A Comparison of Deterministic and Stochastic Approaches for Allocating Spatially Dependent Tasks in Micro-Aerial Vehicle Collectives

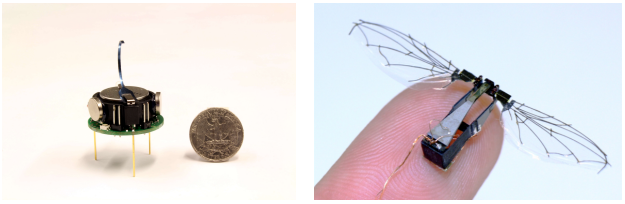
Karthik Dantu, Spring Berman, Bryan Kate, and Radhika Nagpal

***Abstract*— We compare our previously developed deterministic [7] and stochastic [3], [4] strategies for allocating tasks in robotic swarms[1] consisting of very large populations of highly resource-constrained robots. We study our two task allocation approaches in a simulated scenario in which a collective of insect-inspired micro-aerial vehicles (MAVs) must produce a specified spatial distribution of pollination activity over a crop field. We investigate the approaches' requirements, advantages, and disadvantages under realistic conditions of error in robot localization, navigation, and sensing in simulation. Our results show that the deterministic approach, which requires region-based robot navigation, yields higher task progress in all cases. For robots without such navigation capabilities, the stochastic approach is a feasible alternative, and its resulting task progress is less sensitive to error in localization, error in navigation, and a combination of high error in localization, navigation, and sensing.**

I. Introduction

Miniaturization of computing, sensing, actuation, and control technologies is enabling the development of robotic swarms[1] in which each robot is small and relatively inexpensive, making large-scale swarms affordable. Examples of platforms for such multi-robot systems include the Kilobot [28] and the RoboBee [25] (Fig. 1). At the forefront of multi-robot systems is an effort to construct *insect-scale* flapping-wing micro-aerial vehicles (MAVs) [34] like the prototype in Fig. 1(b) [25]. Recent advances in airframe construction [33], flight dynamics and control [30], and sensor design [15] are quickly driving insect-scale MAVs closer to mass production. Applications of MAV swarms include micro-manipulation tasks such as crop pollination, search-and-rescue, disaster response, exploration of hazardous environments, and surveillance.

The problem of controlling MAV swarms to accomplish a collective goal presents unique challenges due to the extreme resource constraints of the platforms [13]. MAVs have very limited sensing and computing capabilities, and they may not have the resources for inter-robot communication. In addition, operations over large areas will require repeated recharging of the MAVs since their flight times are expected to be on the order of minutes, even with projected advances in battery technology. Besides accommodating these limitations, control strategies for MAV swarms must exhibit adaptability to environmental conditions and robustness to robot failures and errors in sensing and actuation.



(a) Kilobot (b) RoboBee

Fig. 1. Examples of miniature robot platforms[1]

In this work, we address the problem of allocating a swarm of MAVs to tasks that are associated with different spatial regions. We compare the utility of a deterministic task allocation approach using the resource management system *Karma* [7] and a stochastic approach *OptRAD* [4], which is based on the optimization of a spatiotemporal model of the swarm population dynamics. These two approaches were developed for swarms of highly resource-constrained robots that must adapt to disturbances and failures. The approaches incorporate a supervisory agent that optimizes the task allocation strategy using feedback from the robots on their individual task progress. Despite the presence of a centralized component, the two approaches are scalable with the number of robots since the robot actions depend on locally sensed information or a broadcast signal, and the supervisor does not require knowledge of this individual activity [22]. We evaluate the effectiveness of the two approaches for varying degrees of error in robot localization, navigation, and sensing in a simulated commercial crop pollination scenario.

II. Related Work

Task allocation in multi-robot systems has received considerable attention in the literature. We define a *task* as an activity, here designated as pollination, that a robot performs in a particular region of the environment. A taxonomy for classifying multi-robot task allocation problems is proposed in [9], [16]. Our application can be classified as an instance of the single-task robot, multi-robot task (ST-MR) problem, in which robots can execute at most one task at a time and tasks require multiple robots, with either instantaneous

K. Dantu, B. Kate, and R. Nagpal are with the School of Engineering and Applied Sciences, Harvard University, 33 Oxford Street, Cambridge, MA 02138 (email: {kar,bkate,rad}@eecs.harvard.edu)

S. Berman is with the School for Engineering of Matter, Transport and Energy, Mechanical and Aerospace Engineering, Arizona State University, 551 E. Tyler Mall, Tempe, AZ 85281 (email: spring.berman@asu.edu)

[1] Note that in this work, we use the term *swarm* to refer to a large collective of interchangeable robots that react autonomously to local information and work toward a common goal. Our swarm is not distributed, and it makes use of a central authority for task allocation decisions.

[1] Images courtesy of 1(a) Mike Rubenstein; 1(b) Ben Finio

assignment (IA) of robots to tasks or a time-extended assignment (TA) using planning for future demand.

Existing approaches to the ST-MR problem that allocate robots to tasks in a *deterministic* manner require extensive communication between robots that is not possible in large MAV swarms. The problem of assigning teams to tasks is known as *coalition formation* when applied to software agents. Approaches to this problem, which is NP-hard, rely on extensive agent cooperation that is not easily implemented in robotic systems [32]. The algorithm in [29] was adapted to multi-robot systems in [31], but robots must compute all possible coalitions and agree on the best ones, and coalition sizes are limited to small teams. The ST-MR problem has been addressed with market-based techniques, which may be centralized or decentralized [8] and apply to IA problems [5], [19] or TA problems [35]. However, the computation and communication requirements of market-based algorithms often scale poorly with increasing numbers of robots and tasks.

Our deterministic allocation approach, Karma, eliminates the need for inter-robot communication by using a central authority to allocate individual robots to tasks in a way that scales with the population size. This approach can be classified as IA since robots are each assigned a single task after the central authority receives feedback on their progress.

Stochastic task allocation approaches for robotic swarms that apply to the ST-MR problem have also been developed. In these approaches, tasks are executed at random times by unidentified robots, and an allocation emerges from the collective activity of the swarm. Threshold-based algorithms [1], [17], [18], inspired by division of labor mechanisms in social insects, use a decentralized paradigm in which robots decide to perform a task if its stimulus exceeds their activation threshold. Recent works address the optimization of the stochastic robot task-switching rates using non-spatial macroscopic models that describe the time evolution of the robot population in each state [2], [6], [21], [24]. In [23], optimal control is applied to a spatial population model for the specific problem of maximizing swarm presence in a desired location.

Our stochastic task allocation approach, OptRAD, both describes the *spatial task distribution* of a swarm with an advection-diffusion-reaction (ADR) PDE population model and employs *optimization* of the model parameters to derive the robot control policies for a target global objective. Our modeling methodology is similar to that in [11], [27], which develop spatial models of robotic swarms that are based on the Fokker-Planck PDE; these works do not address the problem of controller optimization. The OptRAD approach can be classified as TA since each set of policies, which are optimized using feedback from the robots, causes the robots to perform activities in a random sequence of regions, *i.e.*, execute a sequence of tasks.

III. Problem Statement

A. Application Objective

We apply our task allocation approaches to a scenario in which an MAV swarm must achieve a target number of repetitions of a task in each of a set of regions of an environment with a known layout. Specifically, we simulate a commercial pollination scenario in which the task to be repeated is the transfer of pollen from an MAV to a flower. The environment, illustrated in Fig. 2, is modeled after a section of a rabbiteye blueberry orchard [26] that consists of R rows of plants. The allocation strategies should be able to incorporate feedback from the MAVs in order to fulfill the objective in the presence of unknown environmental disturbances such as wind.

B. Swarm Control Architecture

We assume the existence of a central authority called the *hive* (see Fig. 2) that has substantial computing and storage resources and the ability to recharge the MAVs, which have the capabilities outlined in [4]. Each MAV is considered to have the ability to localize itself within the *region* of the environment that it currently occupies (see Fig. 2), where the set of regions forms a coarse discretization of the field. In deterministic task allocation, a scheduler running at the hive makes task allocation decisions for the individual robots. For stochastic task allocation, the hive optimizes the parameters that govern robot motion and stochastic decision-making. In both approaches, the robots undertake brief flights called *sorties* that originate from the hive and last a few minutes. During a sortie, a robot records the number of flower visits that it performs in each region where it lands on flowers. It then returns to the hive to recharge when its batteries run low, uploads the number of flower visits it has executed in each region during the sortie, and receives its next task (deterministic allocation) or parameters for its next sortie (stochastic allocation).

C. Evaluation Criteria

Our goal is to compare the pollination progress achieved using the two task allocation approaches for both ideal conditions and more realistic situations in which the robots have inaccurate navigation, localization, and sensing. Due to their extreme resource constraints, the motion of MAVs is not very accurate. Typically, they navigate using dead reckoning along with detecting environmental features. However, over time this estimate drifts, resulting in actuation error that leads the MAV away from the intended destination. We refer to this effect as *navigation error*. In addition, errors may be introduced in the mechanism by which the MAVs localize themselves in the regions (for example, using beacons or visual landmarks). We term this *localization error*. Finally, the simple MAV sensors, such as a template matching sensor [15], can be subject to *sensing errors* in identifying features of interest such as flowers.

IV. Deterministic Task Allocation

A. Overview

Previously, we built Karma [7], a resource management system for programming and coordinating MAV swarms. The primary objective of Karma is to disentangle what the user wants to accomplish from problems relating to coordination

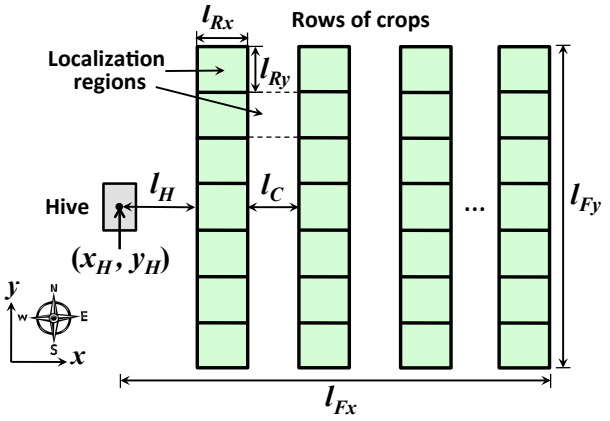


Fig. 2. A section of a crop field with a hive of MAVs. The hive has a map of the field, and the MAVs can localize within coarse regions of the field.

of the swarm. It provides a flexible programming interface to the user that allows for easy specification of the desired tasks, and it allows the user to express varied applications in a uniform manner. Karma assumes that the MAVs have the ability to navigate to a specific region in order to execute the task associated with that region. Details of this approach and example applications such as plume detection, target tracking, and crop monitoring are described in detail in [7].

In our commercial pollination scenario, the application programmer provides information on the size of the field, a map of the plants divided into coarse-grained regions (see Fig. 2), and a target number of flower visits per region that will ensure an adequate crop yield from the resulting pollination. The hive then assigns each MAV to a region where pollination activity remains to be completed. The MAVs fly to their assigned regions and repeatedly execute the following behaviors to perform the pollination task: flying random walks in the region, taking periodic sensor measurements to detect flowers, and landing on a flower to deposit and collect pollen. The hive uses the cumulative flower visit counts recorded during by the MAVs during their sorties to determine the next set of task allocations.

B. Task Allocation Mechanism

Karma optimizes the task allocation strategy for two main objectives: (1) shortest time to application completion, and (2) fairness across all active tasks, *i.e.*, achieving proportional progress on all currently active tasks. For every active task:

- Karma evaluates the progress of task i in region r using a progress function p_i^r. The increment in progress of task i in region r between a past time t and the current time t' per MAV sortie is defined as

$$d_i^r(t,t') = (p_i^r(t') - p_i^r(t))/k,$$

  where k is the number of MAVs that have contributed to the progress of task i in region r between times t and t'.
- Karma estimates the amount of work remaining to complete task i in region r at time t' as

$$W_i^r(t') = \frac{(1 - p_i^r(t'))}{E[d_i^r(t,t')]},$$

  where $E[d_i^r(t,t')]$ is computed from a weighted average of historical $d_i^r(t,t')$ values.

Karma allocates the n MAVs available at time t' to a queue of tasks sorted according to the ratio of $W_i^r(t')$ to the total work to be done across all tasks, $W(t') = \Sigma_i \Sigma_r W_i^r(t')$. This results in an allocation of $n \cdot W_i^r(t')/W(t')$ MAVs to each task i, which is proportional to the amount of work remaining to complete the task.

V. Stochastic Task Allocation

A. Overview

We have previously presented a scalable approach to optimizing robot control policies for a target collective behavior in robotic swarms with arbitrary spatial distributions [4]. This approach, which we refer to here as OptRAD, abstracts the swarm to a macroscopic continuous model. OptRAD does not require that the robots have the ability to navigate to specified regions. It does require *a priori* information on the locations of robots and environmental features, and it applies to swarms that follow a particular type of motion model and stochastic decision-making policies. While Karma can accommodate more general robot controllers, this last requirement facilitates OptRAD's methodology of abstraction and top-down control synthesis and ensures probabilistic guarantees on the system performance.

The user inputs the same information to OptRAD as in Karma, and in addition provides the macroscopic model, or the robot motion specification and behavioral transition parameters from which this model may be constructed, and the optimization variables, objective function, and constraints. The hive optimizes the parameters of the macroscopic model for the desired objective and transmits them to the robots before each sortie. These parameters map to deterministic and random components of the MAVs' velocities and their stochastic policies of switching behaviors, which in the pollination scenario are flying and hovering at a flower. The robots pass through random regions, *i.e.,* randomly perform the *tasks* of pollinating the regions, while executing these behaviors. Upon their return, the hive incorporates their flower visit counts into the macroscopic model and optimizes the parameters for their next sortie.

In [4], we developed an ordinary differential equation macroscopic model over a discretization of the domain for a pollination scenario, and in earlier work [3], we validated an ADR PDE macroscopic model for a similar scenario. Here we use the latter type of model in our optimization method and numerically solve it using a technique that can compute more accurate solutions than the one in [4].

B. Task Allocation Mechanism

1) Robot Control Policies: The tunable robot parameters for each sortie $s \in \{1,...,S\}$ are given in the following vector,

which is optimized and input to the robots at the hive:

$$\mathbf{p}^s = [\ v \quad D \quad k_{hov,1} \quad \ldots \quad k_{hov,R} \quad \phi \quad \rho\ ]^T. \tag{1}$$

These parameters define the robot motion and decision-making control policies as follows. The robots switch stochastically between the behaviors *fly* and *hover at a flower*. We assume that the flowers are distributed densely enough such that a robot can always detect at least one flower in its sensing range when it flies over plants. While a robot is flying over row $j \in \{1,...,R\}$, it decides with probability $k_{hov,j}\Delta t$ per timestep Δt to pause at a flower in its sensing range and hover for pollination. The robot resumes flying with probability $k_{fly}\Delta t$ per time step. At every timestep during flight, the commanded velocity of each robot i is the sum of a deterministic component $\mathbf{v}^d$ and a random component $\mathbf{v}^r$. The deterministic component directs the robot in a compass direction ϕ at a constant speed v until it encounters the edge of a plant row $\rho \in \{1,...,R\}$, at which point it flies eastward at speed v. The random component is actively added as a mechanism for the swarm to achieve thorough coverage of the field. This random motion is modeled as a diffusion with an associated diffusion coefficient D. We define $\mathbf{v}^r = (2D/\Delta t)^{1/2}\mathbf{Z}$, where the components of $\mathbf{Z} \in \mathbb{R}^2$ are independent normal random variables with zero mean and unit variance.

2) Macroscopic Model: From the robot velocity definition, the displacement of a robot during flight during each time step Δt is described by the standard-form Langevin equation [10]. In addition, the robot decisions to visit and leave flowers can be modeled as unimolecular chemical reactions, as detailed in [4]. We can therefore describe the expected spatiotemporal distribution of the swarm with a set of ADR PDE's [3]. The following equations govern the time evolution of the population fields $x = x(\mathbf{q},t)$ of the flying robots (species B_{fly}), the hovering robots (species B_{hov}), and the flower visit instances (species V) at every point $\mathbf{q} \in \mathbb{R}^2$ during a single sortie:

$$\frac{\partial x_{B_{fly}}}{\partial t} = -\nabla \cdot (\mathbf{v}^d(\mathbf{q}) x_{B_{fly}}) + D\nabla^2 x_{B_{fly}} + g_{B_{fly}}(\mathbf{q}),$$
$$\frac{\partial x_{B_{hov}}}{\partial t} = g_{B_{hov}}(\mathbf{q}), \qquad \frac{\partial x_V}{\partial t} = g_V(\mathbf{q}), \tag{2}$$

where, for all points $\mathbf{q}$ in crop row $j \in \{1,...,R\}$,

$$\begin{aligned} g_{B_{fly}}(\mathbf{q}) &= -k_{hov,j} x_{B_{fly}} + k_{fly} x_{B_{hov}}, \\ g_{B_{hov}}(\mathbf{q}) &= k_{hov,j} x_{B_{fly}} - k_{fly} x_{B_{hov}}, \\ g_V(\mathbf{q}) &= k_{hov,j} x_{R_{fly}}, \end{aligned} \tag{3}$$

and $g_k(\mathbf{q}) = 0$ for $\mathbf{q}$ outside of the crop rows.

3) Optimization of Robot Control Policies: The macroscopic model (2), (3) is used to compute the parameter vector (1) for each sortie s that maximizes a metric $f(\mathbf{p}^s)$ of the degree of pollination over the sortie. The solution of the model for sortie s is a function of these parameters, *i.e.*, $x_k = x_k(\mathbf{p}^s, \mathbf{q}, t)$, $k \in \{B_{fly}, B_{hov}, V\}$. We compute this solution numerically as outlined in the Appendix. To specify $f(\mathbf{p}^s)$, we define $\mathscr{F}$ as the set of indices of regions that contain flowers, N_r^V as the target number of flower visits per region $r \in \mathscr{F}$, and C_r as the collection of grid cells c in the numerical solution that are contained in region r. As described in the Appendix, $v_c^{n_f}(\mathbf{p}^s)$ is the numerical solution of the field $x_V(\mathbf{q},t)$ in cell c for sortie s, initialized with the total number of flower visits in cell c recorded by the robots over sorties $1,...,s-1$. The metric for sortie s is defined as the fraction of desired pollination activity over the field that has been achieved during sorties $1,...,s$:

$$f(\mathbf{p}^s) = \frac{1}{M_F} \sum_{r \in \mathscr{F}} \min\left(\frac{\sum_{c \in C_r} v_c^{n_f}(\mathbf{p}^s)}{N_r^V}, 1 \right). \tag{4}$$

The optimization problem is posed as **Problem P** below:

[P] For each sortie $s \in \{1,...,S\}$, minimize $f(\mathbf{p}^s)$ subject to the macroscopic model (2), (3) with boundary condition and initial condition as described in the Appendix and subject to the constraints $p_{i,min} \le \mathbf{p}_i^s \le p_{i,max}$, $i = 1,...,4$; $\mathbf{p}_5^s \in \{1,...,R\}$.

We implement this problem using a Metropolis optimization method similar to the one used with the *closed-loop* control strategy in [4]. The flower visits counts recorded by the robots during a sortie s are used to set $v_c^0(\mathbf{p}^{s+1})\ \forall c$, as specified in the initial condition described in the Appendix, for the optimization of $\mathbf{p}^{s+1}$.

VI. Simulation Setup

As described in Section III, we use commercial crop pollination as a sample application to compare our task allocation approaches. To simulate the robots, we used Simbeeotic [14], a framework built to simulate MAV swarms at scale. Simbeeotic is a discrete event simulator written in Java that allows the user to program custom MAV behavior, different kinds of sensors, virtual environments, and realistic conditions such as wind. It is built on top of JBullet, a 6-DoF physics engine that simulates interactions by modeling them as forces, including gravity. In addition, Simbeeotic provides tools such as plotting and 3D visualization that allow the user to investigate the swarm behavior.

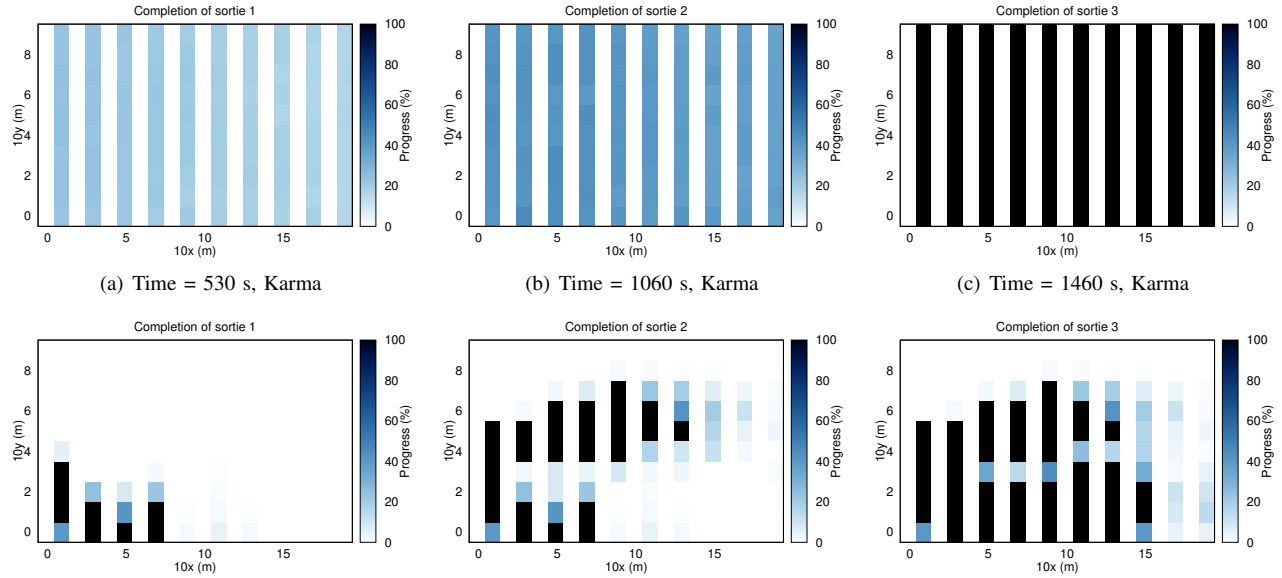
Karma is implemented in Simbeeotic as described in Section IV. In order to accurately compare the two task allocation approaches under the same simulated conditions, we implemented them as two separate schedulers in Karma. Each simulation is initiated by submitting an application to Karma with a map of the environment describing the dimensions of the field, dimensions of the crop rows, and a target number of MAV visits for each region. Karma then executes the application using either the Karma scheduler to allocate MAVs to tasks or the OptRAD scheduler to optimize the set of parameters that governs this assignment.

The simulated crop field has the layout in Fig. 2 with ten rows of crops and the following dimensions: $l_{Fx} = 200$ m, $l_{Fy} = 100$ m, $l_{Rx} = l_{Ry} = 10$ m, $l_H = 10$ m, and $l_C = 10$ m. The hive is placed at (x_H, y_H) = (0 m, 50 m). In OptRAD, the parameters were optimized over the following ranges: $v \in [1\ 10]$ m/s, $D \in [3\ 5]$ m^2/s, $k_{hov,j} \in [0.05\ 1.25]$ s^{-1} for all rows j, $\phi \in [-\pi/2\ \pi/2]$ rad, and $\rho \in \{1,...,10\}$. We run each simulation for 1 hour of simulated time.

TABLE I

PROPERTIES OF KARMA AND OPTRAD

Property	**Karma**	**OptRAD**
Task allocation mechanism	Deterministic	Stochastic
Specificity to application	Allocation mechanism agnostic to application	Modeling and optimization depend on application
Requirements on *a priori* information about the environment	No requirements	Initial spatial distribution of robots and objects with which they interact (ex. flowers)
MAV navigation requirements	MAVs require ability to navigate to specified regions	No requirements
MAV actuation requirements	No requirements	Specified velocity field, degree of random motion, and probability rates of switching behaviors



(a) Time = 530 s, Karma (b) Time = 1060 s, Karma (c) Time = 1460 s, Karma

(d) Time = 450 s, OptRAD (e) Time = 970 s, OptRAD (f) Time = 1520 s, OptRAD

Fig. 3. Snapshots of task progress in Karma and OptRAD over three sorties for $n = 500$ MAVs, no error.

Each simulated MAV can fly at a maximum speed of 10 m/s. The energy consumption of an MAV is modeled after the energy consumption of our protoype MAVs [7], [14]. An MAV uses approximately 500 mW to fly at full speed and about 250 mW to hover, and its energy consumption is assumed to scale linearly with its flight speed. The MAV recharge time is simulated to be 250 s. Each simulated MAV is provided with a sensor that can detect flowers when the MAV is hovering over them, and the act of detecting a flower and landing on it is programmed to take 5 s on average. Each MAV is also given an associated "location" sensor that localizes the MAV to the region it currently occupies.

VII. COMPARISON OF TASK ALLOCATION APPROACHES

The objective in our simulated crop pollination scenario is to uniformly pollinate ten rows of crops with $N_r^V = 125$ target flower visits per region r. It is also possible to specify nonuniform pollination with a different required number of visits in each region. Table I summarizes the salient differences between Karma and OptRAD. Both have different requirements on the MAV capabilities; hence, only one of the approaches may be suitable for any particular application.

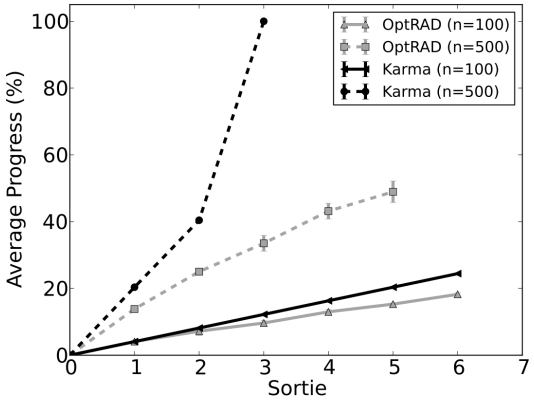


Fig. 4. Cumulative pollination progress for each sortie using Karma and OptRAD to allocate $n = 100$ and $n = 500$ MAVs with no error.

A. Ideal localization, navigation, and sensing

Fig. 3 shows snapshots of the spatial distribution of task progress over three sorties when both Karma and OptRAD are used to allocate $n = 500$ MAVs with no error in navigation, localization, or sensing. Task progress is measured in each region as the percentage of target flower visits achieved. The figure illustrates that the allocation of robots to

specified regions in Karma produces a uniform distribution of pollination activity over the field after each sortie. In contrast, OptRAD yields a coverage pattern that reflects the directed movement and diffusion of the swarm under different sets of optimized parameters, with each subsequent sortie generating flower visits in previously unpollinated regions. OptRAD initially produces more sufficiently pollinated regions than Karma, but it does not achieve full coverage in all regions as Karma does by the end of the simulation. This is largely due to the decreased density of robots on the eastern side of the field as a result of their diffusion over time.

Fig. 4 compares the cumulative task progress per sortie achieved using both Karma and OptRAD with $n = 100$ and $n = 500$ MAVs. Progress is measured as the ratio of the total number of flower visits completed to the total target number of visits. Each plot in Fig. 4 represents an average over five trials, and the error bars indicate standard deviations. For $n = 100$, both approaches yield approximately 20% progress by the end of the simulation. A population of $n = 500$ is sufficient for Karma to achieve 100% progress, while OptRAD produces 49% progress. The quicker progress of Karma toward the goal is a result of its ability to target specific underpollinated regions with MAVs, as opposed to the OptRAD technique of relying on probabilistic region occupation arising from the MAV motion. Although OptRAD makes less progress than Karma, it shows a smaller final decrease in progress compared to Karma when n is reduced from 500 to 100.

B. Error in localization

To model error in localization, we add a normal random variable with zero mean and standard deviation σ_l (in meters) to the exact position recorded by the simulated location sensor on each MAV. Denoting the actual position of a robot i by $\mathbf{q}_i = [q_{i,x}\ q_{i,y}]^T$, the position returned by the noisy location sensor is $\mathbf{q}_i + \sigma_l \mathbf{Z}$. In both Karma and OptRAD, localization error can cause an MAV to incorrectly attribute its flower visit to a region other than the one it actually occupies. Since each region that overlaps a crop row spans the entire width of the row (see Fig. 2), a region that contains crops ("crop region") is adjacent to at least two regions that do not contain crops ("empty" regions). Thus, MAVs that are flying over a crop region near its boundary are likely to attribute their flower visits to an adjacent empty region, resulting in reduced visit counts per crop region. Fig. 5 shows the effect of this error on cumulative task progress, averaged over five trials, with $\sigma_l = 1.0$ m and $n = 500$. Progress in Karma at the end of sortie 3 to drops from 100% to 65%, requiring another sortie for completion. Progress in OptRAD at the end of sortie 5 shows a smaller decrease, from 49% to 32%. This greater robustness to localization error is due to the high probability of correctly-localized flower visits from the cumulative activity of the dense swarm passing over different sets of crop regions during each sortie.

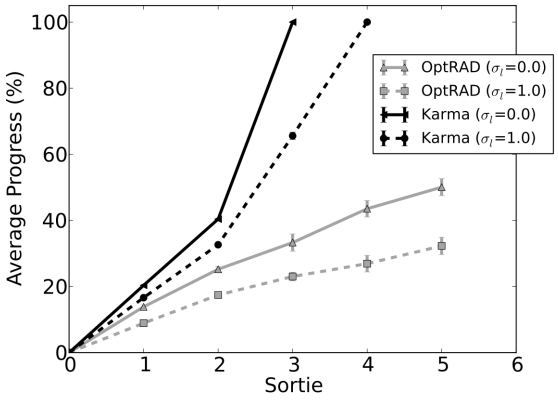


Fig. 5. Effect of localization error on task progress with $n = 500$.

C. Error in navigation

We model error in navigation by adding noise to the velocity commanded to each MAV. Defining the velocity vector of the i^{th} MAV as $\mathbf{v}_i = [v_{i,x}\ v_{i,y}]^T$, we generate a corresponding noisy MAV velocity $\mathbf{v}_i^n$ at each timestep by adding a normal random variable with zero mean and standard deviation σ_n to the normalized velocity vector:

$$\mathbf{v}_i^n = ||\mathbf{v}_i|| \left(\frac{\mathbf{v}_i}{||\mathbf{v}_i||} + \sigma_n \mathbf{Z} \right). \tag{5}$$

In Karma, the navigation error causes each MAV to deviate from the direct path to its assigned region, delaying its entry into the region and thus reducing the number of flower visits that it can accomplish during a sortie. In OptRAD, the error adds a random component to the motion of the robots, effectively increasing their diffusion coefficient D. This added diffusion is not incorporated into the macroscopic model, and so the model parameters are optimized for a higher density of robots and hence an overestimated density of flower visits in the regions that they pass through. Fig. 6 shows that for $\sigma_n = 0.2$ and $n = 500$, these effects cause a smaller reduction in the average progress over 5 trials for OptRAD in comparison to Karma. In Karma, progress drops from 100% to 77% at the end of sortie 3, while in OptRAD, progress drops from 49% to 34% at the end of sortie 5. Hence, OptRAD is slightly more robust to navigation error than Karma.

D. Error in sensing

We simulate error in the MAV flower sensors by assigning a probability p_e that a hovering MAV does not detect a flower within its sensing range, and hence does not pollinate the flower. As Fig. 7 shows, for the case of $p_e = 0.05$ and $n = 500$ with results averaged over 5 trials, this reduction in flower visits in the regions causes a similar degradation in performance in Karma and OptRAD. In Karma, progress decreases from 100% to 92% at the end of sortie 3, while in OptRAD, progress decreases from 49% to 38% at the end of sortie 5.

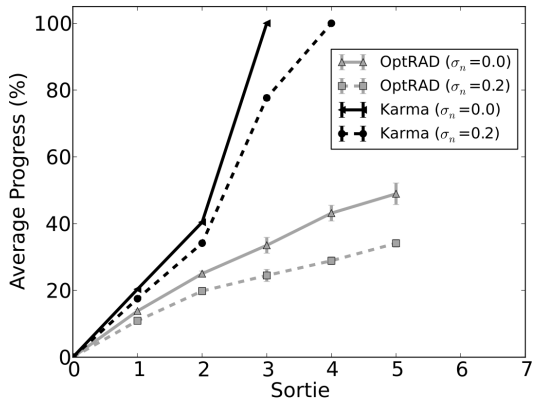


Fig. 6. Effect of navigation error on task progress with $n = 500$.

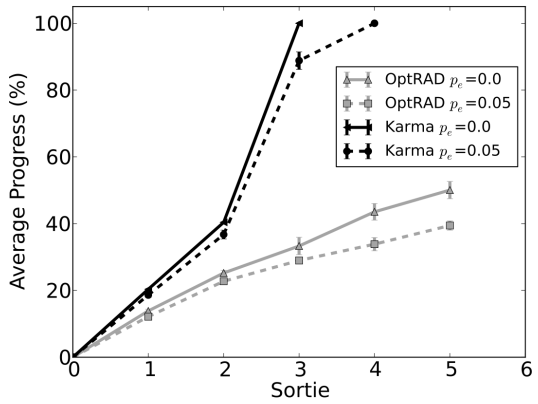


Fig. 7. Effect of sensing error on task progress with $n = 500$.

E. Error in localization, navigation, and sensing

We simulate all three types of error simultaneously for two sets of the parameters σ_l, σ_n, and p_e, running five trials per parameter set. As Fig. 8 shows, Karma achieves greater progress than OptRAD in both cases, while progress in OptRAD is more robust to the combined high errors. In Karma, progress achieved after 3 sorties drops from 100% for low error to 60% for high error, while in OptRAD, progress achieved after 5 sorties shows a much smaller reduction, from 37% to 32%. This demonstrates the robustness of OptRAD despite it not performing as well in ideal conditions. The stochastic nature of OptRAD smoothens the effect of real-world error in its performance.

SPRING: Put plots in the graph for the case n=100 with all 3 types of error, *High err* parameters? OptRAD may do better in this case.

VIII. Conclusion

Rapid innovations in computing, sensing, and actuation are quickly making large-scale swarms of MAVs a reality. One of the prominent challenges in realizing their potential is understanding how to best use such swarms at scale. In this work, we have compared the effectiveness of a deterministic task allocation approach (Karma) and a stochastic approach (OptRAD) for robotic swarms in the representative application of crop pollination. Karma explicitly assigns MAVs

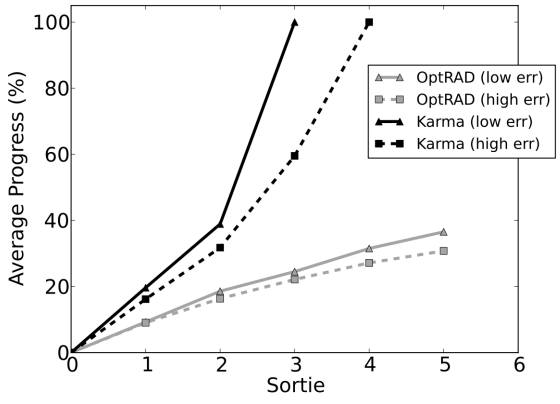


Fig. 8. Effect of error in localization, sensing, and navigation with $n = 500$. *Low err* has error parameters $\sigma_l = 0.2$ m, $\sigma_n = 0.05$, $p_e = 0.05$; *high err* has parameters $\sigma_l = 1.0$ m, $\sigma_n = 0.2$, $p_e = 0.15$.

to specific regions, while OptRAD optimizes the parameters that govern the MAV motion and stochastic decisions.

Our results demonstrate that when it is possible to implement region-based navigation for the robots, then Karma is the more effective task allocation approach. The Karma strategy is more capable than OptRAD at targeting regions that are most in need of MAVs, and it facilitates a uniform distribution of activity over the domain. The diffusive motion of MAVs in the OptRAD strategy provides broader coverage than velocity-driven motion alone, but it leads to underserved regions far from the hive. Under ideal conditions of no MAV error, Karma makes more efficient use of a large swarm of 500 MAVs than OptRAD, producing twice the total task progress of OptRAD within fewer sorties. If the robots are too resource-constrained for region-based navigation to be implemented, even navigation with high error, then OptRAD is a feasible strategy for task allocation. A larger swarm size would be required to achieve the same level of task progress as Karma. Due to its higher redundancy in swarm activity over visited regions, OptRAD has the advantage over Karma of significantly greater robustness in progress in cases where there is either localization error alone or a combination of high error in localization, navigation, and sensing.

In the future, we would like to compare our task allocation approaches for other applications, including ones that incorporate inter-MAV communication, dynamic environmental features such as wind, and dynamic features of interest such as mobile targets to be tracked. Another direction of future work is to investigate distributed programming models for swarm task allocation. A distributed approach would reduce information latency and increase the robustness of the allocation mechanism by eliminating the single point of failure (the hive) in our current control architecture.

Appendix: Numerical Solution of the Macroscopic PDE Model in OptRAD

We numerically solve the macroscopic model (2), (3) with an explicit finite-volume method in which the advection term is solved using the Lax-Wendroff method, implemented with a superbee flux limiter to prevent spurious oscillations [20].

We use an Euler method to numerically integrate the model with timestep $\Delta\tau$, operator splitting [12] to sequentially solve the advection, diffusion, and reaction components of the model at each timestep, and dimensional splitting to split the 2D advection component into 1D subproblems.

The robots in our scenario may exit the domain of interest Ω. Hence, the field $x_{B_{fly}}(\mathbf{q},t)$ evolves on an unbounded domain and should satisfy the condition $\lim_{||\mathbf{q}||\to\infty} x_{B_{fly}} = 0$. We approximate this condition by solving for the field over an enlarged domain Ω_e and imposing a Dirichlet boundary condition $x_{B_{fly}}(\mathbf{q},t) = 0$ at its boundary.

We approximate the duration of each sortie as a constant t_s. The macroscopic model is solved on a uniform 2D Cartesian grid over the domain Ω_e at a set of times $\{t_n\}$, $n = 0, 1, ..., n_f$, where $t_0 = 0$, $t_{n_f} = t_s$, and $t_{n+1} - t_n = \Delta\tau$. We label each grid cell as $c \in \{1, ..., C\}$. An approximated average of the population fields $x_{B_{fly}}$, $x_{B_{hov}}$, and x_V over cell c at time t_n during sortie s is denoted by $b^n_{fly,c}(\mathbf{p}^s)$, $b^n_{hov,c}(\mathbf{p}^s)$, and $v^n_c(\mathbf{p}^s)$, respectively. At the beginning of each sortie s, we set $b^0_{fly,c}(\mathbf{p}^s) = \lfloor N/M_H + 0.5 \rfloor$ for all M_H grid cells c in the hive domain and $v^0_c(\mathbf{p}^s) = \sum_{l=1}^{s-1} \upsilon_c^{n_f}(\mathbf{p}^l)$ when $s > 1$; all other field values are initialized to 0. The value $\upsilon_c^{n_f}(\mathbf{p}^l)$ is the number of visits to cell c by time t_{n_f} during sortie l; this number is recorded by the robots and uploaded to the hive upon their return. When a region size is greater than the grid cell size, this value is estimated by dividing the number of visits recorded in the region during the sortie equally among the grid cells that the region contains.

ACKNOWLEDGMENT

The authors gratefully acknowledge support from NSF Award CCF-0926148.

REFERENCES

[1] W. Agassounon and A. Martinoli. Efficiency and robustness of threshold-based distributed allocation algorithms in multi-agent systems. In *Proc. First Int'l. Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1090–1097, 2002.

[2] S. Berman, Á. Halász, M. A. Hsieh, and V. Kumar. Optimized stochastic policies for task allocation in swarms of robots. *IEEE Trans. Robotics*, 25(4):927–937, Aug. 2009.

[3] S. Berman, V. Kumar, and R. Nagpal. Design of control policies for spatially inhomogeneous robot swarms with application to commercial pollination. In *Int'l. Conf. Robotics and Automation (ICRA)*, pages 378–385, 2011.

[4] S. Berman, R. Nagpal, and Á. Halász. Optimization of stochastic strategies for spatially inhomogeneous robot swarms: A case study in commercial pollination. In *Int'l. Conf. Intelligent Robots and Systems (IROS)*, pages 3923–3930, 2011.

[5] H.-L. Choi, L. Brunet, and J. P. How. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 25(4):912–926, August 2009.

[6] N. Correll. Parameter estimation and optimal control of swarm-robotic systems: A case study in distributed task allocation. In *Int'l. Conf. Robotics and Automation (ICRA)*, pages 3302–3307, 2008.

[7] K. Dantu, B. Kate, J. Waterman, P. Bailis, and M. Welsh. Programming micro-aerial vehicle swarms with Karma. In *Proc. 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys'11, pages 121–134, New York, NY, 2011. ACM.

[8] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz. Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257 –1270, July 2006.

[9] B. Gerkey and M. Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *Int'l. J. of Robotics Research*, 23(9):939–954, 2004.

[10] D. T. Gillespie. Information flow, opinion polling and collective intelligence in house-hunting social insects. *Phil. Trans. Roy. Soc. London*, B(357):1567–1584, 2002.

[11] H. Hamann and H. Wörn. A framework of space-time continuous models for algorithm design in swarm robotics. *Swarm Intelligence*, 2(2-4):209–239, 2008.

[12] W. Hundsdorfer and J. G. Verwer. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*, volume 33 of *Springer Series in Computational Mathematics*. Springer, Berlin, 2003.

[13] M. Karpelson, J. P. Whitney, G.-Y. Wei, and R. J. Wood. Energetics of flapping-wing robotic insects: Towards autonomous hovering flight. In *Proc. IEEE/RSJ Int'l. Conf. on Intelligent Robots and Systems (IROS)*, pages 1630–1637, Oct. 2010.

[14] B. Kate, J. Waterman, K. Dantu, and M. Welsh. Simbeeotic: A simulator and testbed for micro-aerial vehicle swarm experiments. In *Proc. 11th ACM/IEEE Conf. on Information Processing in Sensor Networks*, IPSN-SPOTS '12. IEEE/ACM, April 2012.

[15] S. Koppal, I. Gkioulekas, G. Barrows, and T. Zickler. Wide-angle micro sensors for vision on a tight budget. In *Proc. IEEE Conf. on Computer Vision and Pattern Recog. (CVPR)*, pages 361–368, 2011.

[16] G. A. Korsah. *Exploring bounded optimal coordination for heterogeneous teams with cross-schedule dependencies*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2011.

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

[18] T. H. Labella, M. Dorigo, and J.-L. Deneubourg. Division of labor in a group of robots inspired by ants' foraging behavior. *ACM Trans. Auton. Adapt. Syst.*, 1(1):4–25, 2006.

[19] M. G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, and S. Jain. Auction-based multi-robot routing. In *In Robotics: Science and Systems*, pages 343–350, 2005.

[20] R. J. Leveque. *Finite-Volume Methods for Hyperbolic Problems*. Cambridge Univ. Press, 2004.

[21] W. Liu and A. F. T. Winfield. Modeling and optimization of adaptive foraging in swarm robotic systems. *Int'l. J. of Robotics Research*, 29(14):1743–1760, 2010.

[22] N. Michael, J. Fink, S. Loizou, and V. Kumar. Architecture, abstractions, and algorithms for controlling large teams of robots: Experimental testbed and results. *Int'l. Symposium of Robotics Research (ISRR)*, 2007.

[23] D. Milutinovic and P. Lima. Modeling and optimal centralized control of a large-size robotic population. *IEEE Trans. Robotics*, 22(6):1280–1285, 2006.

[24] L. Odhner and H. Asada. Stochastic recruitment control of large ensemble systems with limited feedback. *ASME J. Dyn. Sys. Meas. Control*, 132(4):041008–1–041008–9, 2010.

[25] H. S. of Engineering and A. Sciences. *RoboBees: A Convergence of Body, Brain, and Colony*, 2012. `http://robobees.seas.harvard.edu/`.

[26] A. Powell, W. A. D. Jr., and D. G. Himelrick. Commercial blueberry production guide for Alabama, 2002. Alabama Cooperative Extension System, ANR-904.

[27] A. Prorok, N. Correll, and A. Martinoli. Multi-level spatial modeling for stochastic distributed robotic systems. *Int'l. J. of Robotics Research*, 30(5):574–589, 2011.

[28] M. Rubenstein, C. Ahler, and R. Nagpal. Kilobot: A low cost scalable robot system for collective behaviors. In *Proc. IEEE Int'l. Conf. on Robotics and Automation (ICRA)*, 2012.

[29] O. Shehory and S. Kraus. Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101:165–200, 1998.

[30] P. S. Sreetharan and R. J. Wood. Passive torque regulation in an underactuated flapping wing robotic insect. *Autonomous Robots*, 31(2-3):225–234, Oct. 2011.

[31] L. Vig and J. Adams. Multi-robot coalition formation. *IEEE Trans. on Robotics*, 22(4):637–649, 2006.

[32] L. Vig and J. Adams. Coalition formation: From software agents to robots. *J. Intellig. and Robotic Syst.*, 50(1):85–118, 2007.

[33] R. J. Wood. The first takeoff of a biologically inspired at-scale robotic insect. *IEEE Transactions on Robotics*, 24(2):341 –347, April 2008.

[34] R. J. Wood. Fly, robot, fly. *Spectrum, IEEE*, 45(3):25–29, Mar. 2008.

[35] R. Zlot. *An auction-based approach to complex task allocation for multirobot teams*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 2006.