L. Garattoni and M. Birattari (2016). *Swarm robotics.* In J.G. Webster (Eds.) Wiley Encyclopedia of Electrical and Electronics Engineering. John Wiley & Sons, Hoboken, NJ.

Swarm Robotics

Lorenzo Garattoni and Mauro Birattari

IRIDIA, Université Libre de Bruxelles, Belgium

Abstract

Swarm robotics is an approach to coordinating a highly redundant group of robots. A robot swarm is an autonomous entity that acts in a self-organized way: the complexity of its collective behaviors is the result of the local interactions between the individual robots. A robot swarm neither has a leader nor any other centralized entity that is responsible for its coordination. Self-organization, high redundancy, and the lack of single points of failure promote fault tolerance, scalability and flexibility. These are desired properties for systems deemed to successfully function in the real world. However, these properties also pose a challenging engineering problem: the behavior of the individual robot cannot be conceived individually;it must be conceived by considering the collective behavior that it produces when executed by a large number of robots. Designing the robot-robot and the robot-environment interactions that would result in the desired collective behavior is a difficult endeavor. Research towards the definition of an engineering methodology for designing, analyzing, and maintaining robot swarms is currently ongoing. In this article, we present swarm robotics from an engineering perspective: we describe works that contribute to the advancement of swarm robotics as an engineering field and to its forthcoming uptake in real-world applications.

Keywords—swarm robotics, collective robotics, robotics, self-organization

1 Introduction

Swarm robotics is an approach to robotics in which a mission is entrusted to a large group of robots, a so called *robot swarm*.

A robot swarm operates in an autonomous and self-organized way. A swarm does not rely on any centralized entity for making decisions and for coordinating its activities. In particular, a swarm does not rely on a leader robot or on external infrastructures: the collective behavior of the swarm is the result of the interactions between the individual robots and between robots and environment.

A robot swarm might comprise only robots that are endowed with identical hardware and control software. In this case, the swarm is called a *homogeneous swarm*. Alternatively, a swarm might comprise robots endowed with hardware and/or control software that belong to a number of classes. In this case, the swarm is called a *heterogeneous swarm*. What homogeneous and heterogeneous swarms have in common is that they are highly redundant: there are multiple

robots in the swarm that are able to perform each of the individual actions that are required to accomplish the given mission. In other words, no single robot is indispensable.

A characteristic of robot swarms is locality of interaction: each robot has a limited range of communication and perception. As a consequence, at any moment in time, each robot directly interacts only with the other (relatively few) robots that happen to be in its neighborhood. The main implication of this is that each robot is unaware of the overall size of the swarm and is unaffected by it.

In a typical swarm robotics application, robots operate in parallel on multiple tasks. They switch from task to task according to contingencies. Coherently with what has been stated above, task allocation is autonomous, self-organized, and based only on locally available information.

Autonomy, self-organization, redundancy, locality, and parallel execution are highly appreciated characteristics of a robot swarm as they are commonly deemed to promote fault tolerance, scalability, and flexibility.

Fault tolerance: high redundancy and the lack of a single point of failure (no leader robot, no external infrastructure) promote the realization of a system that is robust to failures of individual robots.

Scalability: locality of interaction promotes the realization of a system in which the addition (or removal) of robots does not qualitatively change the behavior of the system and therefore does not require modifying the behavior of the individual robots—provided that robot density is not dramatically altered.

Flexibility: parallel execution and autonomous task allocation promote the realization of a system that reacts and adapts to contingencies, modifications of the environment, and variations of the working conditions.

Swarm robotics juxtaposes itself to the single-robot approach [1] and to classical multi-robot approaches [2, 3, 4].

In the single-robot approach, a mission to be performed is entrusted to a single, monolithic robot, for example, a humanoid robot. With respect to the classical single robot approach, swarm robotics appears to be more promising in applications in which fault tolerance, scalability, and flexibility are particularly desirable. Moreover, each of the individual robots composing a swarm is mechanically simpler than a single monolithic robot whose capabilities are comparable to the one of the swarm. As a consequence, it should be expected that the cost of hardware design is reduced in the case of swarm robotics. On the other hand, designing the individual robot behavior that, through robot-robot and robot-environment interactions, would produce the desired collective behavior is more complex than designing the behavior of a single, monolithic robot.

In the classical multi-robot approaches, the mission of interest is entrusted to a relatively small team of robots [5, 2], smaller than a typical swarm. Usually, the team behaviors are tailored to the specific team size and thus need to be adjusted as the team size varies. In the classical multi-robot approaches, each team member has a role that is defined at design time. Also the patterns of interaction are defined at design time and are typically more rigid than those that characterize a robot swarm. As a consequence, a classical multi-robot system is not as fault tolerant, scalable, and flexible as a robot swarm. On the other hand, the interaction protocols of a classical multi-robot system are typically simpler to define than those of a robot swarm, as interactions are well

defined and predictable and all relevant information is available at design time.

Beside being a promising engineering approach to the development of complex robotics systems, swarm robotics can be a powerful tool for studying social behaviors in biology as it is attested by a significant body of literature [6]. When swarm robotics is used as a tool to study social behaviors, the robots are programmed to reproduce as faithfully as possible the behavior observed in the biological system under analysis.

This article presents swarm robotics from an engineering perspective. The focus is on methods for designing and analyzing robot swarms with the ultimate goal of adopting them in real-world applications. When swarm robotics is intended as a field of engineering, social behaviors of insects, birds, and mammals are often a valuable source of inspiration for the designer. Nonetheless, as the goal of a designer is the pragmatic one of producing a system that accomplishes a given mission, the source of inspiration is loose and the designer is ready to depart from the biological system should this be needed to meet the requirements. In an engineering perspective, the biological plausibility of the final result is not a value in itself.

This article is an introduction to swarm robotics and covers the most important contributions to date, although it is not meant to be a comprehensive and exhaustive account of the swarm robotics literature. Contributions are classified into design methods, modeling methods, and collective behaviors. This classification is inspired by the one proposed by Brambilla et al. [7]. Additionally, the article describes few particularly notable robot swarms that serve as concrete examples of the recent achievements in the field of swarm robotics. Finally, we cover some promising prospective applications of swarm robotics. Other articles have previously reviewed the swarm robotics literature: The already mentioned Brambilla et al. [7] recently reviewed the swarm robotics literature from an engineering perspective. Şahin [8] was the first to formally define the basic concepts of swarm robotics and to provide a survey of the literature. Bayindir and Şahin [9] presented the literature via five taxonomies: modeling, behavior design, communication, analytical studies and problems. Iocchi, Nardi, and Salerno [4] classified multi-robot systems depending on their degree of awareness, coordination, and decentralization and dedicates a section to applications of multi-robot systems. Finally, Gazi and Fidan [10] surveyed the literature from a control-theory perspective, focusing on the problems of modeling the dynamics of a robot swarm, and presenting approaches for its control and coordination.

The rest of the article is organized as follows: Section 2 presents the most common approaches used to design collective behaviors of robot swarms; Section 3 presents methods for modeling and analyzing the behavior of robot swarms; Section 4 describes a number of collective behaviors that have been realized and discussed in the literature; Section 5 describes six notable robot swarms that have been demonstrated; Section 6 discusses some promising application areas; and Section 7 concludes the article.

2 Design

Designing a robot swarm that is able to accomplish a given mission is a difficult endeavor. Usually, requirements on the mission are expressed at the swarm level, the so called *macroscopic* level, while the designer works at a lower level

by implementing the behavior of the individual robots that compose the swarm, the so called *microscopic* level. The interaction of the individuals gives rise to a collective behavior that should satisfy the swarm-level requirements. In particular, the resulting collective behavior should allow the swarm to accomplish the given mission. To date, no general formal method exists to derive the individual behavior from the swarm-level requirements. The problem of designing robot swarms is tackled either manually or via automatic methods.

2.1 Manual design

In manual design, the designer of the swarm develops, by hand, the behavior of the individual robots that yields the desired collective behavior. In swarm robotics, the behavior of the individual robots is typically reactive—that is, robots act in response to contingencies (possibly influenced by their memory), without planning their future actions nor reasoning on their effects. The software architecture that is most broadly adopted is a particular class of finite state machines, the probabilistic finite state machine [11].

Although most robot swarms are still developed through trial and error, in recent years principled design approaches have been proposed. The following two sections are devoted to the trial-and-error design approach and to some of the most promising principled design approaches.

Trial-and-error design

Designing a robot swarm by trial and error is more of an art than a science. The designer operates in an unstructured way with little scientific basis and technical tools: the designer searches for an individual-level behavior that, through the complex interaction of a large number of robots, would result in the desired collective behavior. The search process is performed via educated guesses that rely solely on the expertise and the ingenuity of the designer. The designer starts by defining a first implementation of the individual robot behavior. The designer then tests the behavior, usually by means of computer-based simulations, and iteratively adjusts it until the resulting collective behavior meets the swarm-level requirements. Often, the designer takes inspiration from biological systems: when the goal is to design a robot swarm whose swarm-level behavior is similar to the one of a biological system (e.g., a swarm of insects, a flock of birds, or a herd of mammals), the designer might find convenient to design the behavior of the individual robot by mimicking the one of the individual member of the biological system.

The relationship between the microscopic and the macroscopic levels poses challenging issues to the trial-and-error design approach. In particular, the behavior of the individual robot cannot be evaluated directly and *per se*: it must be evaluated indirectly by observing the collective behavior of a swarm composed by a large number of individual robots that execute the behavior under analysis.

Notwithstanding its limitations, the trial-and-error approach has been successfully used to develop several collective behaviors, including aggregation [8], chain formation [12], and task allocation [13]. These behaviors are described in more detail in Section 4.

Principled design

Although a general engineering framework for designing robot swarms is not available yet, a number of promising principled design methods have been proposed. These methods borrow concepts and tools from different disciplines and address different issues.

In virtual physics-based design [14] each robot is considered as a virtual particle that exerts forces on other particles—that is, other robots. Each robot is thus immersed in a field of forces that depends on the presence and distance of neighboring robots. The virtual force acting on each robot is $\mathbf{f} = \sum_{i=1}^{k} f_i(d_i)e^{j\theta_i}$, where j denotes the imaginary unit, d_i and θ_i are the distance and the direction of the ith neighboring robot, and the function $f_i(d_i)$is the derivative of an artificial potential function. The Lennard-Jones potential [15] (Figure 1(a)) is commonly used in this context [e.g., 14, 16, 17]. Figure 1(b), (c), and (d) show three examples of the virtual force that acts on a robot depending on the position of its neighboring peers. The designer can associate virtual repulsive forces to obstacles and other objects in the environment to prevent collisions. Each robot estimates the virtual forces that act on it and translates them into motion commands. The main benefit of virtual physics-based design is that it allows the designer to formally prove properties of swarm-level behaviors including stability, convergence, and robustness. An extension of virtual physics-based design is the Hamiltonian method [18]. Starting from a mathematical description of the swarm at the macroscopic level, the Hamiltonian method derives the microscopic behavior that minimizes or maximizes the value of a relevant quantity (e.g., the virtual potential energy of the state of the swarm). The major drawback of virtual physics-based design and of the Hamiltonian method is that they are suitable only for designing spatially-organizing collective behaviors (see Section 4.1).

Control theory is the theoretical framework of a few principled design methods that have been proposed. Some of these methods combine virtual physics with sliding mode control to design robot swarms that perform aggregation, foraging, and pattern formation [19, 20] (see Section 4). Other methods use kinematic equations to model the motion of robots and a set of control-Lyapunov functions to develop an individual behavior for pattern formation [21, 22] (see Section 4). The main advantage of methods based on control theory is that some properties of the resulting robot swarms (e.g., stability and robustness) can be proved using theoretical tools such as Lyapunov stability theory. However, the application of control theory typically relies on assumptions—such as deterministic behavior, global communication, and full synchronization—that are often unrealistic in swarm robotics.

Defining a general link between the desired swarm-level (macroscopic) behavior and the individual (microscopic) behavior is the key issue in the principled design of robot swarms. Some studies showed that, under a series of assumptions, a microscopic implementation can be derived from a macroscopic model [23, 24]. Through analytical means, the parameters of a macroscopic model described by a set of advection-diffusion-reaction partial differential equations are mapped onto the individual behavior. This method was successfully used to design robot swarms that perform task allocation and area coverage. However, the underlying assumptions—such as infinite number of robots and global communication—are often violated in swarm robotics. As a result, the

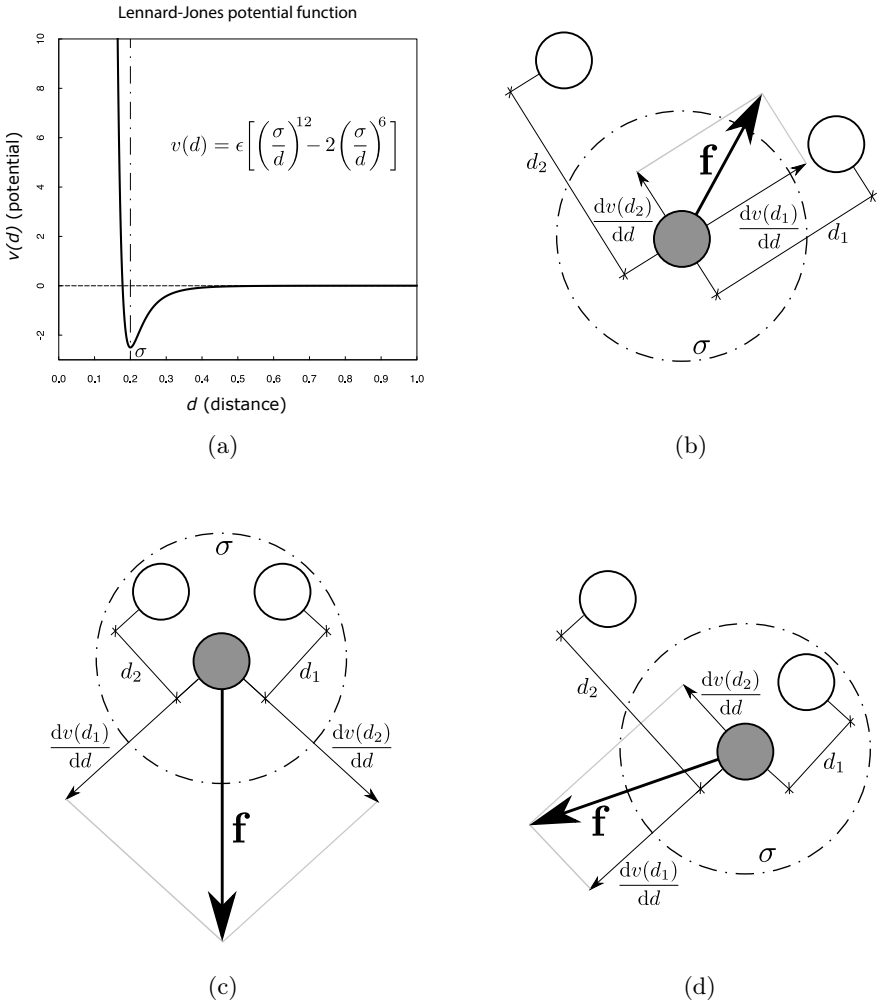

Figure 1: Virtual physics-based design. (a) The Lennard-Jones potential function. The potential v depends on the current distance d between two robots. σ is the desired distance between the robots. ϵ is a parameter called well depth and corresponds to the depth of the potential function. In this example, $\sigma = 0.2$ and $\epsilon = 2.5$. (b), (c), and (d) are examples of the virtual force that acts on a robot (gray-filled circle) depending on the position of two neighboring robots (white-filled circles). In (b), the two neighboring robots are farther than the desired distance σ. Therefore, the robot is attracted by the two neighbors with forces that are determined by the derivative of the Lennard-Jones potential function at the point given by the distance of the neighbors. The resulting force is the sum of the individual forces. In (c), the two neighboring robots are closer than the desired distance σ and hence exert repulsive forces on the robot. In (d), one neighbor is at a distance $d_1 < \sigma$ and thus exert a repulsive force on the robot, while the other neighbor is at a distance $d_2 > \sigma$ and thus attracts the robot.

macroscopic model often fails in predicting the performance of the final system, as shown by a comparative experimental analysis [25]. Another study proposed a method to design the individual switching probabilities in task allocation under soft deadlines [26]. The total amount of work performed by the swarm is described as a Poisson process. Via formal means, the proposed method derives off-line the switching probabilities—that is, the parameters of a Markov chain that describes the individual behavior. The method is based on the assumption that the size and the deadline of the target tasks are all known at design time.

Recently, a formal method borrowed from supervisory control theory has been used to design a segregation behavior [27]. The method is intended to be platform independent: the behavior produced was shown to successfully accomplish the mission both when executed by a swarm of e-pucks [28] and when executed by a swarm of kilobots [29]. Supervisory control theory is a method largely applied in manufacturing for automatically synthesizing control software that drives the behavior of a plant so that specifications are met. In its adaptation to swarm robotics, the method uses a description of the swarm-level requirements and a set of specifications for the behavior of individual robots to generate the individual control logic. In particular, the method requires that the designer specifies the set of possible events that may occur and the corresponding responses of the individual robots necessary to generate the desired collective behavior. In other terms, the method does not support the designer in the most crucial step: devising the appropriate individual behavior that generates the desired swarm-level behavior.

Property-driven design [30] was introduced with the ultimate goal of deriving an individual behavior from swarm-level requirements. The method is based on prescriptive modeling and model checking. The design process is composed of four phases: First, the designer defines a set of desired properties that the swarm should meet. Second, the designer produces a prescriptive model of the swarm and uses model checking to verify that the model complies with the specified properties. Third, the designer implements a simulated version of the robot swarm using the prescriptive model as a blueprint. Fourth, the designer implements the final robot swarm and validates the previous steps. Models are described by means of Markov chains and properties are defined by statements in probabilistic computation tree logic, a probabilistic temporal logic that captures well both temporal and stochastic aspects. Property-driven design is a structured design process supported by formal tools. However, the step from the prescriptive macroscopic model and the correspondent microscopic implementation is not yet automatic, and it is still reliant on the intuition of the designer.

Rather defining a unifying framework to obtain any possible collective behavior, a number of works have proposed the idea of defining a catalogue of design patterns [31, 32, 33]. In the context of swarm robotics, a design pattern is a collection of guidelines to obtaining a specific collective behavior. It must provide: i) a macroscopic model that describes the swarm-level requirements,ii) a description of the microscopic behavior, and iii) a mapping from the parameters of macroscopic model to those of the microscopic behavior. A first example of a design pattern for collective decision-making has been proposed and successfully used to design a collective foraging behavior [34].

An interesting design method has been proposed to obtain self-assembly [35] (see also Section 4) and construction [36] (see also Section 4 and Section 5). The

method promotes the decomposition of the design problem [37]: First, the user provides a global description of the desired aggregate/structure (e.g., the shape of the aggregate, the height of the structure). Second, the method defines a set of steps necessary to build the desired aggregate/structure. Third, the method maps these steps onto individual rules (e.g., rules of motion along the border of the aggregate or over the structure). The method enables the validation of some properties of the final system, such as correctness and convergence. It has been applied successfully to self-assembly and construction. Unfortunately, its applicability is limited to a restricted set of missions.

Finally, although programming and scripting languages cannot be considered as design methods on their own, they can significantly ease the principled design of robot swarms. Two prominent examples are Protoswarm [38] and Buzz [39]. Protoswarm [38] is a scripting language based on the abstraction of an amorphous computational medium [40].The amorphous computational medium assumes that the environment is filled with entities that can compute and communicate locally with each other. Protoswarm enables the definition of behaviors for the individual robots by writing scripts at the level of the swarm. The scripting language features swarm-level primitives that deal both with space and time. These primitives are translated approximately into individual robot behaviors by a runtime library. Recently, the idea of languages based on manipulations of computational mediums has received increasing attentions. New languages have been proposed for the creation of swarm-level programs with a sound mapping between the swarm-level and the individual-level primitives [41, 42]. On the other hand, Buzz [39] is a programming language for heterogeneous robot swarms. Buzz offers primitives to work either at the individual level or at the swarm level. For the time being, the swarm-level primitives mostly serve to create different teams and assign robots to each team. Moreover, Buzz provides mechanisms to share information locally and globally, thanks to a virtual stigmergy mechanism based on a distributed tuple space. One of the prominent features of Buzz is its modularity: primitives can be combined or defined anew to create modules that can be tested, compared, and reused.

2.2 Automatic design

In automatic design methods, the individual behavior of the robots that compose the swarm is generated automatically through an optimization process. The burden of searching for the individual behavior that results in the desired collective behavior moves from the designer to a computer program. The majority of work on automatic design of control software for robot swarms has been produced within the evolutionary robotics domain. A few alternative methods have been recently proposed.

Evolutionary robotics

Evolutionary robotics [43] takes inspiration from the Darwinian principles of natural selection and evolution to automatically design control software for single and multi-robot systems. In evolutionary robotics, the design process starts typically from a population of behaviors generated at random. At each iteration, the behaviors are evaluated over a set of experiments via computer-based simulations. The same behavior is used as control software for all the robots of

the swarm. The evaluation is performed by a fitness function that measures the performance of the swarm. The best scoring behaviors are used to produce the next generation by means of genetic operators:cross-over, which mixes treats from parent solutions to produce a child behavior from them, and/or mutation, which alters small treats of single behaviors. The process terminates when a time limit or a certain performance threshold are attained or when the fitness function stops improving. The individual behavior can be represented in several ways, but the most common one is via an artificial neural network. The evolutionary process searches the parameters of a neural network that, when used as control software on all the robots, maximizes the performance of the swarm.

Despite the large number of works that showed the effectiveness of evolutionary techniques, an engineering methodology for the application of evolutionary robotics is still unavailable [44]. The main issues are that the evolutionary process does not give any guarantee of convergence and the neural networks that result from the design process are black-boxes: they are difficult to analyze and understand. Moreover, most of the behaviors produced so far via evolutionary robotics are relatively simple and thus easily obtainable via manual design. Some promising ideas have been proposed that could contribute to the development of an engineering methodology for evolutionary robotics. Multi-objectivization is deemed to improve the effectiveness of the design process by guiding the evolutionary search in rugged fitness landscapes [45]. Novelty search [46] is deemed to promote diversity among candidate behaviors and improve the exploration of the search space [47]. Finally, the hierarchical decomposition of the control software into modules is deemed to ease the design process [48, 49]. For a recent review and critical discussion of the evolutionary robotics literature, see [50].

Other methods

Because of the limitations of evolutionary robotics [44, 50], other automatic design methods for robot swarms have been proposed in the recent years.

Reinforcement learning is widely adopted in robotics. It has been elegantly defined and successfully used in single-robot scenarios [51, 52]. The multi-robot case has been considered only by few works with limited scope [53, 54]. Swarm robotics appears to pose major problems to reinforcement learning and only a very limited number of studies have been proposed [55, 56]. The results are limited to specific tasks and have been demonstrated in experiments with only few robots.

A number of studies focused on on-line adaptation in multi-robot systems. In these studies, the execution of population-based algorithms is distributed over a group of robots [57]. In this form of embodied evolution the robots are used as computation nodes. Several works have tested the feasibility of this approach, proposing different solutions, including open-ended and task-dependent evolution and the use of finite state machines [58, 59, 60]. The implementation of distributed evolutionary algorithms in robot swarms has been tested in other variants: some study explored the idea of cultural evolution in robot swarms using an imitation-based algorithm [61]. The particle swarm optimization algorithm was compared to genetic algorithms for on-line adaptation and proven to provide a higher degree of diversity in the robot swarm [62, 63].

Another promising and effective approach that has been proposed adopts

a fixed control architecture and focuses on tuning only a small set of parameters. Genetic algorithms and evolutionary strategies were used to optimize the parameters of finite state machines for a cooperative foraging and object clustering [64, 65]. Exhaustive search was used to determine the optimal parameters for self-organized aggregation [66]. A similar approach combines evolutionary computation with virtual-physics based design to learn off-line the parameters for the Lennard-Jones potential function in a navigation task [67].

A recently proposed novel approach to the automatic design of control software for robot swarms is AutoMoDe [68]. In AutoMoDe, the control software is automatically designed in the form of a probabilistic finite state machine. The design process works by combining and fine-tuning preexisting modules, which have parameters that regulate their functioning. A search algorithm optimizes these parameters along with the topology of the probabilistic finite state machines to maximize a task-dependent performance measure. This design method was proven effective in overcoming the reality-gap and in subsequent studies was shown to outperform human designers in designing control software for five different missions [69].

3 Modeling

A robot swarm can be modeled at two levels: the microscopic level, which is the level of the single individuals and the interactions among them, or the macroscopic level, which is the level of the swarm and its collective dynamics.

3.1 Microscopic models

Modeling a robot swarm at the microscopic level involves creating a detailed representation of each individual. A microscopic model can be defined with different levels of abstraction: simple models consider robots as point-masses, while the most complex ones include detailed representations of each sensor and actuator of the robots. Often, microscopic models are inspired by biochemical systems and chemical reactions [70, 71, 24]. The behavior of each individual robot is also an element of microscopic models and, because of its stochastic nature, it is usually represented by probabilistic finite state machines or Markov chains [72].

The large number of robots and interactions involved make microscopic modeling problematic. For this reason, microscopic modeling often relies on computer-based simulations [73, 74]. Simulations are among the most used tools for validation and analysis of robot swarms. The vast majority of the collective behaviors presented in Section 4 have been studied by means of computer-based simulations.

3.2 Macroscopic models

Macroscopic models consider a robot swarm as a whole. The details of the individuals and their behavior are neglected in favor of a model of the system at a higher level. The remainder of this section is devoted to the description of the most important techniques for modeling robot swarms at the macroscopic level.

Rate and differential equations

Rate equations describe the evolution in time of the portion of robots that are in a certain state over the total number of robots. Deriving the rate equations of the swarm from the probabilistic finite state machine that describes the individual behavior is straightforward [75]. First, a variable is defined for each state of the state machine. These variables count the fraction of robots that are in the corresponding states. Second, for each variable a rate equation is defined that describe the time evolution of the variable—that is, the time evolution of the portion of robots in the corresponding state. A rate equation contains a set of parameters, one for each input and output transition of the corresponding state. In a seminal work [75], rate equations were used to model an object clustering behavior. In subsequent works, rate equations have been used to model several other tasks, among which stick pulling [76, 72], wireless network formation [77], aggregation [78], and foraging [13]. The advantage of rate equations is that they allow the derivation of macroscopic models directly from microscopic ones. The main limitations are that the time is assumed to be discrete and motion patterns or complex spatial aspects are difficult to model.

A modeling technique for robot swarms that considers spatiality, stochasticity and noise is based on differential equations [79]. This technique uses Langevin and Fokker-Planck equations. The Langevin equation describes the motion of a robot in an environment populated by other peers. It consists of two components: a deterministic one, which expresses the laws of motion of the robot (microscopic component); and a stochastic one, which describes the effects of the interactions with its peers (macroscopic component). Because of this double nature, the Langevin equation is a mesoscopic model—an intermediate level between microscopic and macroscopic. From the Langevin equation, it is possible to derive a Fokker-Planck equation that describe the dynamics of the entire swarm. The derivation is possible by means of tools of statistical mechanics and problem-dependent intuition. This technique was applied to analyze coordinated motion, aggregation, and foraging [80]. In further studies, the predictions of four models based on Fokker-Planck equations were compared with the results of computer-based simulations and real-robot experiments [81]. The comparison revealed that spatial models are more accurate than non-spatial ones for short time spans, but the difference is very small for long time spans. In principle, the modeling technique based on Langevin and Fokker-Planck equations can be used to model any robot swarm and any collective behavior. However, this technique still depends on the intuition of the designer, who must properly model communication aspects between robots. Moreover, the Fokker-Planck equation is difficult to solve analytically and often requires demanding numerical algorithms. A similar modeling technique uses a set of advection-diffusion-reaction partial differential equations [23]. These equations were used to model the behavior of robot swarms performing task allocation and area coverage [24]. The Gillespie algorithm [82] has been used to numerically approximate stochastic differential equations that model robot swarms of finite size [83, 33, 84]. These equations present non-linearities that prevent the use of analytical methods for their resolution.

Control and stability theory

Classical control and stability theory can be used to verify whether a collective behavior will eventually drive the swarm to a desired state. The first works that adopted control and stability theory in swarm robotics modeled swarms in a one-dimension space, using discrete time and discrete event dynamical systems [85]. By using Lyapunov stability theory, a collective behavior was shown able of social foraging in presence of noise [86, 20]. Other works used delay differential equations to model task allocation and to prove the stability of the configuration obtained [87]. The modeling techniques based on control and stability theory have the advantage of a strong mathematical formulation. However, these techniques rely on several assumptions that are often violated in swarm robotics because of noise, stochasticity, and asynchronism.

Model checking

Model checking is a method for formally and automatically verifying whether a system meets its specifications. In swarm robotics, model checking involves encoding the collective behavior by means of a mathematical model and checking whether the model possesses the desired properties. Properties of a system are expressed as temporal logic formulas.

In one of the first applications of model checking to swarm robotics, researchers used linear temporal logic to define and prove two properties of a robot swarm: safety, which is a property verified if the swarm does not display undesirable behaviors, and liveness, which is a property verified if the dynamics of the swarm evolves over time [88]. A recent work proposes the use of probabilistic computation tree logic to describe desired properties of models expressed by Markov chains [30]. Probabilistic computation tree logic is well suited for swarm robotics because it captures well both temporal and stochastic aspects. In other studies, a high level modeling language called Bio-PEPA was used to analyze the properties of robot swarms [70]. From a description in Bio-PEPA, it is possible to derive different models and analyze them by means of stochastic simulations and model checking.

Recent studies have proposed novel solutions to the main problem of model checking [89, 90]—that is, the state-space explosion problem when the number of robots is higher than few units. These studies present a robust methodology for identifying cutoffs with respect to expressive temporal-epistemic specifications. This methodology enables the verification of properties of robot swarms independently of the number of robots in the swarm.

Markov chains

Markov chains are stochastic processes that undergo transitions between states in a given state space. Markov chains are memoryless: the following state of the process depends only on the current state, and not on the past history. Markov chains are applied as statistical models of many real-world processes [91].

Because of the ability to model stochastic processes, Markov chains are well suited for swarm robotics. Several collective behaviors have been modeled using Markov chains. A first example is aggregation [92, 93]; the predictions of the models of aggregation were then validated using computer-based simulations. More recently, Markov chains have been used to model a protocol of spatially

targeted communication between aerial and ground robots of a swarm [94]. The protocol allows robots to open communication links with target robots depending on their location in space. This enables spatial coordination (i.e., pattern formation, morphogenesis) in the swarm. The model was validated using computer-based simulations and real-robot experiments.

Markov chains enabled the analysis of several collective decision-making instances. For example, the designers could gain insights into the distribution of the number of individual decisions (by the majority rule) necessary to reach consensus [95], and into the effects of a dynamic neighborhood size on the decision dynamics [96]. In other studies, urn models and Markov chains have been used to study collective decision making [97, 96]. The advantage of urn models is their ability to capture qualitatively key features of a system notwithstanding their simplicity. Finally, Markov chains and death-birth processes have been used to study a scenario in which, at any moment in time, each robot can be either inactive or engaged in a task [98]. Death-birth processes enable the estimation of several properties of the swarm, such as the energy consumption, the amount of work accomplished, the time required to complete the task, and the expected cost-reward.

4 Collective behaviors

Collective behaviors are basic behavioral units of robot swarms that can be combined to create complex collective behaviors. Here we describe the main collective behaviors studied in the literature and we divide them into five categories: spatially-organizing behaviors, navigation behaviors, collective decision-making, interaction with humans, and other behaviors.

4.1 Spatially-organizing behaviors

Spatially-organizing behaviors are collective behaviors that focus on how the robots distribute and organize in space.

Aggregation

The goal of aggregation is to group the robots in a region of the environment. Aggregation is a useful building block for many complex behaviors as it allows robots to gather and thus to interact with each other. The implementation of aggregation in robot swarms is often inspired by similar behaviors observed in natural systems such as bacteria, bees, and cockroaches. Aggregation has been obtained with either manual or automatic design methods.

Manual design methods typically adopt a simple probabilistic finite state machine: the robots wander in the environment and, when they find other robots, they decide stochastically whether to stay in their proximity or depart from them. Typically, robots join an aggregate (or leave it) with a probability that is a function of the size of the aggregate itself: the larger the aggregate, the higher the probability of staying. This favors the formation of a single, large aggregate, as small aggregates tend to disband. This basic behavior can be adapted and tuned to obtain either static or moving aggregates [99, 100]. Aggregation has been obtained also via a principled design method based on control theory [19]

(see Section 2). Automatic design methods mostly use artificial evolution to find the parameters of a neural network that produces the desired aggregation behavior. Either static or moving aggregates can be obtained with this approach [101, 92]. Other automatic design approaches work on a fixed control architecture and tune a small set of parameters. This approach successfully produced an aggregation behavior with memoryless robots that are equipped only with a single binary sensor [66].

Aggregation can be modeled using different modeling techniques. Rate equations are particularly suited because of their ability to describe the evolution in time of the portion of robots in a particular state (the aggregate) [78]. Other modeling methods used in the literature are based on Langevin and Fokker-Planck equations [79, 80], on Markov chains [92, 93], and on control and stability theory [102, 103].

Pattern formation

Pattern formation is a behavior that aims at positioning robots in space according to a certain, well defined, pattern. Pattern formation can be useful for a number of purposes such as covering an area with a fixed number of robots, achieving a certain network topology and forming the initial configuration for coordinated motion (see Section 4.2). Examples of pattern formation that often inspire research in swarm robotics can be found both in biology (e.g., the chromatic patterns on some animal's coat) and physics (e.g., crystal formation and Bénard cells).

Pattern formation in robot swarms is typically obtained using virtual physics-based design. As already mentioned in Section 2, in virtual physics-based design robots are considered immersed in the virtual potential field generated by the neighboring robots. Motion commands are computed by each robot based on the sum of the virtual forces exerted by its neighbors. If all the robots exert the same force, this simple mechanism yields an hexagonal lattice [14]. By dividing the swarm in two groups with different attraction/repulsion thresholds, it is possible to obtain a square lattice [16, 17]. Virtual physics can be combined with tools borrowed from control theory. In this case, the stability of the resulting formation can be proved analytically [19, 22] (see Section 2). Virtual springs can be used alternatively to compute the forces of attraction and repulsion. Combined with different interaction rules (e.g., full connectivity, nearest neighbor, K-nearest neighbors), they can produce different patterns [104, 105].

Recently, a pattern formation behavior with a thousand robots has been demonstrated [35]. Few robots act as the seed of the pattern and define the origin and orientation of the coordinate system that is used to build the desired shape. Starting from the seed robots and using also an internal representation of the target pattern, other robots of the swarm gradually join the pattern. Robots localize themselves with respect to the initial seed using an information gradient. The thousand robots have been shown to successfully form different shapes.

An important application of pattern formation is area coverage: when the number of robots is limited, a lattice formation of equally-spaced robots optimizes the coverage of the space [106]. Area coverage is often modeled using differential equations: two examples of differential equations used to model

area coverage are a set of advection–diffusion–reaction partial differential equations [24] and the Fokker–Planck equations [81]. In the latter example, the accuracy of four models based on Fokker-Planck equations was tested by comparing their predictions with the results of computer-based simulations and real-robot experiments.

Chain formation

In chain formation, robots arrange themselves in the environment to create a chain that connects two locations. The chain is then used by other robots as a navigation aid (see Section 4.2). This behavior is inspired by Argentine ants, which form chains of individuals that connect their nest to foraging sites [107].

Chain formation can be developed using different design methods. Typically it is obtained by manually designing control software in the form of a probabilistic finite state machine. The chain is built incrementally from the starting location. The robots that find a growing chain follow it until the end and join it in the last position with a certain probability. The last robot in the chain can always leave the chain with a certain probability. This prevents the chain from becoming entrapped in dead ends and allows an effective exploration of the environment. When the chain reaches the target location, it becomes stable. The robots in the chain might use a tricolor-pattern to indicate the direction of the chain [108, 12]. A variant of this solution is based on a probabilistic finite state machine and network routing. The result is a chain of moving robots [109].

Virtual physics-based design and automatic design methods can also be used to design chain formation. In virtual physics, virtual forces are used to maintain a desired distance between robots in the chain and between robots and walls in order to create chains that strongly depends on the shape of the environment [110]. Concerning automatic design, artificial evolution has been shown able to produce chains of moving robots [111].

Self-assembly and morphogenesis

Self-assembly is the process in which robots physically connect to each other. Self-assembly can be useful, for example, to increase mechanical stability and ease navigation on rough terrains. When the connected robots form a particular pattern or shape, the process is called morphogenesis. Morphogenesis is used when a particular structure allows the swarm to perform a specific task. For instance, a line of connected robots can navigate over a hole whereas a single robot would fall into it. Several natural systems show self-assembly and morphogenesis behaviors: ants are able to create bridges, rafts and walls to perform specific tasks; cells self-organize structures to form tissues and organs.

Self-assembly and morphogenesis can be designed in several ways. These behaviors pose many challenges to the design process: when and how the assembly should start, which robots should connect to each other, which shape should be formed. Each of this problems can be addressed in different ways.

Robots can trigger the self-assembly process when they encounter obstacles or adversities that they are not able to overcome on their own. Empirical studies showed that connected robots are able to navigate in hazardous terrains better than individual robots [112], they can overcome obstacles that a single robot cannot overcome [113], and they can transport heavy objects faster and for

longer distances [114]. Robot swarms have also been demonstrate capable of creating 3D structures through self-assembly [115].

Homogeneous robots can self-organize the assembly process by signaling the docking points in different locations of their bodies. Other robots can then connect stochastically to those docking points. In this way the robots can form different structures, such as lines, stars and circles [116]. Alternatively, the capabilities of heterogeneous robots can ease the process of self-assembly. For example, an aerial robot can recognize the task to perform and indicate to the ground robots which robots should self-assemble and what structure they should create to perform the task [117].

Self-assembly and morphogenesis have not been modeled often in the literature. A study showed that a self-assembly behavior that allows robots to form lines can be modeled using a set of chemical reactions [118]. This set of chemical reactions was then abstracted by a set of differential equations, solved approximately by means of stochastic simulations (e.g., Gillespie algorithm), and compared to computer-based simulations.

Object clustering and assembling

Object clustering and assembling refer to behaviors in which the robots create aggregates of objects. The difference between object clustering and assembling is that in the former the aggregates are clusters of unconnected objects, whereas in the latter the objects must be connected by some kind of physical link. These two behaviors are at the basis of any swarm construction system. For the design of object clustering and assembling, researchers often take inspiration from social insects: brood clustering has been observed in ants, termites can build mounds that are orders of magnitude larger than the single individuals.

Object clustering is usually obtained using a probabilistic finite state machine. The robots explore randomly the environment and react with appropriate responses when they find an object or partially formed clusters. In the simplest form of object clustering, a robot picks up an object and deposit it with a probability that is proportional to the number of other objects perceived [119]. The final position of the clusters can be controlled by marking the ground with colors or using other signals recognizable by the robots [120, 121]. Object clustering can also be obtained via automatic design methods. Recently, a clustering behavior for extremely simple robots was successfully developed through evolutionary robotics [122]: the robots are not capable of arithmetic computation and are only able to detect the presence of an object or another robot in their direct line of sight. Despite these limitations, the swarm is able to successfully create clusters of objects within a limited amount of time. Object clustering was modeled in a seminal work on the use of rate equations in swarm robotics [75].

Concerning assembling, a recent work demonstrated a behavior that enables the creation of arbitrary 3D structures [36]. This solution generates off-line a set of traffic rules and assigns them to the robots, together with a static representation of the target structure. Respecting the traffic rules, a group of climbing robots builds the structure by placing a building block at a time. More details on this work can be found in Section 5.

4.2 Navigation behaviors

Navigation behaviors are collective behaviors that aim at coordinating the movements of a robot swarm.

Collective exploration

Collective exploration includes behaviors whose goal is to explore an environment, or interesting portions of it. Work on collective exploration takes frequently inspiration from behaviors observed in natural systems. Control software for collective exploration is typically implemented in the form of probabilistic finite state machines. Often the swarm relies on static robots that act as way points to guide the navigation of moving robots. To do that, the static robots can form either physical or virtual structures.

Physical structures are usually the result of pattern formation and chain formation (see Section 4.1). Once the physical structure is formed, the moving robots can follow it, way point after way point, to navigate in the environment. In virtual structures, the static robots are not necessarily close to each other, but they are connected by a virtual medium. For example, pre-deployed robots can create a virtual structure between two locations by exchanging messages. Moving robots can exploit these messages for navigation [123, 124]. Similarly, a network of pre-deployed sensors can be used by the robots to navigate towards their goal location [125]. The navigation route is calculated by the robots via a distributed variant of the Bellman-Ford algorithm. An hybrid solution was developed in the Swarmanoid project [126] (see also Section 5). In this solution, a set of aerial robots deploy sequentially to form a chain, using the position of the previously deployed robots to determine their target position. Once deployed, the robots establish also a virtual structure by acting as communication relays.

Lastly, a solution has been proposed in which the robots of a swarm both navigate and guide the navigation of others, simultaneously [127]. While moving, the robots share navigation information between them and hence cooperatively guide each other towards a target location. The advantage of this solution is that it does not bind any robots to a specific location. All the robots can thus move and be involved in other tasks, possibly unrelated to navigation.

Coordinated motion

In coordinated motion, also known as *flocking*, the robots move in formation through the environment, similarly to flocks of birds or schools of fish. In nature, coordinated motion is used by many animals to reduce energy consumption and increase the chance they survive attacks of predators. Flocking can be obtained with either manual or automatic design methods. The most common design method uses virtual physics. Virtual forces of attraction and repulsion maintain a desired constant distance between the robots and a uniform alignment during the motion [128]. The robots are capable of coordinated motion even in absence of a common goal, thanks to the sole knowledge of heading and distance of their neighbors [129]. Under this configuration, it is sufficient to insert few "informed" robots to direct the movement of the other "uninformed" robots, and hence of the whole swarm, toward a goal [130]. Further works showed that this behavior does not require an explicit alignment rule, and thus robots do not need to perceive the orientation of their neighbors. The swarm is still able to navigate with and

without the presence of informed robots [131]. Flocking of a swarm of aerial robots was obtained through evolutionary robotics [132]. Without relying on any external infrastructure, the aerial robots establish and maintain a wireless communication network to connect a base station and a user station that are located on the ground.

In the literature, flocking is typically modeled using differential equations. An example is the application of a method based on a Fokker-Planck equation [79]. In another study, researchers performed preliminary steps towards linking the models of flocking produced in statistical physics with the studies produced in swarm robotics [133]. The authors focused on the alignment of robots and verified the existence of a phase transition between order and disorder that depends on the level of noise and on the neighborhood size. The results were validated using computer-based simulations.

Collective transport

Collective transport refers to a set of behaviors in which the goal of the swarm is to cooperatively move objects from one location to another. The objects are too heavy for a single robot, thus cooperation is necessary. Collective transport can be observed in ant colonies. To achieve collective transport, ants use a trial-and-error process in order to determine the right pulling/pushing direction [134].

Collective transport is usually designed via manual methods or artificial evolution. Different strategies can be employed for transporting the object: robots can connect directly to the object and move it, they can connect to each other and then to the object, or they can surround the object and push it with their movement [114, 135]. Consensus on the direction of movement and cooperation are achieved either through direct or indirect communication. For example, when direct communication is used, robots can agree on a common direction of movement by averaging their individual desired directions [136]. When communication is indirect, robots can position themselves around the object depending on the position already taken by other robots [135], or depending on an estimation of the forces applied by other robots on the object or on their own chassis [137].

As an alternative to reaching consensus on the direction on movement, some robots can form a chain to connect the source and the destination of the objects (see Section 4.1). The chain is then used as navigation aid by other robots that transport the objects [138].

4.3 Collective decision-making

Collective decision making focuses on how a robot swarm can make decisions. Two categories of situations can require a swarm to make a choice: the first category comprises situations in which the robots have to reach a consensus on a single choice among a set of possible alternatives. The behaviors that aim at solving these problems are called *consensus achievement*. The second category is composed of problems in which the robots have to distribute themselves among a set of possible tasks and operate in parallel on those tasks in order to maximize the performance of the system. This process is called *task allocation*.

Consensus achievement

Consensus achievement behaviors allow a robot swarm to converge on a single choice among a set of alternatives. The choice is usually the one that maximizes the performance of the system. Consensus achievement can be observed in many insect species; for example, ants can determine the shortest of different paths using pheromone [139] and bees collectively choose the best nest location among several alternatives [140].

The robots of a swarm can reach consensus using either direct or indirect communication. Several strategies have been proposed in the literature. By mimicking a form of consensus achievement used by cockroaches, it is possible to develop robot swarms that choose a single aggregation zone using indirect communication [99]—that is, decisions are taken using indirect clues, such as the density of neighbors. Consensus achievement can be achieved via quorum sensing, an algorithm inspired by the choice of the best over N alternatives in ants and bees [141]. The algorithm is based on direct communication: robots evaluate alternatives and advertise them through recruiting messages that are broadcast with a frequency proportional to their perceived quality. This allows the swarm to eventually converge on the best alternative. The nest-site selection in honeybee colonies inspired other work: a thorough study of its analytical model and the identification of the parameters that determine its working regime enabled the definition of guidelines for the implementation of the individual robot behavior. This approach was applied to the shortest path selection problem [33]. Consensus achievement has been used also to let the robots agree on a common reference orientation [142]. The algorithm, which uses only relative positioning and local communication, can be used as a preliminary step for collective motion. Finally, two novel strategies were recently proposed: the first is based on a weighted voter model and ensures a high decision accuracy and robustness to noisy assessments of alternatives [83]; the second couples a mechanism of time-modulation with individual robots' decisions based on the majority rule and has been shown to speed up the decision process considerably [143, 144, 145].

Consensus achievement is typically modeled using Markov chains [95, 96, 97, 96]. Another modeling technique used to model consensus achievement is based on Bio-PEPA [70], a process algebra originally introduced for modeling biochemical systems. From a formal specification of the system in Bio-PEPA, the authors were able to analyze the behavior of a swarm that had to identify the shortest path between two possible choices.

Task allocation

Task allocation behaviors deal with the distribution of robots over a set of different tasks. The allocation is usually dynamic and aims at maximizing the overall performance of the swarm. Ants and bees use task allocation. For example, while some individuals are foraging, a number of other individuals are in charge of looking after the larvae. Task allocation is usually obtained through manual design methods. In the first works on task allocation the choice of the robots was limited to whether to engage in a foraging task or remain in the nest, resting. The robots would choose on the basis of the level of energy in the nest (level raised by the preys collected and consumed by robots resting) [146], or on

the basis of individual observation of the environment and of other robots [147]. A similar approach is used in recent works to design task allocation for two sequentially interdependent tasks [148, 149]. The process is based on individual observations of the environment and of the current performance of the swarm, and hence it does not require direct communication between robots. Other works focus on whether, when, and how the robots should perform the overall task or partition it and allocate themselves to one of the subtasks [150]. For example, a robot could choose whether to carry a prey from the source to the nest or store it in a cache [151]. In the latter case, the prey would then be collected by robots waiting on the other side of the cache and carried to the nest. The choice is made on the basis of the estimation of the costs involved.

Some attempts have been made to design task allocation via automatic methods. A recent study demonstrated how self-organized task allocation can be obtained through evolutionary robotics [152]. Differently from the previous attempts that used artificial neural networks [153, 154, 155], the authors used a method of grammatical evolution to automatically design task allocation strategies. This method was proven more effective, and was able to successfully design task allocation both when provided few behavioral building blocks and when the strategy had to be defined anew.

Task allocation is usually tested on foraging. However, there are examples of application of task allocation to other practical problems. The allocation of robots to different operations on a construction site [156] and a stick-pulling problem [157] are two further examples of application. Often, task allocation is used as a testbed for modeling techniques. A set of advection-diffusion-reaction partial differential equations was used to model and design a behavior for task allocation [23]. Delay differential equations were used to model task allocation and verify the stability of the system obtained [87]. A population dynamics model was used to determine some parameters of the individual behavior and the optimal distribution of robots in two task-allocation scenarios [158]. Finally, a Poisson process was used to describe the performance of a swarm performing a set of tasks and to design the individual switching probabilities for task allocation [26].

4.4 Interaction with humans

Robot swarms are designed to work autonomously and to act in a distributed way. These characteristics limit the degree of control that a human operator can exercise on the system. However, there are several cases in which forms of human control over the swarm are necessary. Human-swarm interaction studies how a human operator can control a robot swarm and receive feedback from it. Studies in this field can be categorized on the basis of the nature of interactions that they propose.

The most common approach relies on an intermediate modeling layer between the operator and the swarm. Usually, the modeling layer produces an abstract representation of the robots and their environment that is then displayed to the operator through a graphical user interface. By acting on the GUI, the operator can select robots and send them commands. The selection can contain single robots [159], or a group of robots, which can be selected, for instance, by drawing a rectangular zone that contains the robots in the GUI [160]. A robot controlled by a human operator is perceived by the swarm as just another

robot, and thus the influence of the operator is very limited. This problem can be solved partially by a hierarchical communication architecture in which the operator sends orders to the selected robot, which is called "the sergeant" [161].

Other studies focus on the use of augmented reality. Part of the studies that use augmented reality propose solutions only for the visualization of feedback from the robots to the operator. For instance, an optical see-through head-worn device receives robots' messages, analyses them and augments the environment with their representation [162]. Similarly, firefighters are helped in their mission by a robot swarm, which gives them direction information displayed by augmented helmets [163]. Other works provide bi-directional communication solutions: through a device that display the augmented environment, the operator can also give commands to the robots by acting on the real-time video stream [164].

Finally, there are studies that aim at realizing a direct interaction,without relying on intermediate modeling levels. In fact, creating and maintaining an updated model of the robots and their environment is a demanding task. It often requires ad-hoc infrastructures and it becomes intractable in dynamic (real) environments or when the number of robots is greater than few units. For these reasons techniques of direct interaction based on gestures recognition, face engagement and speech recognition have been proposed. Combinations of such techniques are possible [165, 166, 167]. Performing gesture recognition directly on the robots might lead to mismatches, and thus require distributed consensus algorithms in order to reach an agreement of all the robots on the same gesture [168]. Other works proposed the use of external sensors, such as the Microsoft Kinetic sensor to give commands through gestures to a robot swarm [169].

4.5 Other behaviors

Some notable studies in swarm robotics do not belong in any of the previous categories.

In collective fault detection, the swarm recognizes faulty robots and initiates appropriate responses. Despite being robust to individual robot failures, there might be situations in which robot swarms need to be aware of the presence of faulty robots and react properly. In a pioneering work on collective fault detection, the robots use an algorithm inspired by firefly synchronization [170]. The robots emit a periodic signal and eventually synchronize with each other using a model of pulse-coupled oscillators [171]. If a robot does not synchronize with its neighbors, it is assumed to be faulty and a response can be triggered.

In group size regulation, the robots have the ability to estimate and regulate their number in a group. This ability is useful, for example, when an excessive number of robots in a group lowers the performance of the swarm. One of the first solutions proposed to estimate the number of robots in a group takes inspiration from the behavior of fireflies [172]: the robots emit a signal at random times and count the number of signals perceived over a period. This number is used to estimate and tune the size of the group. An improvement of this algorithm based on a more strict signaling order can obtain more reliable group size estimates [173]. In other studies a set of flying robots aids the aggregation of ground robots [174]. The aerial robots estimate the size of the aggregate and communicate to the ground robots the accordingly adjusted probabilities

Figure 2: **Swarm-bot** – Previously unreleased photo. Copyright: Marco Dorigo.

Figure 3: **Swarmanoid** – Still from the video: *Swarmanoid, the movie*. Copyright: Mauro Birattari *et al.* Reprinted by permission.

of joining or leaving the aggregate. Through this mechanism the ground robots are able to form groups of different sizes. Finally, a recent study proposed an algorithm inspired by cockroaches aggregation under shelters that is able to partition the swarm in groups of different sizes [175]. The aggregation in different groups develops in parallel, therefore the convergence time of the algorithm is independent of the number of groups.

5 Notable systems

Among the vast production of fundamental research work in swarm robotics, a few systems have been shown to be able to perform complex missions. In the following we describe a selection of six notable systems. For each system, we report an illustrative picture that includes a QR code in the bottom-right corner. To watch a demonstrative video, either click on the QR code or scan it through the camera of a mobile device.

Swarm-bot [12] is a robot swarm composed of relatively simple robots, the s-bots, that are able to attach to each other—see Figure 2. The ability to self-assemble, along with control algorithms inspired by self-organized behaviors of social insects, allow the swarm-bot to effectively adapt to its environment. For example, by self-assembling in different shapes, the swarm-bot can navigate through rough terrains and drag objects that are too heavy for single s-bots. The swarm-bot has been shown to be able to find a target, heavy object and retrieve it. In the first phase, the s-bots form a chain between the object and the nest (see chain formation in Section 4). In the second phase, a group of s-bots surround the object, self-assemble, and drag the object to the nest along the path described by the chain.

Swarmanoid [176] is a heterogeneous swarm composed of three types of robots: eye-bots, hand-bots, and foot-bots—see Figure 3. Eye-bots are flying

Figure 4: **TERMES** Still from the video: *Designing collective behavior in a termite-inspired robotic construction team.* Copyright: Justin Werfel *et al.* Reprinted by permission.

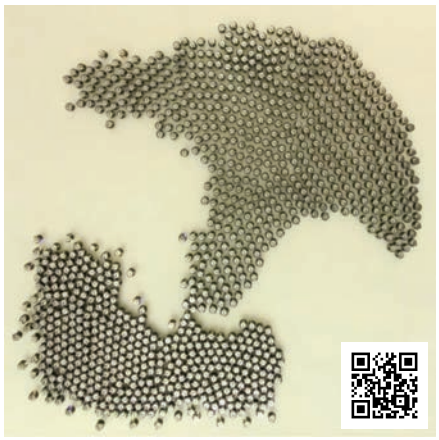

Figure 5: **Thousand-robot Swarm** – Still from the video: *Programmable self-assembly in a thousand-robot swarm.* Copyright: Michael Rubenstein *et al.* Reprinted by permission.

robots specialized in sensing the environment and providing an overview to foot-bots and hand-bots. Hand-bots can climb walls or other vertical surfaces and grab objects, but they cannot move on the ground without the help of other robots. Foot-bots are specialized in moving on the ground and transporting either objects or other robots. The Swarmanoid has been shown to be able to explore an unknown indoor environment, locate a target object (a book), and retrieve it. First, the eye-bots explore the environment, find the book on a shelf, and highlight the path to it. Then, the foot-bots transport a hand-bot to the shelf following the path indicated by the eye-bots. At this point, the hand-bot climbs the shelf, grab the object, and returns to the ground where the foot-bots transport it back to the initial location.

TERMES [36] is a robot swarm inspired by how termites construct mounds—see Figure 4. TERMES allows a user to specify a high level representation of the target 3D structure. From the specified structure, an off-line software generates a set of traffic rules that direct the flow of robots over the growing structure and regulate the building activity. Essentially, this set of rules, namely structpath, is a 2D representation of the structure in which each stack is annotated with its height and a travel direction between each adjacent pair of stacks. Thanks to the structpath and to a static internal representation of the target structure, a group of custom-designed climbing robots proceeds autonomously to the construction process by depositing one brick at a time. The effectiveness of the system has been shown both in a simulation environment and with a real-world implementation.

Thousand-robot Swarm [35] is a robot swarm composed of 1 024 Kilobots—see Figure 5. The Kilobot is a small, low-cost robot equipped only with vibration motors for movement and an infrared transceiver for communication and distance sensing. A swarm of 1 024 Kilobots was demonstrated capable of forming different user-specified patterns. The pattern is built gradually, starting from

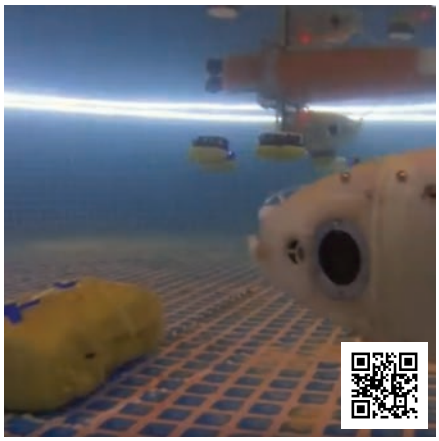

Figure 6: **CoCoRo** – Still from the video: *TYOC#52/52: Final Demonstrator*. Copyright: Thomas Schmickl *et al*. Reprinted by permission.

Figure 7: **BioMachines Lab's Aquatic Robot Swarm** – Previously unreleased photo. Copyright: BioMachines Lab. Printed by permission.

few central robots that act as a seed. The other robots travel along the edge of the pattern under formation and stop in a proper position, according to an internal representation of the target pattern. The thousand-robot swarm is the largest robot swarm demonstrated so far.

CoCoRo [177] is a heterogeneous swarm of underwater robots—see Figure 6. The swarm is composed of a base station and two types of robots: Jeff robots and Lily robots. Jeff robots are fast searching robots, Lily robots are slow information carriers. A CoCoRo can monitor, search, maintain, explore and harvest resources in underwater habitats. The CoCoRo was shown able to locate an object and guide the base station to its position. The Jeff robots search the seabed, until one of them finds the target object and starts recruiting more Jeff robots. At this point the Lily robots start to build a relay chain between the base station and the cluster of Jeff robots. The base station can then navigate to the object location using the information carried by the chain.

BioMachines Lab's Aquatic Robot Swarm [178, 179, 180] is a swarm of 10 aquatic surface robots—see Figure 7. The robots are equipped with few sensors and actuators and thus are relatively inexpensive. The control software was designed automatically using evolutionary robotics (see Section 2). Four collective behaviors were developed to perform four different tasks: flocking, clustering, dispersion, and area coverage. By combining these collective behaviors, the swarm was shown able to perform a complex mission of environmental monitoring: the robot swarm collectively navigates towards an area of interest, optimizes the coverage of the area, monitors the water temperature in the area, clusters, and finally heads back to the base.

6 Some prospective applications

There are a number of possible applications that appear to be appropriate for robot swarms: search and rescue of survivors in disaster areas, humanitarian demining, waste and pollutant removal (i.e., urban waste, oil spills), surveillance, automation and management, health care and medical treatments, exploration of hazardous environments or extraterrestrial planets. In the following, we highlight some of these applications, for which initial promising results have been achieved.

A swarm of aquatic surface robots is currently under development for performing maritime tasks such as patrolling, intruder detection, aquaculture inspection, and environmental monitoring [179]. A swarm of 10 robots has been shown capable of performing a complex environmental monitoring mission (see Section 5). A fundamental characteristic of this swarm is the decentralized control based on a combination of artificial evolution, manual design, and hierarchical decomposition of behaviors. The swarm is composed of relatively simple and cheap robots. Simulated experiments have shown that 1 000 drones are able to patrol the 20 km long coastal strip of the Lampedusa Island [180].

Warehouse automation is a promising application area for robot swarms. Researchers are working on the development of robot swarms to improve the flow of materials in warehouses [181]. The robots will retrieve items, will transport them to human operators, and will keep the warehouse organized. The robots will use communication and ant-inspired algorithms to choose the best paths while they perform these operations [182]. Autonomy and decentralization would allow the addition of more robots at any time, if required by contingencies. Failures of individual robots would not be problematic, as other robots could take over the pending tasks. The parallelization introduced by robot swarms could drastically reduce the average order processing time.

Agriculture appears to be another promising application area. Prototype robot swarms have been developed to automate agricultural processes [183, 184]. Further, several academic projects investigate multi-robot coordination for application in agriculture [185, 186]. These projects focus on the collaboration between one or more aerial robots and a fleet of ground robots. The aerial robots scan the environment and provide information to the ground robots (e.g., locations of weeds to be removed). This information, along with the data collected on the field, allows the ground robots to make decisions about their movements and operations.

Lastly, swarm robotics appears to be the appropriate approach to coordinate large groups of nano-robots for medical applications. Swarms of nano-robots are meant to be injected in a patient to perform tasks including diagnosis and targeted drug-delivery. The main challenge to be faced is the design and realization of the nano-robots. Two different approaches are arising. The first approach focuses on minimalist nano-robots that do not have any sensing/actuating capabilities. Their behavior is fully determined at production time by engineering their material, shape, charge, and coating [187, 188]. The second approach focuses on relatively more complex nano-robots that mimic single-cell organisms [189, 190]. These robots move autonomously, recognize a target, chemically process molecules, and release them on demand. In both approaches, nano-robots rely on swarm behaviors to navigate in blood vessels, protect themselves from macrophages, adhere to target tissues, and accomplish the intended

mission.

7 Conclusions

Swarm robotics is a promising approach for coordinating large groups of robots that need to be fault tolerant, scalable, and flexible. So far, swarm robotics research has focused on developing methods to design collective behaviors (see Section 2) and to model and analyze them (see Section 3). A number of collective behaviors have been produced (see Section 4) and a few notable robot swarms have been demonstrated (see Section 5). The adoption of robot swarms in the real-world is on the horizon, as there exist several prospective fields of application. Among these, the most promising appear to be surveillance, environmental monitoring, agriculture, warehouse automation, and health care (see Section 6). Notwithstanding the significant results achieved so far, there are issues that prevent the immediate uptake of swarm robotics.

The first issue concerns the hardware. Robots that are currently available have limited functionalities and are not sufficiently reliable. Further research is required to produce autonomous robots that can reliably operate in dynamic and possibly hazardous environments. Another issue is the lack of effective ways to interact and command a robot swarms. Although autonomy is a defining property of a robot swarm, a human operator should always be able to control the activity of a swarm. For example, a human operator should be able to stop a robot swarm that is behaving in an unpredicted or dangerous way. Research in human-swarm interaction aims at solving this issue by proposing forms of bidirectional interaction between human operators and robot swarms. However, due mainly to the autonomous and distributed nature of a robot swarm, this issue has not yet been addressed in a fully satisfactory way. A last fundamental issue is the lack of a reliable engineering methodology for specifying, designing, analyzing, verifying, validating, and maintaining a robot swarm. Further research is required to define or adapt formal languages to specify the requirements of robot swarms. In the design process, the definition of general methods to derive the individual behavior from the requirement specification remains an open challenge. The analysis of a robot swarms can be performed by means of several methods, but the lack of well-defined common metrics prevents a proper validation and verification of its properties. In particular, properties such as reliability and safety are often claimed by definition, without the support of an empirical or theoretical analysis. Only few preliminary works have performed steps towards the definition of methods for analyzing, validating, and verifying reliability and safety in robot swarms [191, 192]. The definition of a reliable engineering methodology with all the described components is required to promote the real-world uptake of swarm robotics.

References

[1] N. J. Nilsson. *Shakey the robot.* Tech. rep. 323. Menlo Park, CA, USA: AI Center, SRI International, 1984.

[2] G. Dudek, M. R. M. Jenkin, E. Milios, and D. Wilkes. In: *Auton. Robot.* 3(4), 1996, pp. 375–397.

[3] L. E. Parker. In: *Distributed Autonomous Robotic Systems 4*. Ed. by L. E. Parker et al. Tokyo, Japan: Springer, 2000, pp. 3–12.

[4] L. Iocchi, D. Nardi, and M. Salerno. In: *Balancing Reactivity and Social Deliberation in Multi-agent Systems*. Vol. 2103. LNCS. Berlin, Germany: Springer, 2001, pp. 9–32.

[5] B. P. Gerkey and M. J. Matarić. In: *Int. J. Robot. Res.* 23(9), 2004, pp. 939–954.

[6] S. Mitri, S. Wischmann, D. Floreano, and L. Keller. In: *Biol. Rev.* 88(1), 2013, pp. 31–39.

[7] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo. In: *Swarm Intell.* 7(1), 2013, pp. 1–41.

[8] E. Şahin. In: *Swarm Robotics*. Vol. 3342. LNCS. Berlin, Germany: Springer, 2005, pp. 10–20.

[9] L. Bayindir and E. Şahin. In: *Turk. J. Elec. Eng. & Comp. Sci.* 15(2), 2007, pp. 115–147.

[10] V. Gazi and B. Fidan. In: *Swarm Robotics*. Vol. 4433. LNCS. Berlin, Germany: Springer, 2007, pp. 71–102.

[11] M. O. Rabin. In: *Inform. Control* 6(3), 1963, pp. 230–245.

[12] S. Nouyan, R. Groß, M. Bonani, F. Mondada, and M. Dorigo. In: *IEEE T. Evolut. Comput.* 13(4), 2009, pp. 695–711.

[13] W. Liu and A. F. Winfield. In: *Int. J. Robot. Res.* 29(14), 2010, pp. 1743–1760.

[14] W. M. Spears, D. F. Spears, J. C. Hamann, and R. Heil. In: *Auton. Robot.* 17(2–3), 2004, pp. 137–162.

[15] J. E. Jones. In: *P. Roy. Soc. Lond. A Mat.* 106(738), 1924, pp. 463–477.

[16] W. M. Spears and D. F. Spears. *Physicomimetics: Physics-Based Swarm Intelligence*. Springer, 2012.

[17] C. Pinciroli, M. Birattari, E. Tuci, M. Dorigo, M. del Rey, T. Vinko, and D. Izzo. In: *Proceedings of the Sixth International Conference on Ant Colony Optimization and Swarm Intelligence (ANTS-2008)*. LNCS 5217. Berlin, Germany: Springer, 2008, pp. 347–354.

[18] S. Kazadi. In: *IJICC* 2(4), 2009, pp. 672–694.

[19] V. Gazi. In: *IEEE T. Robot.* 21(6), 2005, pp. 1208–1214.

[20] V. Gazi and K. M. Passino. In: *IEEE T. Syst. Man Cy. B* 34(1), 2004, pp. 539–557.

[21] P. Ögren, M. Egerstedt, and X. Hu. In: *Proceedings of the 40th IEEE Conference on Decision and Control 2001*. Vol. 2. Piscataway, NJ, USA: IEEE Press, 2001, pp. 1150–1155.

[22] M. Egerstedt and X. Hu. In: *IEEE T. Robotic. Autom.* 17(6), 2001, pp. 947–951.

[23] S. Berman, Á. M. Halász, M. A. Hsieh, and V. Kumar. In: *IEEE T. Robot.* 25(4), 2009, pp. 927–937.

[24] S. Berman, V. Kumar, and R. Nagpal. In: *IEEE International Conference on Robotics and Automation (ICRA 2011)*. Piscataway, NJ, USA: IEEE Press, 2011, pp. 378–385.

[25] G. Mermoud, U. Upadhyay, W. C. Evans, and A. Martinoli. In: *Experimental Robotics*. Ed. by O. Khatib et al. Vol. 79. STAR. Berlin, Germany: Springer, 2014, pp. 615–629.

[26] Y. Khaluf, M. Birattari, and H. Hamann. In: *From Animals to Animats 13*. Ed. by A. P. del Pobil et al. Vol. 8575. LNCS. Berlin, Germany: Springer, 2014, pp. 270–279.

[27] Y. K. Lopes, A. B. Leal, T. J. Dodd, and R. Groß. In: *Swarm Intelligence, ANTS 2014*. Ed. by M. Dorigo et al. Vol. 8667. LNCS. Berlin, Germany: Springer, 2014, pp. 62–73.

[28] F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli. In: *9th Conference on Autonomous Robot Systems and Competitions, Robótica 2009*. Castelo Branco, Portugal: IPCB-Instituto Politécnico de Castelo Branco, 2009, pp. 59–65.

[29] M. Rubenstein, C. Ahler, N. Hoff, A. Cabrera, and R. Nagpal. In: *Robot. Auton. Syst.* 62(7), 2014, pp. 966–975.

[30] M. Brambilla, A. Brutschy, M. Dorigo, and M. Birattari. In: *ACM Trans. Auton. Adap.* 9(4), 2014, pp. 17.1–28.

[31] O. Babaoglu, G. Canright, A. Deutsch, G. A. Di Caro, F. Ducatelle, L. M. Gambardella, N. Ganguly, M. Jelasity, R. Montemanni, A. Montresor, and T. Urnes. In: *ACM Trans. Auton. Adap.* 1(1), 2006, pp. 26–66.

[32] L. Gardelli, M. Viroli, and A. Omicini. In: *Multi-Agent Systems and Applications V*. Ed. by H.-D. Burkhard et al. Vol. 4696. LNCS. Berlin, Germany: Springer, 2007, pp. 123–132.

[33] A. Reina, R. Miletitch, M. Dorigo, and V. Trianni. In: *Swarm Intell.* 9(2-3), 2015, pp. 75–102.

[34] A. Reina, G. Valentini, C. Fernández-Oto, M. Dorigo, and V. Trianni. In: *PLoS ONE* 10(10), 2015, e0140950.

[35] M. Rubenstein, A. Cornejo, and R. Nagpal. In: *Science* 345(6198), 2014, pp. 795–799.

[36] J. Werfel, K. Petersen, and R. Nagpal. In: *Science* 343(6172), 2014, pp. 754–758.

[37] R. Nagpal. In: *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1*. AAMAS '02. New York, NY, USA: ACM, 2002, pp. 418–425.

[38] J. Bachrach, J. Beal, and J. McLurkin. In: *Neural Comput. Appl.* 19(6), 2010, pp. 825–847.

[39] C. Pinciroli, A. Lee-Brown, and G. Beltrame. *Buzz: an extensible programming language for self-organizing heterogeneous robot swarms*. Available online at http://arxiv.org/abs/1507.05946. 2015.

[40] J. Beal. In: *Proceedings of the International Workshop on Unconventional Programming Paradigms (UPP)*. Vol. 3566. LNCS. Berlin, Germany: Springer, 2004, pp. 97–97.

[41] M. Viroli, F. Damiani, and J. Beal. In: *Advances in Service-Oriented and Cloud Computing*. Ed. by C. Canal and M. Villari. Vol. 393. CCIS. Berlin, Germany: Springer, 2013, pp. 114–128.

[42] J. Beal and M. Viroli. In: *Philos. T. Roy. Soc. A* 373(2046), 2015.

[43] S. Nolfi and D. Floreano. *Evolutionary Robotics*. Intelligent Robots and Autonomous Agents. Cambridge, MA, USA: MIT Press, 2000.

[44] V. Trianni and S. Nolfi. In: *Artif. Life* 17(3), 2011, pp. 183–202.

[45] V. Trianni and M. López-Ibáñez. In: *PLoS ONE* 10(8), 2015, e0136406–27.

[46] J. Lehman and K. O. Stanley. In: *Evol. Comput.* 19(2), 2011, pp. 189–223.

[47] J. Gomes, P. Urbano, and A. L. Christensen. In: *Swarm Intell.* 7(2-3), 2013, pp. 115–144.

[48] M. Duarte, S. M. Oliveira, and A. L. Christensen. In: *Artificial Life 14: Proceedings of the International Conference on the Synthesis and Simulation of Living Systems*. Ed. by H. Sayama et al. Cambridge, MA, USA: MIT Press, 2014, pp. 657–664.

[49] M. Duarte, S. M. Oliveira, and A. L. Christensen. In: *J. Intell. Robot. Syst.* 78(3-4), 2014, pp. 463–484.

[50] V. Trianni. In: *Front. Robot. AI* 1(13), 2014, pp. 1–6.

[51] L. P. Kaelbling, M. L. Littman, and A. W. Moore. In: *J. Artif. Intell. Res.* 4, 1996, pp. 237–285.

[52] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[53] L. Panait and S. Luke. In: *Auton. Agent Multi-ag.* 11(3), 2005, pp. 387–434.

[54] L. Bušoniu, R. Babuška, and B. De Schutter. In: *Innovations in Multi-Agent Systems and Applications - 1*. Ed. by D. Srinivasan and L. Jain. Vol. 310. SCI. Berlin, Germany: Springer, 2010, pp. 183–221.

[55] M. J. Matarić. In: *Auton. Robot.* 4(1), 1997, pp. 73–83.

[56] M. J. Matarić. In: *J. Exp. Theor. Artif. In.* 10(3), 1998, pp. 357–369.

[57] R. A. Watson, S. G. Ficici, and J. B. Pollack. In: *Robot. Auton. Syst.* 39(1), 2002, pp. 1–18.

[58] N. Bredeche, J.-M. Montanier, W. Liu, and A. F. Winfield. In: *Math. Comp. Model. Dyn.* 18(1), 2012, pp. 101–129.

[59] E. Haasdijk, N. Bredeche, and A. E. Eiben. In: *PLoS ONE* 9(6), 2014, e98466.

[60] L. König and S. Mostaghim. In: *IJICC* 2(4), 2009, pp. 695–723.

[61] A. F. Winfield and M. D. Erbas. In: *Memetic Computing* 3(4), 2011, pp. 261–270.

[62] J. Pugh and A. Martinoli. In: *Proceedings of the IEEE Congress on Evolutionary Computation*. Piscataway, NJ, USA: IEEE press, 2007, pp. 3839–3846.

[63] E. Di Mario and A. Martinoli. In: *Robotica* 32(2), 2014, pp. 193–208.

[64] J. P. Hecker, K. Letendre, K. Stolleis, D. Washington, and M. E. Moses. In: *Swarm Intelligence, ANTS 2012*. Ed. by M. Dorigo et al. Vol. 7461. LNCS. Berlin, Germany: Springer, 2012, pp. 252–259.

[65] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß. In: *Autonomous Agents and Multiagent Systems (AAMAS 2014)*. Ed. by A. Lomuscio et al. Richland, SC, USA: IFAAMAS, 2014, pp. 421–428.

[66] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß. In: *Int. Journal of Robotics Research* 33(8), 2014, pp. 1145–1161.

[67] S. D. Hettiarachchi. "Distributed evolution for swarm robotics". PhD thesis. Laramie, WY: University of Wyoming, 2007.

[68] G. Francesca, M. Brambilla, A. Brutschy, V. Trianni, and M. Birattari. In: *Swarm Intell.* 8(2), 2014, pp. 89–112.

[69] G. Francesca, M. Brambilla, A. Brutschy, L. Garattoni, R. Miletitch, G. Podevijn, A. Reina, T. Soleymani, M. Salvaro, C. Pinciroli, F. Mascia, V. Trianni, and M. Birattari. In: *Swarm Intell.* 9(2-3), 2015, pp. 125–152.

[70] M. Massink, M. Brambilla, D. Latella, M. Dorigo, and M. Birattari. In: *Swarm Intell.* 7(2–3), 2013, pp. 201–228.

[71] N. Napp, S. Burden, and E. Klavins. In: *Auton. Robot.* 30(1), 2010, pp. 57–71.

[72] A. Martinoli, K. Easton, and W. Agassounon. In: *Int. J. Robot. Res.* 23(4-5), 2004, pp. 415–436.

[73] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. A. Di Caro, F. Ducatelle, M. Birattari, L. M. Gambardella, and M. Dorigo. In: *Swarm Intell.* 6(4), 2012, pp. 271–295.

[74] J. Kramer and M. Scheutz. In: *Auton. Robot.* 22(2), 2007, pp. 101–132.

[75] A. Martinoli, A. J. Ijspeert, and F. Mondada. In: *Robot. Auton. Syst.* 29(1), 1999, pp. 51–63.

[76] K. Lerman, A. Galstyan, A. Martinoli, and A. J. Ijspeert. In: *Artif. Life* 7(4), 2001, pp. 375–393.

[77] A. F. Winfield, W. Liu, J. Nembrini, and A. Martinoli. In: *Swarm Intell.* 2(2-4), 2008, pp. 241–266.

[78] R. O'Grady, C. Pinciroli, A. L. Christensen, and M. Dorigo. In: *9th Conference on Autonomous Robot Systems and Competitions, Robótica 2009*. Castelo Branco, Portugal: IPCB-Instituto Politécnico de Castelo Branco, 2009, pp. 113–119.

[79] H. Hamann and H. Wörn. In: *Swarm Intell.* 2(2–4), 2008, pp. 209–239.

[80] T. Schmickl, H. Hamann, H. Wörn, and K. Crailsheim. In: *Robot. Auton. Syst.* 57(9), 2009, pp. 913–921.

[81] A. Prorok, N. Correll, and A. Martinoli. In: *Int. J. Robot. Res.* 30(5), 2011, pp. 574–589.

[82] D. T. Gillespie. In: *J. Phys. Chem.* 81(25), 1977, pp. 2340–2361.

[83] G. Valentini, H. Hamann, and M. Dorigo. In: *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems*. Ed. by A. Lomuscio et al. AAMAS '14. Richland, SC, USA: IFAAMAS, 2014, pp. 45–52.

[84] H. Hamann, G. Valentini, Y. Khaluf, and M. Dorigo. In: *Parallel Problem Solving from Nature – PPSN XIII*. Ed. by T. Bartz-Beielstein et al. Vol. 8672. LNCS. Berlin, Germany: Springer, 2014, pp. 181–190.

[85] Y. Liu, K. M. Passino, and M. M. Polycarpou. In: *IEEE T. Automat. Contr.* 48(1), 2003, pp. 76–95.

[86] Y. Liu and K. M. Passino. In: *IEEE T. Automat. Contr.* 49(1), 2004, pp. 30–44.

[87] M. A. Hsieh, Á. Halász, S. Berman, and V. Kumar. In: *Swarm Intell.* 2(2–4), 2008, pp. 121–141.

[88] A. F. Winfield, J. Sa, M. C. Fernandez-Gago, C. Dixon, and M. Fisher. In: *Int. J. Adv. Robot. Syst.* 2(4), 2005, pp. 363–370.

[89] P. Kouvaros and A. Lomuscio. In: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. Palo Alto, CA, USA: AAAI Press, 2015, pp. 2081–2088.

[90] P. Kouvaros and A. Lomuscio. In: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*. Palo Alto, CA, USA: AAAI Press / IJCAI, 2015, pp. 1083–1089.

[91] J. G. Kemeny and J. L. Snell. *Finite Markov Chains*. Vol. 356. Princeton, NJ: Van Nostrand, 1960.

[92] O. Soysal, E. Bahçeci, and E. Şahin. In: *Turk. J. Elec. Eng. & Comp. Sci.* 15(2), 2007, pp. 199–225.

[93] N. Correll and A. Martinoli. In: *Int. J. Robot. Res.* 30(5), 2011, pp. 615–626.

[94] N. Mathews, G. Valentini, A. L. Christensen, R. O'Grady, A. Brutschy, and M. Dorigo. In: *Auton. Robot.* 38(4), 2015, pp. 439–457.

[95] G. Valentini, M. Birattari, and M. Dorigo. In: *Proceedings of the European Conference on Complex Systems 2012*. Ed. by T. Gilbert et al. Springer Proceedings in Complexity. Berlin, Germany: Springer, 2013, pp. 651–658.

[96] G. Valentini and H. Hamann. In: *Swarm Intell.* 8(2–3), 2015, pp. 153–176.

[97] H. Hamann. In: *Swarm Intell.* 7(2-3), 2013, pp. 145–172.

[98] Y. Khaluf, M. Pace, F. Rammig, and M. Dorigo. *Integrals of Markov processes with application to swarm robotics modelling*. Tech. rep. TR/IRIDIA/2012-020. IRIDIA, Université Libre de Bruxelles, Belgium, 2012.

[99] S. Garnier, C. Jost, R. Jeanson, J. Gautrais, M. Asadpour, G. Caprari, and G. Theraulaz. In: *Advances in Artificial Life*. Vol. 3630. LNAI. Berlin, Germany: Springer, 2005, pp. 169–178.

[100] O. Soysal and E. Şahin. In: *Proceedings of the IEEE Swarm Intelligence Symposium*. Piscataway, NJ, USA: IEEE Press, 2005, pp. 325–332.

[101] V. Trianni, R. Groß, T. H. Labella, E. Şahin, and M. Dorigo. In: *Advances in Artificial Life: 7th European Conference – ECAL 2003*. Vol. 2801. LNAI. Berlin, Germany: Springer, 2003, pp. 865–874.

[102] V. Gazi and K. M. Passino. In: *IEEE T. Automat. Contr.* 48(4), 2003, pp. 692–696.

[103] V. Gazi and K. M. Passino. In: *Int. J. Control* 77(18), 2004, pp. 1567–1579.

[104] B. Shucker and J. K. Bennett. In: *Distributed Autonomous Robotic Systems 6*. Ed. by R. Alami et al. Tokyo, Japan: Springer, 2007, pp. 379–388.

[105] B. Shucker, T. D. Murphey, and J. K. Bennett. In: *IEEE T. Robot.* 24(6), 2008, pp. 1405–1415.

[106] A. Howard, M. J. Matarić, and G. S. Sukhatme. In: *Proceedings of the 2002 International Symposium on Distributed Autonomous Robotic Systems (DARS 2002)*. Piscataway, NJ, USA: IEEE Press, 2002, pp. 299–308.

[107] J.-L. Deneubourg, S. Aron, S. Goss, and J. M. Pasteels. In: *J. Insect Behav.* 3(2), 1990, pp. 159–168.

[108] S. Nouyan, A. Campo, and M. Dorigo. In: *Swarm Intell.* 2(1), 2008, pp. 1–23.

[109] F. Ducatelle, G. A. Di Caro, C. Pinciroli, F. Mondada, and L. M. Gambardella. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*. Los Alamitos, CA, USA: IEEE Computer Society Press, 2011, pp. 4981–4988.

[110] P. M. Maxim, W. M. Spears, and D. F. Spears. In: *Proceedings of the IFAC Workshop on Networked Robotics*. Oxford, UK: Elsevier, 2009, pp. 19–24.

[111] V. Sperati, V. Trianni, and S. Nolfi. In: *Proceedings of the 7th International Conference on Swarm Intelligence (ANTS 2010)*. Vol. 6234. LNCS. Berlin, Germany: Springer, 2010, pp. 155–166.

[112] R. O'Grady, R. Groß, A. L. Christensen, and M. Dorigo. In: *Auton. Robot.* 28(4), 2010, pp. 439–455.

[113] F. Mondada, M. Bonani, A. Guignard, S. Magnenat, C. Studer, and D. Floreano. In: *Proceedings of the VIIIth European Conference on Artificial Life*. Vol. 3630. LNCS. Berlin, Germany: Springer, 2005, pp. 282–291.

[114] R. Groß and M. Dorigo. In: *Int. J. Bio-Inspir. Com.* 1(1–2), 2009, pp. 1–13.

[115] P. Levi and S. Kernbach. *Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution*. 1st. Springer, 2010.

[116] R. O'Grady, A. L. Christensen, and M. Dorigo. In: *IEEE T. Robot.* 25(3), 2009, pp. 738–743.

[117] N. Mathews, A. Stranieri, A. Scheidler, and M. Dorigo. In: *Proceedings of 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*. Ed. by V. Conitzer et al. Richland, SC, USA: IFAAMAS, 2012, pp. 97–104.

[118] W. C. Evans, G. Mermoud, and A. Martinoli. In: *IEEE International Conference on Robotics and Automation (ICRA 2010)*. Piscataway, NJ, USA: IEEE Press, 2010, pp. 1438–1445.

[119] R. Beckers, O. Holland, and J.-L. Deneubourg. In: *Prerational Intelligence: Adaptive Behavior and Intelligent Systems Without Symbols and Logic, Volume 1, Volume 2 Prerational Intelligence: Interdisciplinary Perspectives on the Behavior of Natural and Artificial Systems, Volume 3*. Ed. by H. Cruse et al. Vol. 26. Studies in Cognitive Systems. Netherlands: Springer, 2000, pp. 1008–1022.

[120] C. Melhuish, J. Welsby, and C. Edwards. In: *Proceedings of Towards Intelligent Mobile Robots (TIMR'99)*. Manchester, UK: The University of Manchester, 1999.

[121] R. L. Stewart and R. A. Russell. In: *Adapt. Behav.* 14(1), 2006, pp. 21–51.

[122] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß. In: *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*. AAMAS '14. Richland, SC, USA: IFAAMAS, 2014, pp. 421–428.

[123] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee. In: *Auton. Robot.* 11(3), 2001, pp. 319–324.

[124] G. A. Di Caro, F. Ducatelle, and L. M. Gambardella. In: *Applications of Evolutionary Computing*. Vol. 5484. LNCS. Berlin, Germany: Springer, 2009, pp. 21–30.

[125] K. J. O'Hara and T. R. Balch. In: *Distributed Autonomous Robotic Systems 6*. Ed. by R. Alami et al. Tokyo, Japan: Springer, 2007, pp. 305–314.

[126] T. Stirling and D. Floreano. In: *Proceedings of the 7th International Conference on Swarm Intelligence (ANTS 2010)*. LNCS. Berlin, Germany: Springer, 2010, pp. 562–563.

[127] F. Ducatelle, G. A. Di Caro, A. Förster, M. Bonani, M. Dorigo, S. Magnenat, F. Mondada, R. O'Grady, C. Pinciroli, P. Rétornaz, et al. In: *Swarm Intell.* 8(1), 2014, pp. 1–33.

[128] C. W. Reynolds. In: *Comp. Graph.* 21(4), 1987, pp. 25–34.

[129] A. E. Turgut, H. Çelikkanat, F. Gökçe, and E. Şahin. In: *Swarm Intell.* 2(2–4), 2008, pp. 97–120.

[130] H. Çelikkanat and E. Şahin. In: *Neural Comput. Appl.* 19(6), 2010, pp. 849–865.

[131] E. Ferrante, A. E. Turgut, C. Huepe, A. Stranieri, C. Pinciroli, and M. Dorigo. In: *Adapt. Behav.* 20(6), 2012, pp. 460–477.

[132] S. Hauert, J.-C. Zufferey, and D. Floreano. In: *Auton. Robot.* 26(1), 2009, pp. 21–32.

[133] A. E. Turgut, C. Huepe, H. Çelikkanat, F. Gökçe, and E. Şahin. In: *Proceedings of the 6th International Conference on Ant Colony Optimization and Swarm Intelligence, ANTS 2008*. Vol. 5217. LNCS. Berlin, Germany: Springer, 2008, pp. 108–119.

[134] C. R. Kube and E. Bonabeau. In: *Robot. Auton. Syst.* 30(1–2), 2000, pp. 85–101.

[135] S. Wilson, T. P. Pavlic, G. P. Kumar, A. Buffin, S. C. Pratt, and S. Berman. In: *Swarm Intell.* 8(4), 2014, pp. 303–327.

[136] A. Campo, S. Nouyan, M. Birattari, R. Groß, and M. Dorigo. In: *Ant Colony Optimization and Swarm Intelligence*. Ed. by M. Dorigo et al. Vol. 4150. LNCS. Berlin, Germany: Springer, 2006, pp. 191–202.

[137] G. Baldassarre, D. Parisi, and S. Nolfi. In: *Artif. Life* 12(3), 2006, pp. 289–311.

[138] S. Nouyan, R. Groß, M. Dorigo, M. Bonani, and F. Mondada. In: *Proceedings of the 9th International Conference on Intelligent Autonomous Systems*. Amsterdam, Netherlands: IOS Press, 2006, pp. 433–442.

[139] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, G. Theraulaz, and E. Bonabeau. *Self-Organization in Biological Systems*. Princeton Studies in Complexity. Princeton, NJ: Princeton University Press, 2001.

[140] I. D. Couzin, J. Krause, N. R. Franks, and S. A. Levin. In: *Nature* 433(7025), 2005, pp. 513–516.

[141] C. A. C. Parker and Z. Hong. In: *Int. J. Robot. Res.* 30(5), 2011, pp. 524–535.

[142] I. Navarro and F. Matía. In: *Auton. Robot.* 33(4), 2012, pp. 445–465.

[143] M. A. Montes de Oca, E. Ferrante, A. Scheidler, C. Pinciroli, M. Birattari, and M. Dorigo. In: *Swarm Intell.* 5(3-4), 2011, pp. 305–327.

[144] G. Valentini, H. Hamann, and M. Dorigo. In: *Proceedings of the 14th Int. Conf. on Autonomous Agents and Multiagent Systems*. Ed. by R. Bordini et al. AAMAS '15. Richland, SC, USA: IFAAMAS, 2015, pp. 1305–1314.

[145] G. Valentini, E. Ferrante, H. Hamann, and M. Dorigo. In: *JAAMAS*, 2015, pp. 1–28.

[146] M. J. B. Krieger and J.-B. Billeter. In: *Robot. Auton. Syst.* 30(1–2), 2000, pp. 65–84.

[147] W. Agassounon and A. Martinoli. In: *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems*. Richland, SC, USA: IFAAMAS, 2002, pp. 1090–1097.

[148] G. Pini, A. Brutschy, C. Pinciroli, M. Dorigo, and M. Birattari. In: *Adapt. Behav.* 21(2), 2013, pp. 117–135.

[149] A. Brutschy, G. Pini, C. Pinciroli, M. Birattari, and M. Dorigo. In: *Auton. Agent Multi-ag.* 28(1), 2014, pp. 101–125.

[150] G. Pini, A. Brutschy, M. Frison, A. Roli, M. Dorigo, and M. Birattari. In: *Swarm Intell.* 5(3–4), 2011, pp. 283–304.

[151] A. Brutschy, L. Garattoni, M. Brambilla, G. Francesca, G. Pini, M. Dorigo, and M. Birattari. In: *Swarm Intell.* 9(1), 2015, pp. 1–22.

[152] E. Ferrante, A. E. Turgut, E. Duéñez-Guzmán, M. Dorigo, and T. Wenseleers. In: *PLOS Comput. Biol.* 11(8), 2015, e1004273.

[153] E. Tuci. In: ed. by M. Dorigo et al. Vol. 8667. LNCS. Berlin, Germany: Springer, 2014, pp. 98–109.

[154] G. S. Nitschke, M. C. Schut, and A. E. Eiben. In: *Swarm Evol. Comput.* 2, 2012, pp. 25–38.

[155] G. S. Nitschke, A. E. Eiben, and M. C. Schut. In: *Genet. Program. Evol. M.* 13(4), 2012, pp. 493–536.

[156] S.-k. Yun, M. Schwager, and D. Rus. In: *Robotics Research*. Ed. by C. Pradalier et al. Vol. 70. STAR. Berlin, Germany: Springer, 2011, pp. 607–623.

[157] Á. M. Halász, Y. Liang, M. A. Hsieh, and H.-J. Lai. In: *Distributed Autonomous Robotic Systems*. Ed. by A. Martinoli et al. Vol. 83. STAR. Berlin, Germany: Springer, 2013, pp. 403–416.

[158] N. Correll. In: *IEEE International Conference on Robotics and Automation (ICRA 2008)*. Piscataway, NJ, USA: IEEE Press, 2008, pp. 3302–3307.

[159] S. Bashyal and G. K. Venayagamoorthy. In: *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE*. Piscataway, NJ, USA: IEEE Press, 2008, pp. 1–8.

[160] A. Kolling, K. Sycara, S. Nunnally, and M. Lewis. In: *JHRI* 2(2), 2013, pp. 103–128.

[161] D. J. Bruemmer, D. D. Dudenhoeffer, and J. L. Marble. *Mixed-initiative remote characterization using a distributed team of small robots*. Tech. rep. WS-01-01/WS01-01-005. AAAI, 2001.

[162] M. Daily, Y. Cho, K. Martin, and D. Payton. In: *Proceedings of the 36th Annual Hawaii International Conference on System Sciences, 2003*. Piscataway, NJ, USA: IEEE Press, 2003, pp. 125–130.

[163] A. M. Naghsh, J. Gancet, A. Tanoto, and C. Roast. In: *Robot and Human Interactive Communication, 2008. RO-MAN 2008. The 17th IEEE International Symposium on*. Piscataway, NJ, USA: IEEE press, 2008, pp. 255–260.

[164] F. Ghiringhelli, J. Guzzi, G. A. Di Caro, V. Caglioti, L. M. Gambardella, and A. Giusti. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*. Piscataway, NJ, USA: IEEE Press, 2014, pp. 1195–1201.

[165] A. Couture-Beil, R. T. Vaughan, and G. Mori. In: *Computer and Robot Vision (CRV), 2010 Canadian Conference on Computer and Robot Vision (CRV)*. Piscataway, NJ, USA: IEEE Press, 2010, pp. 159–166.

[166] S. Pourmehr, V. M. Monajjemi, R. T. Vaughan, and G. Mori. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013)*. Piscataway, NJ, USA: IEEE Press, 2013, pp. 137–142.

[167] J. Nagi, A. Giusti, L. M. Gambardella, and G. A. Di Caro. In: *Proceedings of the 27th IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Piscataway, NJ, USA: IEEE Press, 2014, pp. 3834–3841.

[168] A. Giusti, J. Nagi, L. M. Gambardella, and G. A. Di Caro. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2012)*. Piscataway, NJ, USA: IEEE Press, 2012, pp. 551–558.

[169] G. Podevijn, R. O'Grady, Y. S. G. Nashed, and M. Dorigo. In: *Towards Autonomous Robotic Systems - 14th Annual Conference, TAROS 2013*. Ed. by A. Natraj et al. Vol. 8069. LNCS. Berlin, Germany: Springer, 2013, pp. 390–403.

[170] A. L. Christensen, R. O'Grady, and M. Dorigo. In: *IEEE T. Evolut. Comput.* 13(4), 2009, pp. 754–766.

[171] E. M. Izhikevich. In: *IEEE T. Neural Networ.* 10(3), 1999, pp. 508–526.

[172] C. Melhuish, O. Holland, and S. Hoddell. In: *Robot. Auton. Syst.* 28(2–3), 1999, pp. 207–216.

[173] M. Brambilla, C. Pinciroli, M. Birattari, and M. Dorigo. In: *Fourteenth International Conference on Advanced Robotics – ICAR 2009*. Proceedings on CD-ROM, paper ID 137. 2009, p. 6.

[174] C. Pinciroli, R. O'Grady, A. L. Christensen, and M. Dorigo. In: *Proceedings of the Seventh International Conference on Ant Colony Optimization and Swarm Intelligence (ANTS-2010)*. Vol. 6234. LNCS. Berlin, Germany: Springer, 2010, pp. 558–559.

[175] C. Pinciroli, R. O'Grady, A. L. Christensen, M. Birattari, and M. Dorigo. In: *Journal of SICE* 52(3), 2013, pp. 213–226.

[176] M. Dorigo, D. Floreano, L. M. Gambardella, F. Mondada, S. Nolfi, T. Baaboura, M. Birattari, M. Bonani, M. Brambilla, A. Brutschy, D. Burnier, A. Campo, A. L. Christensen, A. Decugnière, G. A. Di Caro, F. Ducatelle, E. Ferrante, A. Förster, J. Guzzi, V. Longchamp, S. Magnenat, J. Martinez Gonzales, N. Mathews, M. A. Montes de Oca, R. O'Grady, C. Pinciroli, G. Pini, P. Rétornaz, J. Roberts, V. Sperati, T. Stirling, A. Stranieri, T. Stützle, V. Trianni, E. Tuci, A. E. Turgut, and F. Vaussard. In: *IEEE Robot. Autom. Mag.* 20(4), 2013, pp. 60–71.

[177] T. Schmickl, R. Thenius, C. Moslinger, J. Timmis, A. Tyrrell, M. Read, J. Hilder, J. Halloy, A. Campo, C. Stefanini, L. Manfredi, S. Orofino, S. Kernbach, T. Dipper, and D. Sutantyo. In: *Fifth IEEE Conference on Self-Adaptive and Self-Organizing Systems Workshops (SASOW 2011)*. Piscataway, NJ, USA: IEEE Press, 2011, pp. 120–126.

[178] M. Duarte, V. Costa, J. C. Gomes, T. Rodrigues, F. Silva, S. M. Oliveira, and A. L. Christensen. In: *arXiv–CoRR* abs/1511.03154, 2015.

[179] A. L. Christensen, S. M. Oliveira, O. Postolache, M. J. de Oliveira, S. Sargento, P. Santana, L. Nunes, F. Velez, P. Sebastiao, V. Costa, M. Duarte, J. Gomes, T. Rodrigues, and F. Silva. In: *Proceedings of the International Conference on Agents and Artificial Intelligence (ICAART)*. Setúbal, Portugal: SCITEPRESS, 2015, pp. 548–555.

[180] M. Duarte, S. M. Oliveira, and A. L. Christensen. In: *Proceedings of the International Conference on the Synthesis and Simulation of Living Systems (ALIFE)*. Cambridge, MA, USA: MIT Press, 2014, pp. 785–792.

[181] *Warehouse robots get smarter with ant intelligence*. Available at `http://spectrum.ieee.org/automaton/robotics/industrial-robots/warehouse-robots-get-smarter-with-ant-intelligence`. 2012.

[182] *Swarming and transporting*. Available at `http://www.fraunhofer.de/en/press/research-news/2012/march/swarming-and-transporting.html`. 2012.

[183] *Prospero. Dorhout R&D LLC*. Available at `http://www.businessinsider.com/presenting-the-robot-farmers-of-the-future-2011-12`. Accessed January, 2016.

[184] *Harvest Automation, Inc.* Available at `https://www.harvestai.com`. Accessed January, 2016.

[185] *RHEA project*. Available at `http://www.rhea-project.eu`. Accessed January, 2016.

[186] *ASETA project*. Available at `http://plen.ku.dk/english/research/crop_sciences/plant_protection/aseta/`. Accessed January, 2016.

[187] S. Hauert and S. N. Bhatia. In: *Trends Biotechnol.* 32(9), 2014. Special Issue: Next Generation Therapeutics, pp. 448–455.

[188] S. Hauert, S. Berman, R. Nagpal, and S. N. Bhatia. In: *Nano Today* 8(6), 2013, pp. 566–576.

[189] *Chemical robots*. Available at `http://www.chobotix.cz`.

[190] N. Sarvašová, P. Ulbrich, V. Tokárová, A. Zadražil, and F. Štěpánek. In: *Powder Technol.* 278, 2015, pp. 17 –25.

[191] A. F. Winfield and J. Nembrini. In: *IJMIC* 1(1), 2006, pp. 30–37.

[192] J. D. Bjerknes and A. F. Winfield. In: *Distributed Autonomous Robotic Systems*. Ed. by A. Martinoli, F. Mondada, N. Correll, G. Mermoud, M. Egerstedt, M. A. Hsieh, L. E. Parker, and K. Støy. Vol. 83. STAR. Berlin, Germany: Springer, 2013, pp. 431–444.