(b). We show the evolution of the robot distribution in Fig. 2(a) and trait distribution in Fig. 2(b)—to avoid clutter, we plot two selected sites (5 and 6) and one selected trait (corresponding to the 4th trait, shown as the top bar in cyan in Fig. 1). The plots demonstrate a good agreement between the macroscopic model and the discrete microscopic model.

Fig. 3 shows the ratio of misplaced traits $\mu(\mathbf{Y})$ over time for the initial and desired trait distributions depicted in Fig. 1. The macroscopic model demonstrates that the system successfully achieves a negligible error. We run 20 iterations of the discrete microscopic model, once with method **Fixed-NC**, and once with

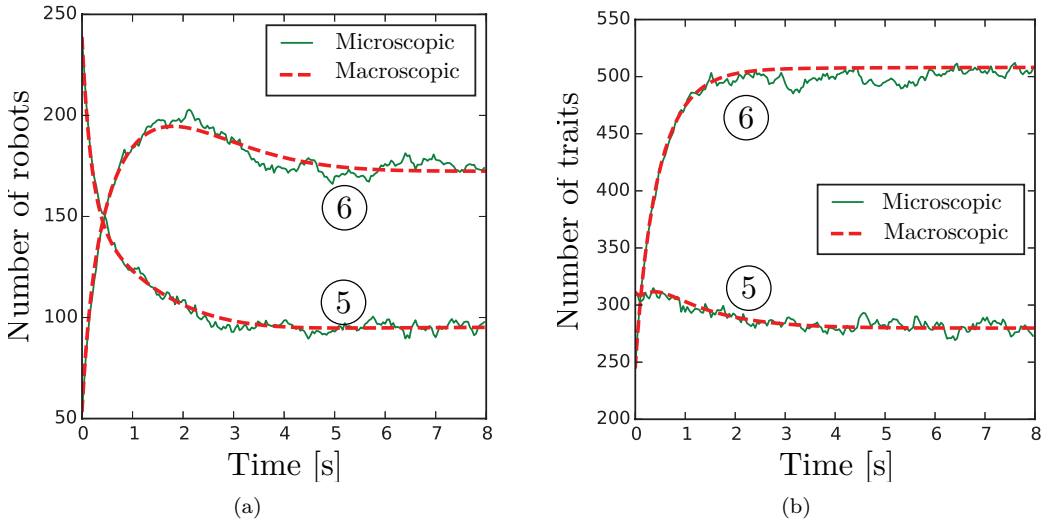


FIGURE 2. The plot shows the macroscopic model as well as the average over 20 iterations of the microscopic model, on the graph shown in Fig. 1. We plot values for nodes 5 and 6, which are highlighted in the graph in Fig. 1. (a) Number of robots of species 1 present at nodes 5 and 6 (b) Number of trait 4 (top bar in cyan in Fig. 1) present at nodes 5 and 6.

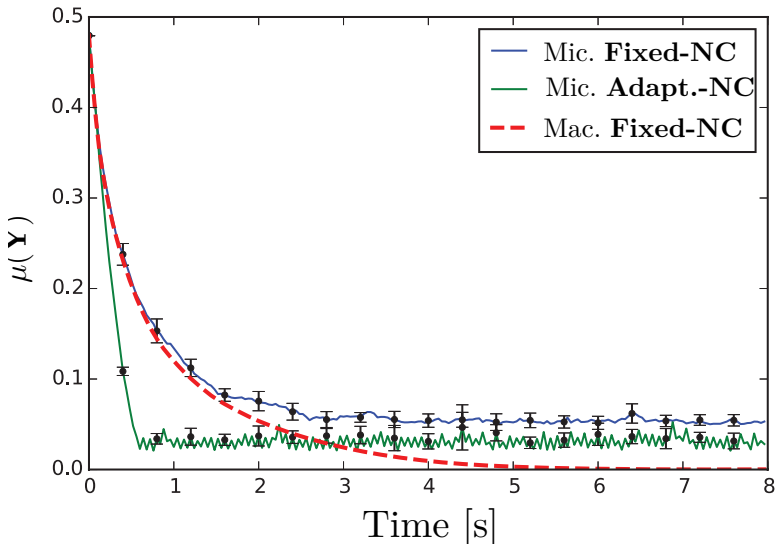


FIGURE 3. Ratio of misplaced traits over time for the initial and desired trait distributions depicted in Fig. 1. The simulation was run with 3893 robots and a total of 7786 present traits. The plot shows the macroscopic model as well as the average over 20 iterations of the microscopic model with and without continuous optimization of transition rates. The errorbars show the standard deviation.

method **Adaptive-NC**. Since the adaptive optimization method takes the current robot configuration into account, it produces transition rates that lead to the desired configuration faster. Also, since it continuously optimizes toward the desired final trait distribution, the error remains low. We note that, due to the stochasticity of the microscopic model, the error ratio (which counts absolute differences) will always be larger than 0.

4.2. Comparison of Methods

We compare the three optimization methods, **Fixed-NC**, **Adaptive-NC**, **Fixed-C** and evaluate their performance with respect to the metric in Eq. 29. We instantiate 40 random graphs with $M = 6$ nodes, and random matrices $\mathbf{Q}$ with $S = 4$ species and $U = 4$ traits, and generate random desired trait distributions $\bar{\mathbf{Y}}$ for each graph. The microscopic model is iterated 4 times on each graph instantiation. For the method **Fixed-C**, we compute a random robot distribution $\bar{\mathbf{X}}$ that satisfies the desired trait distribution. We measure the time $t_{\mu,\text{thresh}}$ at which the system converges to a value $\mu_{\text{thresh}} = 5\%$ of misplaced traits. Since we sample random matrices $\mathbf{Q}$, we obtain different rank values. Fig. 4(a) shows results for $\text{rank}(\mathbf{Q}) = 3$, i.e., a system with redundant species, and Fig. 4(b) shows results for $\text{rank}(\mathbf{Q}) = 4$, i.e., a system with complementary species (cf. the description in Sec. 2.3).

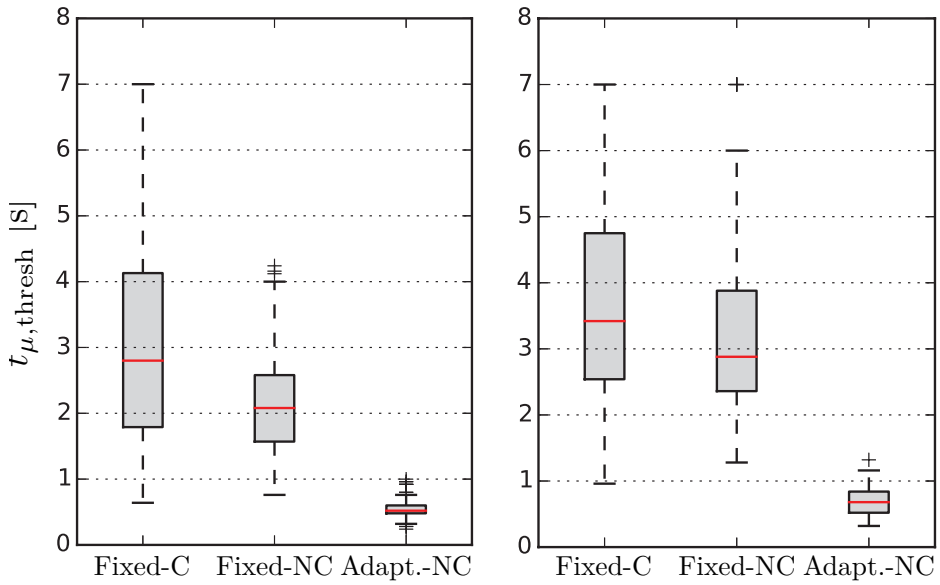


FIGURE 4. The plot shows the convergence time of three optimization methods, evaluated on the microscopic model, with $t_{\mu_{\text{thresh}}}$ for $\mu_{thresh} = 0.05$, for 40 random graphs with $M = 6$ and random matrices $\mathbf{Q}$ with 4 species and 4 traits. The microscopic model was iterated 4 times over each graph instantiation. (a) $\text{rank}(\mathbf{Q}) = 3$ (b) $\text{rank}(\mathbf{Q}) = 4$. The boxplots show the median and the 25th and 75th percentiles.

The plots show that our method **Fixed-NC** is able to improve upon **Fixed-C**: for $\text{rank}(\mathbf{Q}) = 4$ by 16%, and for $\text{rank}(\mathbf{Q}) = 3$ by 25%. The stronger improvement in the lower-rank case points towards the importance of finding a good final robot distribution $\bar{\mathbf{X}}$ when several are possible that satisfy Eq. 7. The results for **Adaptive-NC** confirm the fast convergence towards desired trait distributions, with a 75% improvement over **Fixed-NC** in both cases. It is clear that this method outperforms the other two methods because of its ability to take into account the current state of the robot distribution, and to adapt the transition rates as a function of this.

Finally, we compute the error obtained by our method **Fixed-NC** by comparing the analytical steady-state distribution of traits (obtained by taking the eigenvectors that correspond to the zero-eigenvalues of each rate matrix $\mathbf{K}^{(s)}$ and multiplying them by $\mathbf{Q}$) with the desired trait distribution $\bar{\mathbf{Y}}$. The median, 90th percentile and maximum error from the steady-state to the desired trait distribution are 0.108%, 0.572% and 0.812%, respectively. These results demonstrate that, despite the fact that our method is not explicitly optimizing for the steady-state, it reaches a steady-state error smaller than system noise (at steady-state).

5. CONCLUSION

We present a method that distributes a swarm of heterogeneous robots among a set of tasks with the goal of satisfying a desired distribution of robot capabilities among those tasks. 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 as a function of a desired final trait configuration. To find the optimal transition rates, we pose an optimization problem, and develop a solution based on an analytical gradient that is computationally efficient and capable of producing fast convergence times. Building on this result, we propose a variant real-time optimization method that enables an online adaptation of transition rates as a function of the state of the current robot distribution. We validate our approach on random graph instantiations, and show that our baseline method outperforms a classical alternative approach. Also, we show how, when using the variant adaptive optimization, a significant gain in convergence speed is made. We believe that this method is well-suited to applications that control large-scale teams of robots that need to converge quickly to desired configurations as a function of their capabilities, and that need to adapt to changes in real-time. Future work will include a more in-depth study of the implications of diversity in swarm-robotic systems, as well as an implementation of the proposed framework on real robots.

ACKNOWLEDGEMENTS

We gratefully acknowledge the support of ONR grants N00014-15-1-2115 and N00014-14-1-0510, ARL grant W911NF-08-2-0004, NSF grant IIS-1426840, and TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA.

References

[1] T. Balch, L. E. Parker. Special issue on Heterogeneous Multi-Robot Systems. *Autonomous Robots* **8**:207–383, 2000.

[2] E. G. Jones, B. Browning, M. B. Dias, et al. Dynamically Formed Heterogeneous Robot Teams Performing Tightly Coordinated Tasks. *International Conference on Robotics and Automation (ICRA)* pp. 1–8, 2006. DOI:10.1109/ROBOT.2006.1641771.

[3] B. Charrow. *Information-theoretic active perception for multi-robot teams.* Ph.D. thesis, University of Pennsylvania, 2015.

[4] A. M. Hsieh, A. Cowley, F. J. Keller, et al. Adaptive teams of autonomous aerial and ground robots for situational awareness. *Journal of Field Robotics* **24**:991–1014, 2007. DOI:10.1002/rob.20222.

[5] P. Tokekar, J. Vander Hook, D. Mulla, V. Isler. Sensor Planning for a Symbiotic UAV and UGV system for Precision Agriculture. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* 2013. DOI:10.1109/IROS.2013.6697126.

[6] B. P. Gerkey, M. J. Mataric. A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. *Interntional Journal of Robotics Research* **23**(9):939–954, 2004. DOI:10.1177/0278364904045564.

[7] B. Korte, J. Vygen. Combinatorial Optimization: Theory and Algorithms. Springer-Verlag, Berlin., 2000.

[8] S. Berman, Á. Halasz, M. A. Hsieh, V. Kumar. Optimized Stochastic Policies for Task Allocation in Swarms of Robots. *IEEE Transactions on Robotics* **25**:927–937, 2009. DOI:10.1109/TRO.2009.2024997.

[9] Á. Halasz, M. A. Hsieh, S. Berman, V. Kumar. Dynamic Redistribution of a Swarm of Robots Among Multiple Sites. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* 2007. DOI:10.1109/IROS.2007.4399528.

[10] M. A. Hsieh, Á. Halasz, S. Berman, V. Kumar. Biologically inspired redistribution of a swarm of robots among multiple sites. *Swarm Intelligence* **2**(2-4):121–141, 2008. DOI:10.1007/s11721-008-0019-z.

[11] M. J. B. Krieger, J. B. Billeter, L. Keller. Ant-like task allocation and recruitment in cooperative robots. *Nature* **406**:992–995, 2000.

[12] T. H. Labella, M. Dorigo, J.-L. Deneubourg. Division of labor in a group of robots inspired by ants' foraging behavior. *ACM Transactions on Autonomous Adaptive Systems* **1**:4–25, 2006.

[13] L. Matthey, S. Berman, V. Kumar. Stochastic strategies for a swarm robotic assembly system. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1953–1958. IEEE, 2009. DOI:10.1109/ROBOT.2009.5152457.

[14] D. Tilman. Functional Diversity. *Encyclopedia of Biodiversity* **3**:109–120, 2001.

[15] O. L. Petchey, K. J. Gaston. Functional diversity: back to basics and looking forward. *Ecology Letters* **9**(6):741–758, 2006. DOI:10.1111/j.1461-0248.2006.00924.x.

[16] J. D. Kalbfleisch, J. F. Lawless. The Analysis of Panel Data Under a Markov Assumption. *Journal of American Statistical Association* **80**:863–871, 1985.

[17] D. J. Wales, J. P. K. Doye. Global optimization by basin-hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms. *arXivorg* pp. 1–8, 1998.

[18] A. Mordecai. Nonlinear programming: analysis and methods. Courier Corporation, 2003.

[19] J. Demmel, I. Dumitriu, O. Holtz. Fast Linear Algebra is Stable. *arXivorg* pp. 1–26, 2007.

[20] V. Y. Pan, Z. Q. Chen. The Complexity of the Matrix Eigenproblem. *ACM Symposium on Theory of Computing* pp. 507–516, 1999.

[21] M. Milam, R. Franz, J. Hauser, M. Murray. Receding horizon control of vectored thrust flight experiment. *IEE Proceedings on Control Theory and Applications* **152**:340–348, 2015.

[22] A. Martinoli. Swarm Intelligence in Autonomous Collective Robotics: From Tools to the Analysis and Synthesis of Distributed Collective Strategies. Ph.D. thesis, Ecole Polytechnique Fédérale de Lausanne (EPFL), 1999.

[23] D. J. Watts, S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature* **393**:440–442, 1998.