Swarm Intell (2014) 8:1–33
DOI 10.1007/s11721-013-0089-4

Cooperative navigation in robotic swarms

Frederick Ducatelle · Gianni A. Di Caro · Alexander Förster · Michael Bonani · Marco Dorigo · Stéphane Magnenat · Francesco Mondada · Rehan O'Grady · Carlo Pinciroli · Philippe Rétornaz · Vito Trianni · Luca M. Gambardella

Received: 6 November 2012 / Accepted: 22 October 2013 / Published online: 23 November 2013

Guest editor: Roderich Groß.

Electronic supplementary material The online version of this article (doi:10.1007/s11721-013-0089-4) contains supplementary material, which is available to authorized users.

F. Ducatelle · G.A. Di Caro (✉) · A. Förster · L.M. Gambardella
IDSIA, USI/SUPSI, Galleria 1, 6928 Manno-Lugano, Switzerland
e-mail: gianni@idsia.ch

F. Ducatelle
e-mail: fducatelle@gmail.com

A. Förster
e-mail: alexander@idsia.ch

L.M. Gambardella
e-mail: luca@idsia.ch

M. Bonani · S. Magnenat · F. Mondada · P. Rétornaz
EPFL, ME A3 484 (Bâtiment ME), Station 9, 1015 Lausanne, Switzerland

M. Bonani
e-mail: michael.bonani@epfl.ch

S. Magnenat
e-mail: stephane.magnenat@mavt.ethz.ch

F. Mondada
e-mail: francesco.mondada@epfl.ch

P. Rétornaz
e-mail: philippe.retornaz@epfl.ch

M. Dorigo · R. O'Grady · C. Pinciroli · V. Trianni
IRIDIA, CoDE, ULB, Avenue F. Roosevelt 50, 1050 Brussels, Belgium

M. Dorigo
e-mail: mdorigo@ulb.ac.be

R. O'Grady
e-mail: rogrady@ulb.ac.be

Abstract We study cooperative navigation for robotic swarms in the context of a general event-servicing scenario. In the scenario, one or more events need to be serviced at specific locations by robots with the required skills. We focus on the question of how the swarm can inform its members about events, and guide robots to event locations. We propose a solution based on delay-tolerant wireless communications: by forwarding navigation information between them, robots cooperatively guide each other towards event locations. Such a collaborative approach leverages on the swarm's intrinsic redundancy, distribution, and mobility. At the same time, the forwarding of navigation messages is the only form of cooperation that is required. This means that the robots are free in terms of their movement and location, and they can be involved in other tasks, unrelated to the navigation of the searching robot. This gives the system a high level of flexibility in terms of application scenarios, and a high degree of robustness with respect to robot failures or unexpected events. We study the algorithm in two different scenarios, both in simulation and on real robots. In the first scenario, a single searching robot needs to find a single target, while all other robots are involved in tasks of their own. In the second scenario, we study collective navigation: all robots of the swarm navigate back and forth between two targets, which is a typical scenario in swarm robotics. We show that in this case, the proposed algorithm gives rise to synergies in robot navigation, and it lets the swarm self-organize into a robust dynamic structure. The emergence of this structure improves navigation efficiency and lets the swarm find shortest paths.

Keywords Swarm robotics · Cooperative navigation · Self-organization

1 Introduction

In this paper, we present a new algorithm for cooperative navigation in swarm robotics. With *navigation*, we refer to the task of finding a collision-free path for a robotic system to travel from one place to another. *Swarm robotics* is the study of large groups of relatively simple robots that interact and cooperate with each other in order to jointly solve tasks that are outside each robot's own capabilities (Dorigo and Sahin 2004). Such task solving typically relies on *self-organization* and *emergence*, meaning that swarm's organization comes from within the system (i.e., is not imposed from outside), and comes about in a decentralized way, from local interactions between individual robots (De Wolf and Holvoet 2005). Algorithms in swarm robotics mostly rely on *cooperation* and *simple interactions* between robots, rather than on complex individual behaviors that require powerful sensory capabilities. Concretely, in the context of *navigation*, this means that the focus is on cooperative navigation, where robots guide each other, rather than on the use of maps (see, e.g., Mirats Tur et al. 2009) or map-building strategies (e.g., simultaneous localization and mapping (Durrant-Whyte and Bailey 2006)), or the use of an external infrastructure (e.g., a communication network or a localization system (O'Hara et al. 2008)).

Many studies in the context of swarm robotics navigation consider a scenario where robots need to move back and forth between two locations, e.g. to transport items from one place to another. Most of this work is based on indirect communication between robots,

C. Pinciroli
e-mail: cpinciro@ulb.ac.be

V. Trianni
e-mail: vtrianni@ulb.ac.be

and is inspired by the foraging behavior of certain types of ants in nature (Werger and Matarić 1996; Wodrich and Bilchev 1997; Sharpe and Webb 1999; Garnier et al. 2007; Fujisawa et al. 2008; Nouyan et al. 2009; Ducatelle et al. 2011a). This behavior relies on *stigmergic communication*, which is a form of *indirect communication* through local modification and sensing of the environment. Specifically, ants moving between the nest and a food source leave a chemical substance, called pheromone, in the environment, which attracts other ants and guides them to the food. The interesting aspect is that the collective process of pheromone laying and following reinforces the most efficient paths, so that eventually the shortest path appears as a consequence of the swarm's collective actions (Deneubourg et al. 1990; Bonabeau et al. 1999). This is an example of emergent self-organized behavior. An important difficulty with the use of this pheromone-based navigation model in robotics is the practical implementation of the indirect communication, in terms of a satisfactory artificial replacement for the chemical pheromone used by ants.

In this work, we propose a new approach for navigation in swarm robotics based on *direct communication* between robots, and fully relying on cooperation and simple interactions. We consider a general problem scenario where a swarm of robots equipped with wireless communication devices needs to execute multiple tasks in a confined area. The tasks correspond to events that need to be serviced in given locations. Each event can be taken care of by one or more robots with the appropriate skills to service the event. For instance, a task can consist in transporting multiple items, one at a time, from one location to the base location of the swarm, or vice versa. Another practical example are fire events, which require robots capable to transport water to move back and forth between multiple water sources and fire locations.

A full solution to this class of problems involves mechanisms for detecting the events and announcing them to the swarm, for the allocation of robots to events, and for guiding robots with the appropriate skills to deal with a specific event to event locations. In this work we focus on *robot navigation*: how can a robot navigate to event's location after the event has been advertised and the robot has assigned itself to the task following the reception of event notification. At this aim, we assume that, for each event in the environment, there is one robot T of the swarm that has detected the event and found its location. The robot remains static at that location, and announces its presence (i.e., the presence of the task) through periodic wireless message broadcasts. We refer to T as a *target* robot. A robot S of the swarm that can service the task, needs to navigate to a given target robot T, which is, in the general case, outside the range of its sensors and communication devices. We investigate how S can find T through cooperative support from the other robots in the swarm when no environment maps or external localization systems are available to the robots. An important aspect in our problem definition is that these other robots can be involved in tasks that are independent of the navigation of S. They *do not adapt their movements to guide* S in its navigation task, but they do offer help through *communication*. This means that the behavior of the remaining robots of the swarm does not depend on the navigation of S to T. In fact, these robots may be involved in any task of their own, including a different navigation task, to another target T', or even to the same target T. In this way, depending on the behavior of the different robots and the nature of the events, a variety of different scenarios of practical interest can be obtained, and the swarm can fully exploit the existing individual capabilities to perform *multiple tasks and swarm navigation in parallel*. In this respect, our scenario significantly differs from the typical ones previously considered in swarm navigation, in which some robots adapt their own behavior (or even stand still playing the role of environment landmarks) to support the navigation of other robots, or, more in general, where all robots are involved in solving a single task cooperatively.

We deal with the described problem scenario proposing an algorithm based on *mobile wireless network communications*. Each robot A coming in communication range of a target robot T, and receiving its periodic broadcasts, stores information about T in a local data structure, which we call a *navigation table*. This information consists of a sequence number, indicating the relative age of the message, and a distance value, which is an estimate of the navigation distance to T. As A moves around, it updates the information in its navigation table, and periodically broadcasts it to neighboring robots. This way, *navigation information* can travel through the (possibly intermittently connected) mobile ad hoc network (MANET) formed among the swarm of robots by being carried on board of the mobile robots. A searching robot S receiving new navigation information from a robot B, compares this new information to previously received navigation information, and moves towards B's location if the new information is better.

This way, navigation information spreads throughout the MANET in a wireless multi-hop fashion, but without requiring to establish routing paths, as is common in the area of *delay-tolerant networking* (DTN) (Fall 2003; Karlsson et al. 2008). On the other hand, a searching robot S makes use of the navigation information to physically move from robot to robot locations towards the target, similar to how a data packet follows a multi-hop route through a MANET (Royer and Toh 1999; Di Caro et al. 2005).

The proposed algorithm is relatively simple, but very powerful and versatile. When applied in different scenarios, it can give rise to different swarm-level movement patterns, while each time providing efficient navigation. Since the given problem description is very general, without losing generality, we restrict our study to two scenarios, which we refer to as *single robot navigation* and *collective navigation*. They are representative of a large number of scenarios of both practical and theoretical interest. The two scenarios have been implemented and studied both in *simulation* and using real robots, the *foot-bots* (Dorigo et al. 2013; Bonani et al. 2010) (see Sect. 3.1). Simulation tests assume robots with the characteristics of the foot-bots.

In the single robot navigation scenario, a single robot S needs to find a single target robot T, which remains static. An example application of this scenario could be that T is indicating a place where a certain task needs to be performed, and only S has the capabilities required for this task. All other robots of the swarm execute random movements, expressing that they are involved in other tasks, which are independent of S's navigation. The goal is to show that using the proposed algorithm, they can offer support to S's navigation without having to adapt their own movements. We investigate the performance of the system with varying swarm sizes, environments, and random movement patterns. We show that the approach is efficient, scalable, and robust to robot failures.

The collective navigation problem is essentially the frequently studied scenario in swarm robotics, where all robots of the swarm navigate back and forth between two targets T and T'. Compared to the single robot navigation problem of the first scenario, we show that collective navigation gives rise to synergies, improving navigation performance. In particular, the concurrent execution of communication-based navigation by all robots lets the swarm *self-organize*, and a collective movement pattern *emerges* in the swarm behavior. This self-organized movement improves navigation efficiency and is robust with respect to the swarm size. Moreover, it allows the robots to find the shortest path in cluttered environments. This means that collective navigation based on our communication-based system has similar properties to ant-inspired pheromone-based navigation, while avoiding the problem of how to implement stigmergic communication. Besides showing a new approach for collective navigation, this is also an example of the general applicability of our simple navigation system.

In terms of *requirements*, our approach only relies on wireless message communication between robots to find paths for navigation, leveraging on simple interactions, cooperation, and self-organization, as is common in swarm robotics. However, to make our message-passing approach feasible, we require some specific properties from the robots' wireless communication device. First of all, the device should provide *line-of-sight* communication, so that communication links can be related to obstacle-free paths. Second, the device should be able to link received messages to *relative position information* (angle and distance) about their sender, so that robots can follow paths detected through communication. Similar requirements were formulated in O'Hara and Balch (2004), where a network of embedded communication nodes is used to guide a single robot to a target. Similar to that work, we address these requirements using an *infrared range-and-bearing* (IrRB) communication system, of which implementations exist for various robots (Pugh and Martinoli 2006; Gutiérrez et al. 2008; Roberts et al. 2009; Bonani et al. 2010), and in particular for the foot-bot robots that we used in the experiments.

The rest of this paper is organized as follows. In Sect. 2, we describe the communication aided navigation algorithm. In Sect. 3, we study the working of this algorithm in the scenario of single robot navigation. In Sect. 4, we investigate the scenario of collective navigation: we study how the system self-organizes, and how it is able to find shortest paths. After that, in Sect. 5 we describe the implementation of our system on real robots, and in Sect. 6 we discuss related work. Some of the work presented here appeared earlier in conference papers (Ducatelle et al. 2009, 2011b).

2 Communication aided navigation

In this section, we explain the communication aided navigation system. We first describe the details of the algorithm executed by the robots. Then, we take a look at the swarm as a whole and explain how the joint execution of the proposed algorithm by the robots can support effective navigation.

2.1 The navigation algorithm

The navigation system we propose is loosely based on routing algorithms used in MANETs. Using wireless communication, the robots of the swarm form a MANET among them. The general idea is to build up navigation information through communication in this MANET, and use it to guide a searching robot from hop to hop to its target, similar to how routing information is gathered in a MANET and used to forward data packets to their destination. All robots in the swarm maintain a table with navigation information about all known target robots. The information about a target T contains an estimate of the navigation distance to T, as well as a sequence number that serves as an indication of the relative age of the information. Each robot periodically broadcasts the content of its table to its neighbors, which update their table based on the received information. This way, navigation information spreads throughout the swarm via wireless communication. Robots also update the distance estimates in their table based on their own movements, using odometry information. This way, navigation information can travel between parts of the MANET which are not connected through wireless communication, by being carried on board of the mobile robots, as is common in the area of delay-tolerant networking (DTNs) (Fall 2003; Karlsson et al. 2008), *without* requiring establishing and maintaining routing paths, which could be problematic in MANETs. This is important to let the algorithm operate both in dense and sparse robot swarms. To navigate to a given target robot T, a searching robot S

continuously monitors all received navigation information. Each time it receives improved navigation information (where the quality of navigation information is defined based on its distance and age, as explained below), it moves towards the neighbor robot that sent this information. This way, S moves between the robots of the swarm until it reaches T. In what follows, we describe the different aspects of this system in detail. An overview of how they tie together is given in Algorithm 1, which shows the sequence of actions executed by a robot in each control step.

Algorithm 1 Communication-based navigation: the actions executed at each control step by each robot A

```
1:  /* Update local distance estimates */
2:  for (Each target $T$ in navigation table) do
3:      Update distance information $d(A, T)$ for $T$ based on $A$'s moved distance
4:  end for
5:  /* Process received messages */
6:  for (Each received message from a neighbor robot $B$) do
7:      for (Each target $T$ in the message) do
8:          Receive distance $d'(B, T)$ and sequence number $s'(T)$ from $B$
9:          Compute $d'(A, T) := d(A, B) + d'(B, T)$
10:         /* Update navigation tables if new information is better */
11:         if $((s'(T) > s(T))$ OR $((s'(T) == s(T))$ AND $(d'(A, T) < d(A, T))))$ then
12:             Replace information for $T$ in table: $s(T) := s'(T)$ and $d(A, T) := d'(A, T)$
13:         end if
14:         /* Update navigation behavior if new information is better */
15:         if ($A$ is searching for target $T$) then
16:             if $((s'(T) > s^*(T))$ OR $((s'(T) == s^*(T))$ AND $(d'(B, T) < d^*(T))))$ then
17:                 Replace current navigation information: $s^*(T) := s'(T)$ and $d^*(T) :=$
                    $d'(B, T)$
18:                 Move towards $B$'s position
19:             end if
20:         end if
21:     end for
22: end for
23: /* Send message */
24: if (Time to send update) then
25:     if (The local robot $A$ is a target) then
26:         Increase sequence number $s(A)$ for target $A$ in navigation table
27:     end if
28:     for (Each target $T$ in (subset of) table) do
29:         Add information $s(T)$ and $d(A, T)$ to message
30:     end for
31:     Broadcast message
32: end if
```

Navigation tables and message broadcasts The navigation information about a target T present in a robot A's navigation table consists of a sequence number $s(T)$, indicating the relative age of the information, and a distance $d(A, T)$, indicating the distance traveled by the information between T and A. Since navigation information can only travel via line-of-sight wireless communication or on board of moving robots, $d(A, T)$ is an estimate for the navigation distance between A and T. At the start of swarm deployment, all robots have an empty table. When a robot T becomes a target robot (i.e., it discovers a target location and starts announcing it), it puts an entry about itself in its table. In this entry, both the sequence number $s(T)$ and the distance $d(T, T)$ are set to 0. At periodic intervals, robots broadcast the content of their table to neighbors. When T broadcasts the information about itself, it first increases sequence number $s(T)$ in its table by 1. The distance $d(T, T)$ is broadcast without modification. Another robot A broadcasting information about T does not modify $s(T)$, so that the sequence number marks the relative time when the information left T. The use of sequence numbers to mark the relative age of messages was inspired by MANET routing protocols such as DSDV (Perkins and Bhagwat 1994). The size of each robot's navigation table, and hence of its update messages, depends only on the number of targets in the environment. If bandwidth is limited, robots select a subset of targets to send updates about, in a round-robin fashion.

Processing received broadcasts Any robot A receiving a broadcast from another robot B processes the entries for all targets T in the message. It reads the received sequence number $s'(T)$ and distance $d'(B, T)$ from the message. On the basis of $d'(B, T)$, it calculates a new estimate for its own distance to T, $d'(A, T)$, by adding the distance $d(A, B)$ between itself and B (as measured at message reception with the IrRB communication system). Then, A compares the new values, $s'(T)$ and $d'(A, T)$, to the information about T in its own table, $s(T)$ and $d(A, T)$. The new information is considered better if either $s'(T) > s(T)$ (the new information is more recent), or $s'(T) = s(T)$ and $d'(A, T) < d(A, T)$ (the new information is equally recent, but indicates a shorter path). In that case, the information in the table is replaced by the new information.

Updating distance estimates If A moves around without receiving new updates about T, the distance $d(A, T)$ in its table needs to be updated for it to remain an estimate of the navigation distance to T. Therefore, as A is moving, it measures its moved distance through odometry, and adds this to $d(A, T)$. This way, $d(A, T)$ grows and remains a measure of the distance traveled by the navigation information. The direction of A's movement is not taken into account, so that $d(A, T)$ is not necessarily the shortest distance to T. However, it is an upper bound of the shortest obstacle-free path (since A per definition moved over an obstacle-free path). Using this mechanism, the navigation system can work in sparsely connected swarms: navigation information can bridge gaps in network connectivity by traveling on board of moving robots.

Using the received messages for navigation A *searching* robot S moves towards the location of the neighbor from which it receives the best navigation information about its target T. The information $s(T)$ and $d(A, T)$, received from a neighbor A, is considered better than the information $s'(T)$ and $d'(B, T)$, received from a neighbor B, if $s(T) > s'(T)$ (A's information is more recent), or if $s(T) = s'(T)$ and $d(A, T) < d'(B, T)$ (A's navigation distance to T is less than B's). In case A's information is the best, S stores $s(T)$ and $d(A, T)$ as $s^*(T)$ and $d^*(T)$, respectively, and also A's relative location L_A, as measured by the IrRB system at the moment of message reception. Then S moves towards L_A using odometry. Note that S does not adapt its goal in case A moves: only A's location L_A at the moment of reception of the navigation information is important. Any newly received navigation information (either from A again, or from another neighbor) is compared to $s^*(T)$ and $d^*(T)$.

If the information received from a neighbor C is better, S moves towards C's location L_C. This can happen either before S had reached its previous goal L_A, or after that. In the former case, S just abandons its previous goal in favor of the new one. In the latter case, S is faced with a period in which it has no direction to go to (between the arrival at L_A and the reception of the new information). In this case, we consider two possible strategies: S can either wait statically at L_A, or start performing random movements until new information is received. We refer to the former strategy as *navigation with stopping* (NwS), and to the latter as *navigation with random* (NwR); we compare the two strategies in Sect. 3. The repeated moves let S follow the best navigation information through the network. When S eventually receives a message directly from T, it goes straight to T and finishes the search. Finally, we point out that we let the searching robot S approach any location (be it that of another robot A or of the target T) from the right (by aiming for a location slightly to the right of L_A). This is to avoid head-on collisions between robots (especially useful when two searchers move towards each other, as in the scenario of Sect. 4).

2.2 The system's dynamics

The proposed navigation algorithm lets a searching robot S move towards the location of neighbors that have information about its target T that is better than what S had previously received, where "better" information means either more recent information (higher sequence number), or information that has traveled over a shorter path from T (lower estimated distance). Here, we discuss how such moves can bring S closer to T.

The issue is relatively straightforward in scenarios where robot density is high and the swarm forms a connected MANET including S and T. In this case, the periodic local broadcasting of messages by the robots of the swarm lets each new message from T (each new sequence number) flood the MANET. Flooding spreads as an expanding ring from T, and new navigation information reaches S first over the shortest path through the network. Such flooding mechanisms are the same as those used by reactive MANET routing algorithms to define the shortest path for data forwarding (see, e.g. Perkins and Royer 1999). Hence, when S moves towards the most recent navigation information, it follows the shortest path available for data routing in the MANET. Since T is continuously sending new messages (with increasing sequence numbers), the path followed by S is constantly adapted to changes in the MANET topology. The correspondence between the shortest path for data routing and the shortest path for navigation depends on the density and spread of robots in the environment (see examples in Fig. 1).

When we consider scenarios where the robot distribution is sparser, the MANET formed among the swarm may no longer be connected. At this point, a new message sent out by T does not immediately flood throughout the swarm: to reach disconnected parts of the MANET, a message needs to be carried there by mobile robots. This means that message spreading depends on a combination of robot mobility and message communication. Several studies investigated message spreading in sparsely connected MANETs (Spyropoulos et al. 2004; Groenevelt et al. 2005; Zhang et al. 2007; Jacquet et al. 2010; Klein et al. 2010). In case robot density is not extremely sparse, so that robots can communicate with others relatively frequently, new messages spread from T in an expanding wave-like propagation (Jacquet et al. 2010; Klein et al. 2010). Such propagation is similar to the form of spreading obtained through flooding (but slower, as part of the spreading is based on robots carrying the message away from T). As a consequence, if S goes towards the most recent information (or the information that has traveled the shortest distance), it moves into the direction of the expanding wave, and can therefore be expected to make steps in T's direction.

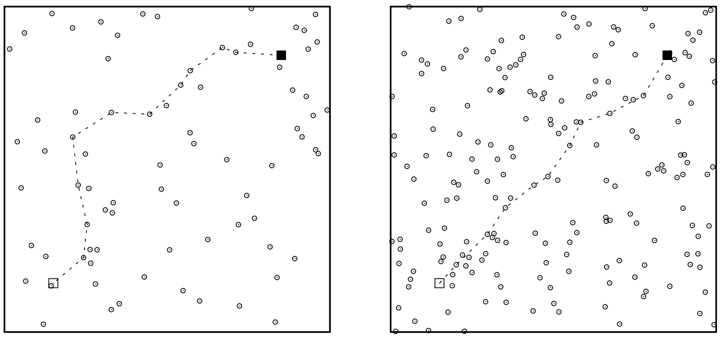


Fig. 1 Shortest communication path between two robots in a MANET. The area is 20×20 m^2, and the communication range is 3 m. The searcher is placed at the *bottom left* and the target at the *top right*. The correspondence with the shortest path for navigation depends on robot placement and density: we show an example with 80 robots (*left*) and one with 200 robots (*right*)

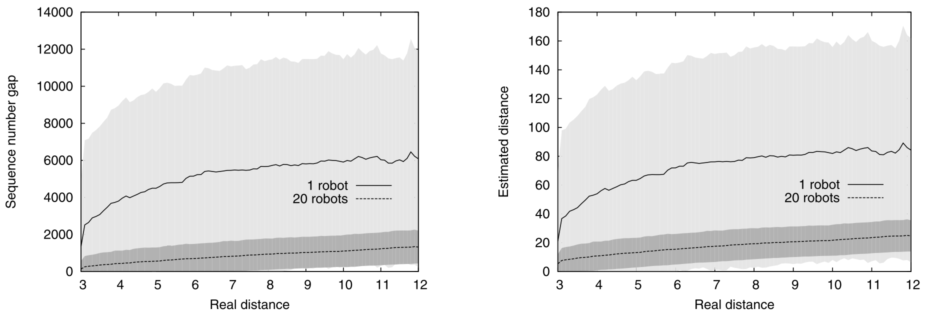


Fig. 2 Navigation information (y-axis) against the distance from the target (x-axis): *on the left* the sequence number gap and *on the right* the estimated distance. We plot data for the case of one robot and 20 robots. The *shaded areas* around the curves indicate the standard deviation. See main text for explanation

In the case of very sparse swarms, robots only occasionally meet each other. In this situation, robot mobility is the main factor defining information spreading: each robot A that meets T picks up a new message and carries it around the environment. If A does not meet any other robot, its sequence number $s(T)$ and distance estimate $d(A,T)$ are defined by, respectively, the time when A met T, and the total length of the movements made by A since then. When S meets A, it moves towards A if A's navigation information is better than what S has received before. Whether this effectively brings S closer to T depends on the relationship between the time/distance that A has traveled from T, and its real distance to T. This obviously depends on the movement patterns followed by A. Nevertheless, several studies in the MANET literature have shown that in general, for most reasonable mobility patterns, there is a positive correlation between the travel time/distance and the actual distance (Dubois-Ferriere et al. 2003; Spyropoulos et al. 2008). This positive correlation has been used to support message forwarding, e.g., based on node encounter histories (Dubois-Ferriere et al. 2003; Grossglauser and Vetterli 2006).

To investigate more in detail the properties of this correlation and its dependence on the number of robots in the network, we performed simulation tests considering both one and multiple moving robots (the specific characteristics of the robot models and of the simulation environment are discussed in the next section). In the first set of experiments, we placed a target robot T in the middle of an uncluttered environment of 20×20 m^2, and let a single other robot A move according to a *random direction mobility* model (see Sect. 3 for details about the simulator and the mobility model). The robots have a communication range of 3 m. We did 10 tests of 10000 s each. At each time step of 0.1 s, we measured the difference between the sequence number on board of A and the most recent sequence number sent out by T. We call this the *sequence number gap*. It is the relative age of the information on board of A, and measures the elapsed time since A last encountered T. We also measured at each time step the real distance between A and T. In Fig. 2, we plot the average sequence number gap against the real distance. The graph shows that the sequence number gap is on average an increasing function of the distance: when A has a lower sequence number gap, it has a higher probability of being closer to T. This means that if a searching robot S moves towards a robot announcing a newer sequence number, it will, in expected value, move closer to the target. However, it must be noted that the curve in Fig. 2 levels out at high distances from T; also, it has a large standard deviation. This means that the information is quite unreliable: many of S's moves will still go in a wrong direction.

The situation improves dramatically when we increase the swarm size. We performed the same tests with 20 randomly moving robots. In this case, we get a density of one robot per 20 m^2. Even though each robot can communicate over an area of more than 28 m^2, this still corresponds to a relatively sparse network connectivity, in which a MANET with randomly placed nodes normally does not provide end-to-end communication connectivity. This is due to the random locations of the robots; see, e.g., Dousse et al. (2002) for a thorough study of the relation between density and connectivity in a MANET. We get therefore in the earlier described situation where the swarm information spreads both through mobility and communication: the robots update each other's sequence number when they meet, and new sequence numbers spread faster through the area, according to a wave-like propagation. This makes the information much more reliable. As shown in Fig. 2, the average sequence number gap decreases, and we get a much smoother relation between the sequence number gap and the distance to T, with lower standard deviation. In tests with higher numbers of robots (not shown here), both the average and the standard deviation of the sequence number gap further decrease.

We also performed these same experiments using the estimated navigation distance $d(A, T)$, rather than the sequence number. This gives very similar behavior, as shown in Fig. 2. This means that both parts of the navigation information, the sequence number and the estimated navigation distance, are positively correlated to the actual distance to the target, and are therefore both useful navigation measures. In our algorithm we use the sequence number and the estimated distance in combination, because this gives the best results. One could, however, also use them separately, e.g., to get a simpler system, which uses less communication bandwidth.

3 Single robot navigation

In this section, we study the single robot navigation scenario. As explained in Sect. 2, the scenario consists of a robot S searching for a static target robot T. All other robots of the swarm are involved in tasks of their own, and perform movements that are unrelated to the navigation of S. To obtain such independent movements, we use random mobility patterns.

We investigate the performance of the communication-based navigation system under varying conditions, using experiments performed in simulation. In what follows, we first

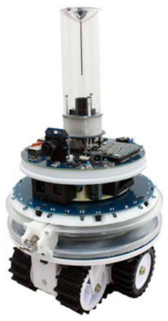


Fig. 3 The foot-bot robot developed within the Swarmanoid project

describe the simulator and the robots we used in these experiments. After that, we study the system in an uncluttered environment, to show its basic working. Next, we investigate the influence of the movement patterns of the robots of the swarm, performing tests with varying mobility models. Then, we study cluttered environments, and show that the system can work even in highly complex environments, such as mazes. Finally, we investigate situations where two paths of different length are available, and show that our algorithm has a preference for the shortest path.

As *performance metrics*, we consider the average navigation time of the servicing robot S moving between target locations. In order to have a baseline reference of how good this time is, in the results we also plot the navigation time of a robot following the *shortest path* between the targets.[1]

All throughout this section, we show that our communication-based algorithm lets the swarm support navigation in a fully autonomous way, without relying on external information or infrastructure, and using only very simple interactions and capabilities. Moreover, the robots of the swarm can support the searching robot's navigation without the need to adapt their own movements. This allows a lot of freedom in possible applications of this approach.

3.1 The robots and the simulator

All tests presented in this and in the next section are executed using a simulated model of the *foot-bot*, a small ground robot developed within the Swarmanoid project (Dorigo et al. 2013) (http://www.swarmanoid.org) on the basis of the *marXbot* platform (Bonani et al. 2010). The tests with real robots, presented in Sect. 5, use this same robot.

The foot-bot is shown in Fig. 3. It has a diameter of about 17 cm and it is about 29 cm high. It moves on the ground using a combination of tracks and wheels, for increased stability. It is quite a powerful robot, carrying various sensors and actuators, including two cameras, a rotating distance scanner, a gripper, etc. For the work presented here, two of these are particularly relevant: the infrared proximity sensors, and the IrRB module. The proximity sensors detect obstacles at a range of a few centimeters. We use them as virtual bumpers, to let robots turn away from nearby obstacles. The IrRB module (Roberts et al. 2009; Bonani et al. 2010) provides local line-of-sight communication. It sends messages of 10 bytes, and has a capacity of 10 messages per second (so robots can broadcast an update

[1] Another reference could be the performance of a randomly moving robot. In many of the experiments this is also shown, since this corresponds to the case where the servicing robot S is the only robot in the swarm.

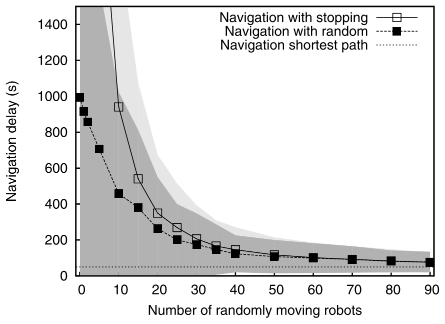


Fig. 4 Test results in an uncluttered environment for a single searching robot and an increasing number of randomly moving robots. In the figure, the results for NwS and NwR (the two different strategies used to deal with the temporary absence of navigation information) are reported, together with the performance that would be obtained by following the shortest path. The *shaded areas* around the curves indicate the standard deviation

every 0.1 s). Its maximum range can be of more than 5 m, but was limited to 3 m here, in order to be able to do tests in smaller environments.

As simulation tool, we use *ARGoS* (Pinciroli et al. 2012), a physics-based simulator for heterogeneous multi-robot systems. Being developed within the Swarmanoid project, ARGoS contains reliable physics models of this robot. It also comes with a middleware for controlling the real robots, so that any code written for the simulator can be ported unchanged to the robots.

3.2 Tests in an uncluttered environment

We use an uncluttered closed area of 20×20 m^2. The robots are placed in the area according to a uniform random distribution. One of the robots is a target and remains static. A second robot needs to navigate to this target. The remaining robots move according to a *random direction mobility model with fixed speed* (Bettstetter 2001). This model is defined as follows: choose a direction θ uniformly from $]-\pi,\pi]$, turn towards θ, choose a time t from an exponential distribution with fixed average (set to 10 s here), move forward for this time t, and then repeat this process. We use a forward speed of 0.15 m/s, both for the searching and the randomly moving robots. We vary the number of robots in the swarm, from two (no randomly moving robots) up to 92 (90 randomly moving), which corresponds to an average robot density ranging from 0.05 to 0.23 robots/m^2. In turn, considering that the communication radius is 3 m, these settings correspond to networks with an average node degree ranging from 0.07 to 6.4 (the average node degree is computed as $\pi r^2((N-1)/A)$, where r is the communication radius, N is the number of robots, and A is the total area). For each data point, we make 500 independent test runs (this high number is needed because the random initial positions of searcher and target induce a high variance). We measure the time between the start of each test and the moment the searching robot comes in range of the target.

The results are shown in Fig. 4. We compare the two variants of the navigation system presented in Sect. 2, navigation with stopping (NwS) and navigation with random (NwR), which differ in the strategy used by the searching robot when it does not have any navigation information (respectively, waiting for new information, or performing a random movement according to the random direction model). The results show a large difference in performance between the two strategies for low numbers of robots. This is because the communication network is sparse, and navigation information spreads slowly from the target, so that the searcher often falls without information. In the extreme case with no randomly moving robots, navigation with stopping can never reach the target. Navigation with random, on the other hand, does find the target, through random search. The expected time for a randomly moving robot to find a static target within a given environment is normally referred to as the expected *hitting time*, ET (Spyropoulos et al. 2006). For many mobility models, including the random direction model used here, ET can be calculated analytically (Spyropoulos et al. 2006). In our case, ET can be considered an upper bound for the performance of the navigation with random strategy. It is interesting to note that even a very low number of randomly moving robots in the environment gives an improvement in the navigation delay compared to ET. This confirms that even in very sparse swarms, the navigation information on board of randomly moving robots can be useful to guide the searcher, as explained in Sect. 2.2.

For larger swarm sizes, performance improves for both strategies. This is on the one hand because the improved connectivity in the swarm makes the navigation information more reliable, as pointed out in Sect. 2.2, and on the other hand because information reaches the searcher more frequently. The latter also means that the searcher finds itself less often without navigation information, so that the difference between the two strategies decreases. For the highest numbers of robots, performance gets close to the time needed to cross the straight line distance between the searcher's initial position and the target. This is indicated in Fig. 4 as "Navigation shortest path".[2] This gives a lower bound for the expected navigation time. The good performance for large swarm sizes shows both the *efficiency* and *scalability* of the system. In additional simulation experiments (not reported here) we could verify that the navigation delay is relatively stable for swarm sizes up to 500 robots. It is also interesting to note the gradual degradation of the system's performance as the number of robots decreases. This indicates that the navigation system does not rely on a specific minimum number of robots.

Finally, we discuss the standard deviation of the navigation delay, as shown in Fig. 4 by the shaded areas around the curves. This is quite large, due to the fact that the searcher and target robots are placed randomly in the arena: the randomness of the start locations contributes to the variations in delay. We opted for this approach in our tests, in order to avoid that a specifically chosen target location could have an unwanted influence on the results. We also point out that at each point along the curves of Fig. 4, the reported standard deviation is more or less equally large as the reported average. This is in line with theoretical results for the propagation delay of communication messages in mobile delay-tolerant networks under similar circumstances, which is usually exponentially distributed (see Sect. 2.2 and Spyropoulos et al. 2008). Very similar observations were made for all other tests in this paper that use randomly chosen searcher and target locations: the standard deviation is always close to the mean. We will in the rest of this text therefore only report standard deviations for tests with different setups.

3.3 Tests with different movement patterns

The performance of our system depends on the movement patterns of the robots of the swarm: this defines for a large part how and where navigation information spreads. Here we

[2] Given the random locations of searcher and target robots in the experiments, we show the delay to navigate the expected length of the shortest path.

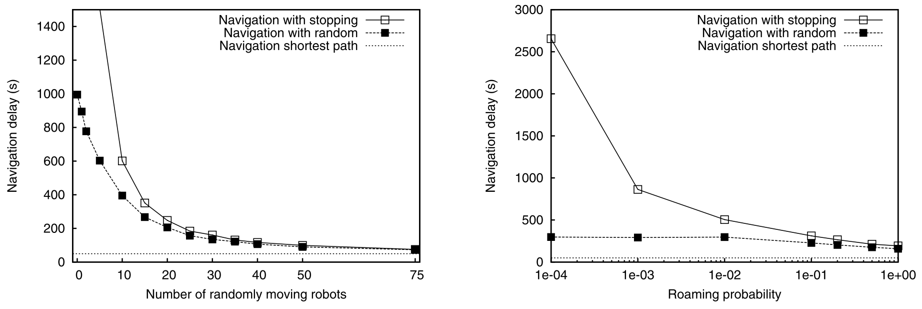


Fig. 5 Test results with different random mobility models for a single searching robot. The *left figure* refers to the case when the other robots in the swarm move according to the random waypoint model, and shows the results for an increasing number of robots. The *right figure* refers to the case when the other robots move according to the restricted random waypoint model. In this case, a swarm of 25 robots is considered and the results show the impact of using different roaming probabilities. In both figures, the results for NwS and NwR are reported together with the performance that would be obtained by following the shortest path

carry out experiments in the same uncluttered environment used in Sect. 3.2, using different mobility models. We use the random waypoint model (RWP) (Johnson and Maltz 1996) and the restricted random waypoint model (RRWP) (Blažević et al. 2002).

Under RWP, each robot randomly chooses a location in the environment to move to, and chooses a speed. It moves to the chosen destination with the chosen speed, and then waits there for a fixed pause time, before choosing a new destination and speed. RWP has very different statistical properties compared to the earlier used random direction model (Bettstetter et al. 2004; Nain et al. 2005). E.g., it lets robots make longer straight movements (since robots can choose any location in the area to move to), it leads to a non-uniform stationary distribution of robots over the area, etc. We use RWP with a fixed speed of 0.15 m/s and a pause time of 10 s. We vary the swarm size from 2 up to 75. The results are shown in Fig. 5 left. They are very similar to the ones obtained with random direction movement in Sect. 3.2, showing that the differences between the mobility models has a limited impact on the performance of the navigation algorithm.

RRWP is a variation of RWP, in which robots can choose their destinations only from predefined destination areas in the environment. A fixed *roaming probability* p defines whether a robot picks its new destination from its current destination area or from a different one (roaming). To define the destination areas, we overlay the environment with a grid of 3×3 cells, where each cell is a different destination area (so, in our experiments, each point in the environment is part of exactly one destination area). We vary p between 10^{-4} and 1. For low values of p, robots remain mainly within their cell, so that we get almost exclusively local robot movements. In this case, navigation information rarely spreads between cells by being carried on board of robots, but rather through communication between robots near the cell boundaries. This has as a side effect that if one or more cells fall without robots, information may not be able to travel between target and searcher for long periods of time. Instead, for high values of p, robots are forced to leave their cell often, so that they execute long movements through the environment and bring navigation information around quickly. We believe that the different movement patterns obtained this way cover a large variety of possible behaviors of the robots of the swarm. For example, the different cells may represent different parts of a factory, where robots perform mainly local movements around assigned work stations, or they could refer to different areas in a warehouse, where robots perform long range movements to bring goods around. We use a swarm with 25 randomly moving

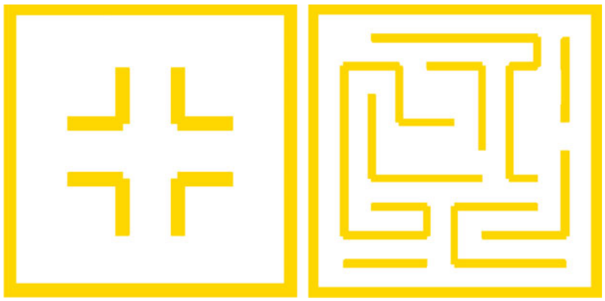


Fig. 6 Layout of cluttered environments used in our experiments. *Left*: a simple environment; *right*: a maze. The area is 20×20 m^2 in both cases

robots, which is a relatively sparse setup, in which the MANET is normally not connected. The results are shown in Fig. 5. The very bad results for the navigation with stopping strategy at low values of p are due to the earlier mentioned effect that cells can fall without robots for a long time, effectively stopping the spreading of navigation information. However, for the navigation with random strategy, these negative effects are rather limited. For larger values of p, we note that the higher mobility of robots improves the performance of both algorithms, though, again, the effect is limited for the navigation with random strategy. We can conclude from these results that if the situation is such that information can flow from target to searcher between the robots, the actual movement patterns of the intermediate nodes does not matter much.

In the following, all tests are executed with the randomly moving robots following the random direction mobility model (see Sect. 3.2).

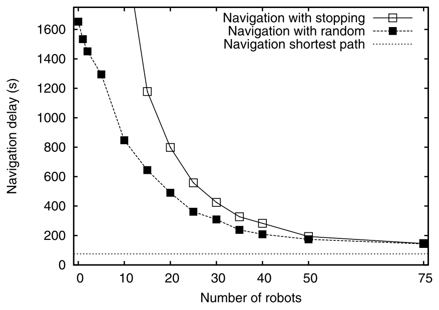
3.4 Tests in cluttered environments

Since our navigation algorithm looks for obstacle-free paths (see Sect. 2), it deals naturally with cluttered environments. We did experiments in the two environments shown in Fig. 6. Again, we deploy the swarm according to a uniform random distribution, and we measure the time needed for the searcher to reach the target. The results are shown in Fig. 7. As can be expected, navigation delays get higher as the environment gets more complex, with the highest values measured in the maze. Also, a larger swarm is needed to bring this delay down, and the system has more difficulties to reach the time required to travel over the shortest path. Nevertheless, we get the same trends in performance as in uncluttered environments, and with a large enough swarm, the system guides a searching robot to its target efficiently.

We point out that our navigation system can also deal with dynamic obstacles. We do not report results here, due to lack of space, but it is clear that a reactive approach such as the one presented here has advantages in dynamic environments compared to, e.g., map-based navigation systems.

3.5 Shortest path

In cluttered environments, a searcher may have several possible paths available to move to its target. Since our algorithm lets a searcher move in the direction from where it receives the navigation information that has traveled the shortest time or the shortest distance (see Sect. 2.2), it should have a preference for the shortest path. We consider the scenario of Fig. 8 to test this property. The target is placed in the upper part of the arena, and the searcher in the bottom part. There are two paths between them: a long one of 24 m, and a short one of



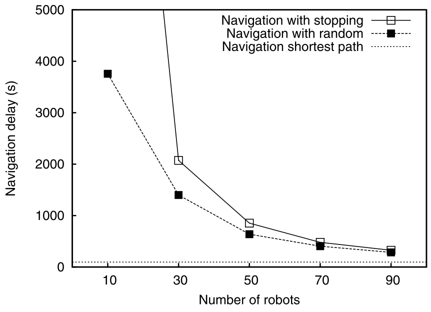
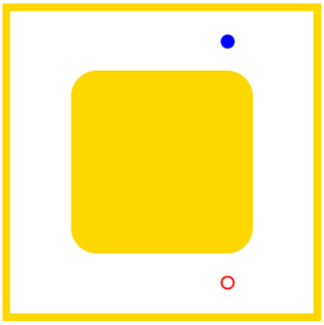


Fig. 7 Test results in the cluttered environments of Fig. 6 for a single searching robot and an increasing number of randomly moving robots. The *left plot* refers to Fig. 6 (left), and the *right one* to Fig. 6 (right). In both figures, the results for NwS and NwR are reported together with the performance that would be obtained by following the shortest path

Fig. 8 Test setup for shortest path testing. The searcher starts from the bottom (indicated by the *red circle*). The target is placed above (indicated by the *blue disk*). The area is 14 × 14 m^2 (Color figure online)



12 m. We do tests with increasing swarm sizes, from three robots (1 searcher, 1 target and 1 randomly moving robot) up to 72 (70 randomly moving robots). The results are shown in Fig. 9. We show how often the searcher chooses the short path (as a fraction of the total number of tests), and we show the time needed for navigation. The results show that the navigation algorithm has a clear preference for the shortest path. Also, this preference leads to lower navigation delays (for the NwR strategy, we plotted the navigation delay separately for the tests in which, respectively, the short or the long path was chosen).

One striking element in these results is that the probability of choosing the short path is related to the swarm size, and that this relationship is different for the two navigation strategies. To explain this, let us first look at the results for the largest and smallest swarm sizes. In the scenarios with largest swarm size (70 randomly moving robots), navigation information travels primarily through multi-hop message forwarding between robots. The swarm is well connected, and navigation information travels equally quickly in all directions from the target T. Since the distance to be covered is less over the short path, the information reaches the searcher S faster this way, letting S prefer the short path. Since S rarely finds itself without navigation information, the behavior and performance are identical for the two navigation strategies.

The situation is very different for the smallest swarm size (1 randomly moving robot). Here, navigation information only travels by being carried on board of the single randomly moving robot A. Under the navigation with random strategy, the influence of A is rather limited, and S finds T mainly through random search. Under the navigation with stopping

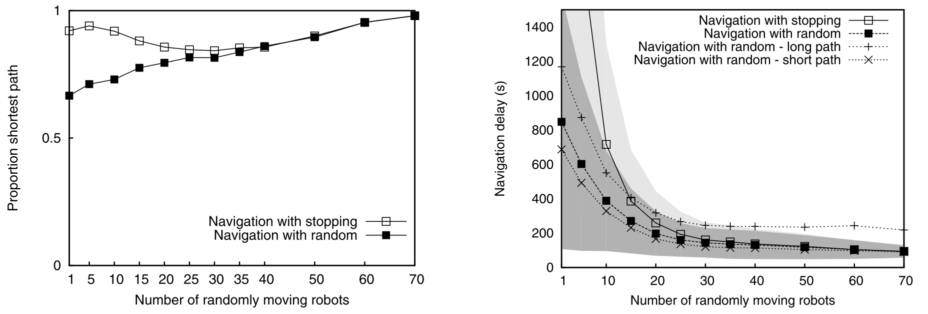


Fig. 9 Test results in the cluttered environment of Fig. 8 for a single searching robot and an increasing number of randomly moving robots. For both the NwS and the NwR strategies, we show the fraction of runs in which the searcher uses the short path (*left*), and the average time needed for navigation (*right*). *Shaded areas* around the average navigation delay curves show the standard deviation (compared to other tests, standard deviations are lower here, because searcher and target do not start from random positions; see Sect. 3.2). For the NwR strategy, we also show the average navigation delay when splitting up the results according to the path taken by the robots

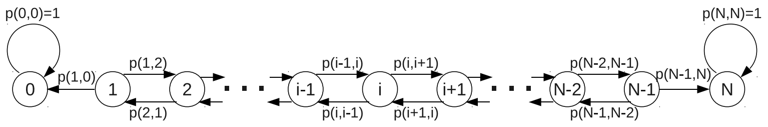


Fig. 10 State space representing the movement of the searching robot

strategy, on the other hand, S moves only when A brings it a new sequence number. This means that each time A moves from T to S, S makes a step towards T, where the step size depends on the communication range. Whether this step is towards the short or the long path, depends on which path was used by A to reach S. To analyze this behavior, we model it as a random walk in a one-dimensional, discrete state space, as shown in Fig. 10: S starts in an initial state i, and moves in discrete steps either to the left or to the right. The walk ends when S reaches either state 0 (which means S reached T over the long path) or state N (S reached T over the short path). Since we use a communication range of 3 m, we set $N = \frac{24+12}{3} = 12$ and $i = \frac{24}{3} = 8$. The model shown in Fig. 10 corresponds to a well-known problem in probability theory, called the gambler's ruin problem (El-Shehawey 2009). The probability for an agent starting in i to end up in N, rather than in 0, is known to be

$$P_N(i) = \frac{1 + \sum_{m=2}^{i} \prod_{k=1}^{m-1} \frac{p(k,k-1)}{p(k,k+1)}}{1 + \sum_{m=2}^{N} \prod_{k=1}^{m-1} \frac{p(k,k-1)}{p(k,k+1)}}. \tag{1}$$

We first use this formula to model the behavior of the randomly moving robot A. In this case, the transition probabilities between states are all equal $p(i, i+1) = p(i, i-1) = 0.5$, and Eq. (1) simplifies to $P_N(i) = \frac{i}{N}$: the probability of choosing the short path depends linearly on the difference in path length. This behavior of A can be compared to the movement of S in the navigation with random strategy (since S moves mainly randomly), where the fraction of runs using the short path is 0.67 (a very close fit, given that $\frac{i}{N} = \frac{8}{12}$). For the navigation with stopping strategy, the transition probabilities depend precisely on the probability of the randomly moving robot A to reach S either over the short or the long path, so that $p(i, i+1) = \frac{i}{N}$ and $p(i, i-1) = 1 - \frac{i}{N}$. Plugging this in Eq. (1), we get $P_N(i) = 0.89$, which is very close to the observed performance of 0.92.

Scenarios with intermediate swarm sizes fall in between these two extremes. We make a distinction between intermediate large swarm sizes (40–60 randomly moving robots) and intermediate small swarm sizes (5–35 randomly moving robots). For intermediate large swarm sizes, the performance of the two navigation strategies is identical. This means that S rarely finds itself without navigation information, which is an indication that there is usually a connected route between S and T in the MANET, over which information flows continuously. However, due to the lower robot density compared to the largest swarm size, network connectivity may be less than perfect. As a consequence, the connected communication route sometimes only exists over one of the two navigation paths, and S may occasionally be attracted towards the long path. For intermediate small swarm sizes, the performance differs between the two navigation strategies, indicating that S regularly finds itself without navigation information. This is because at low densities, a MANET falls apart into smaller connected clusters (Dousse et al. 2002), such that information cannot flow continuously. However, compared to the case of very small swarm sizes (e.g., the case with only one randomly moving robot), the presence of connected clusters has an important consequence. It means that whenever S meets a robot with navigation information, it immediately also finds a number of other robots with similar information, so that it moves longer into the same direction before finding itself again without information. In the context of the state space shown in Fig. 10, this could roughly be modeled by using less states (because each step of S in a given direction will normally go on for longer than the communication range). E.g., if we assume a step size of 6 m, we could use the same model with $N = 6$ and $i = 4$, while keeping the same transition probabilities of $p(i, i+1) = \frac{i}{N}$. This gives a result of $P_N(i) = 0.81$. This preference for the short path is lower than in the case of the navigation with stopping strategy with only 1 randomly moving robot (0.89), but higher than the case of the navigation with random strategy with only one randomly moving robot (0.67). This explains why navigation with random always improves the preference for the short path with increasing swarm sizes, while navigation with stopping first decreases this preference, and only later increases it (when end-to-end connected routes appear).

4 Collective navigation

In the collective navigation problem, *all* robots of the swarm navigate between target event locations. For the experiments, we consider a basic scenario in which all robots navigate back and forth between two targets present in the environment, T and T'. As pointed out in Sect. 1, this is a common task in swarm robotics. To follow swarm terminology, we refer to the two target locations as *nest and food source*. In this case, we report results only for the "navigation with random" strategy, as this gives the best performance. Our goal is to show that our communication-based navigation algorithm can also be used in this scenario. However, the observed performance and properties of robot navigation are different compared to the single robot navigation scenario, due to the specific characteristics of the collective navigation scenario. In particular, the collective execution of the same behavior by all robots lets the swarm *self-organize*, such that coordinated behavior emerges from the local interactions between the individual robots. This self-organization improves the performance and efficiency of navigation.

In what follows, we first investigate the behavior of the system in uncluttered environments, to study the self-organized behavior of the swarm. After that, we study the same

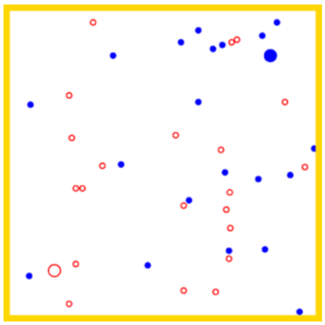


Fig. 11 Setup for collective navigation experiments. The area is 20×20 m^2. The target robots (food and nest) are located in the *top-right corner* (food robot, indicated with a *big blue disk*) and in the *bottom-left corner* (nest robot, indicated with a *big red circle*). Initially, half of the robots (indicated with *circles*) go to the nest source, the other half (indicated with *disks*) go to the food source (Color figure online)

behavior in cluttered environments. Finally, we investigate what happens when two paths of different lengths are available between nest and food source: we show that the self-organized behavior lets the swarm select one of the two paths, with strong preference for the shortest.

As *performance metrics*, we use the average navigation time of the robots moving back and forth between target locations. In addition to this basic metric, we also consider the total "service" performed by the swarm, measured by the number of times a robot reaches a target location, and the amount of self-organized cooperation within the swarm, measured through hierarchic social entropy (Balch 2000). Moreover, we measure to which extent the navigation paths followed by the robots are equivalent to the shortest ones.

4.1 Self-organized behavior in an uncluttered environment

We first use the setup shown in Fig. 11. Two robots, indicating the nest and food source, are placed in opposite corners of the arena, at a distance of about 20 m. All other robots are placed according to a uniform random distribution. Half of these robots initially go to the food source, the other half to the nest. A robot that has reached its target (i.e., food source or nest) starts moving towards the other target. A robot is said to have reached a target when it comes within 0.5 m of it. We vary the total number of searching robots in the swarm from two up to 60. We perform 50 independent test runs of 5000 s for each setup. We measure the average time needed by robots to move from one target to the other. We compare to experiments with the same numbers of robots, but where only one robot is going back and forth between nest and food source, while the other robots of the swarm are moving according to the random direction mobility model (as in the single robot navigation experiments of Sect. 3).

The results of these tests are shown in Fig. 12. For both single robot navigation and collective navigation, performance improves as the number of robots increases, since navigation information spreads more easily in densely connected swarms (see Sect. 3). However, for the collective navigation scenario, the performance improves faster (with 30 robots, navigation delay of collective navigation is about half of that of single robot navigation). Also, the standard deviation of the navigation delay drops faster for the collective navigation scenario, indicating a more stable performance. This is due to cooperation. When a robot moving towards the food source (and hence coming from the nest) and a robot navigating towards the nest meet, they can give each other navigation information about their respective targets.

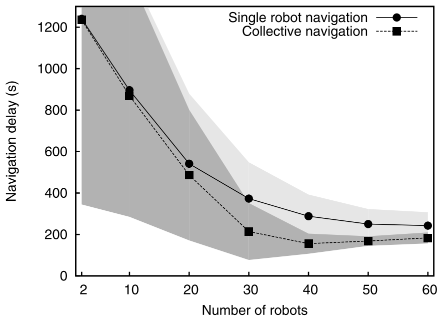
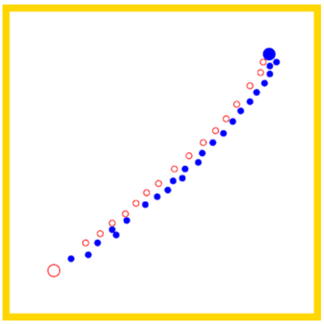


Fig. 12 Test results for collective navigation in the uncluttered scenario of Fig. 11. The figure reports the observed navigation delays (average and standard deviation) for an increasing number of robots, for the case in which all robots in the swarm go back and forth between nest and food source (collective navigation), and for the case in which only one robot is going back and forth between nest and food while the other robots of the swarm move according to the random direction mobility model (single robot navigation)

Fig. 13 Collective navigation after 300 s of simulation: a self-organized dynamic chain has formed, with part of the robots going to the food source (*red circles*) and the others going to the nest source (*blue disks*) (Color figure online)



Moreover, if a group of robots moving towards the same target are in communication range from each other, new information received by any of them spreads throughout the whole group, and they simultaneously move in the same direction. These two effects make robots form clusters moving in opposite directions. When there are enough robots, such clusters can become large enough to cover the whole distance between nest and food source. At that point, the swarm organizes into a stable structure, which we refer to as a *dynamic chain*. Figure 13 illustrates this behavior for a typical run of collective navigation with 40 robots. It is this behavior which causes the strong improvement in performance between 20 and 30 robots in Fig. 12. For larger swarms (50 and 60 robots), congestion of robots near the target locations leads to a slight decrease in performance.

The dynamic chain is an example of *emergent self-organized behavior*: the swarm shows organization at the global level that emerges from local interactions between individual robots. In what follows, we investigate when this self-organization arises and how stable it is. To do this, we first need a measure for self-organization. Several authors use *entropy* to measure self-organization in the context of swarm robotics (Baldassarre 2008; Sperati et al. 2011). If X is a random variable which can take M different states, its entropy $H(X)$ is defined as

$$H(X) = -\sum_{i=1}^{M} p_i \log_2(p_i), \tag{2}$$

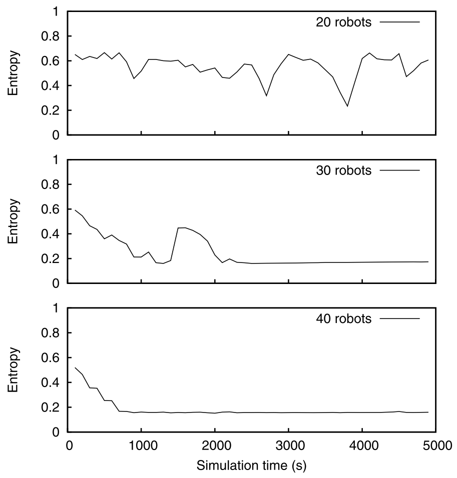


Fig. 14 Evolution of the hierarchic social entropy $S(R)$ over the course of an example test run for 20, 30, and 40 robots

where p_i is the probability that X is in state i (here, we refer to Shannon's information entropy (Shannon 1949)). Strictly speaking, this is a measure of order (or disorder), rather than self-organization: the more a system is ordered, the more you can find it in a limited subset of its possible states, and the lower the entropy. In principle, self-organization is more than just an increase in order, and different measures for self-organization have been proposed (Shalizi et al. 2004). For us, however, it is sufficient to measure whether there is increased order in the behavior of the robots, so we stick with entropy.

To calculate the entropy $H(X)$, we need a discrete variable X that characterizes the swarm behavior. In Baldassarre et al. (2004), Sperati et al. (2011) the authors use the orientation of the robots, discretized into four bins; the entropy based on this variable indicates to what extent the robots move in the same direction. In our case, this measure can be used (once the chain is formed, robots move in similar directions), but it is quite noisy, especially when there is congestion (robots turn to avoid each other). What we really want to measure is whether the robots move in a low number of connected clusters; that is, whether there is order in their physical locations. To do this, we turn to *hierarchic social entropy* (Balch 2000), which proposes an entropy measure for a group of robots characterized by a multi-dimensional variable. In our case, this multi-dimensional variable will be the location coordinates of each robot. The idea behind hierarchic social entropy is to first cluster the robots using hierarchic clustering based on a distance threshold h: a robot is added to a cluster if it is within distance h of all the robots in the cluster. The division of robots into clusters gives a discrete variable X on the basis of which entropy is calculated (the clusters form the M different states for X, and the probabilities p_i are defined by the number of robots in each cluster). Obviously, X depends on the threshold h: if $h = 0$, each robot is in a cluster of its own, and entropy is maximal, while if $h = \infty$, all robots fall in a single cluster, and entropy is 0. Therefore, the notation $H(R, h)$ is used to refer to the entropy of a group of robots R using clustering distance h. The hierarchic social entropy $S(R)$ is then defined by integrating $H(R, h)$ over all values of h:

$$S(R) = \int_0^\infty H(R, h)\, dh. \tag{3}$$

We use $S(R)$ based on the location coordinates of the robots to analyze the behavior of

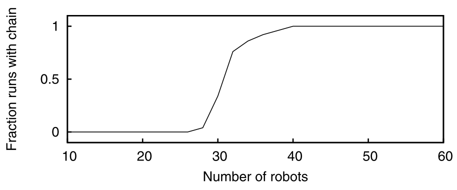


Fig. 15 Fraction of test runs for collective navigation in which a stable dynamic chain forms

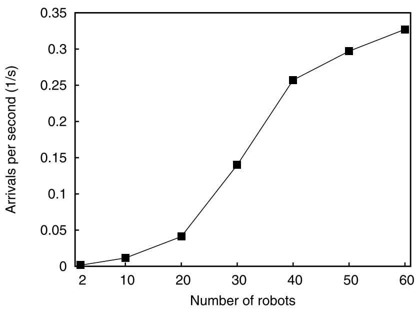


Fig. 16 Transport performance of the swarm: frequency with which the target robots are reached by the other robots of the swarm as a function of the number of robots in the swarm

the swarm. Compared to the definition of $S(R)$ in Balch (2000), we introduce one change, related to the clustering: we use single linkage clustering, which means that a robot is added to a cluster if it is within distance h of any robot in that cluster. Single linkage clustering can find long stretched clusters (Kuiper and Fisher 1975), which enables it to detect the chaining behavior of the swarm. In Fig. 14, we show the evolution of $S(R)$ over the course of example test runs with 20, 30 and 40 robots; we calculate $S(R)$ at every time step of 0.1 s, and average it per 100 s of simulation. When the robots of the swarm move close together, there is a drop in entropy. When the dynamic chain forms, entropy stays low for an extended amount of time. All runs with 20 and 40 robots display patterns similar to the ones shown here: for 20 robots, the chain never forms, while for 40 robots it forms quickly and remains for the whole duration of the simulation. With 30 robots, varying patterns have been observed. In some runs, including the example here, the chain forms after a while. In other runs, it does not form. Interestingly, when it does form, it usually stays for the whole test duration. This suggests that the chain is stable with respect to perturbations.

In Fig. 15, we study the *stability* of the chain. For increasing numbers of robots, we perform each time 50 test runs, and measure in which fraction of those runs a stable dynamic chain appears. We consider the chain stable if for the last 1000 s of the test $S(R)$ remains below 0.2 (these thresholds were set empirically; see Fig. 14 to understand how they allow distinguishing scenarios where a stable chain has formed from others). The graph shows a clear phase transition around 30 robots: with less robots, the system never self-organizes, with more it always does. Such phase transitions are typical of self-organizing systems in physics and in biology, and have also been observed in swarm robotics (Baldassarre 2008). They indicate that within a given range of a control parameter, the self-organizing behavior is robust and takes place independently of perturbations in the system (e.g., loss of robots due to failures, or the arrival of new robots).

Finally, in Fig. 16, we show how frequently the targets are reached by the robots. This indicates how many items the swarm could *transport* between the two locations. Increasing the swarm size, one could expect a sub-linear performance improvement, because more robots can transport proportionally more items (linear improvement), but there is also increased congestion. In our system, increased swarm size also gives more cooperation, which leads

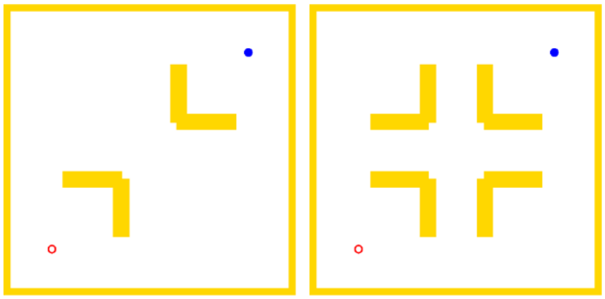


Fig. 17 Layout of cluttered environments used for collective navigation experiments. The area is 20×20 m^2 in both cases

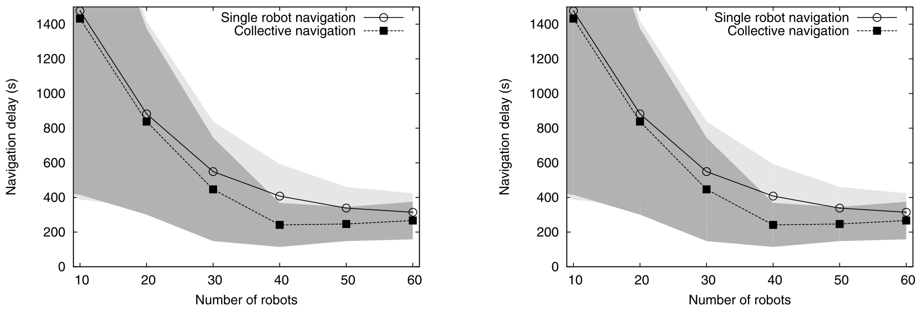


Fig. 18 Test results in the cluttered environments of Fig. 17. The *left plot* refers to Fig. 17 (left), and the *right one* to Fig. 17 (right). The plots report the observed navigation delays (average and standard deviation) for an increasing number of robots, for the case in which all robots in the swarm go back and forth between nest and food source (collective navigation), and for the case in which only one robot is going back and forth between nest and food while the other robots of the swarm move according to the random direction mobility model (single robot navigation). The NwR strategy has been used in all the reported experiments

to a super-linear increase in performance between 10 and 40 robots (the slope of the performance curve increases between each two measurement points). Above 40 robots, congestion makes the performance growth decrease.

4.2 Cluttered environments

In this section we consider the cluttered environments of Fig. 17. The nest and food source are placed in the same locations as in the uncluttered environment, but now obstacles have been placed between them. We compare again single robot navigation and collective navigation. We report the average navigation delay in experiments with varying swarm sizes in Fig. 18. As in the case without obstacles studied in Sect. 4.1, collective navigation is more efficient than single robot navigation. However, as the environment gets more complex, its advantage gets smaller. This is because the swarm has more difficulties to form and maintain the dynamic chain around the obstacles. This also results in a more unstable performance (high standard deviation). We illustrate this in Fig. 19, where we show the evolution of the hierarchic social entropy over time for a typical test run with 40 robots in the scenario of Fig. 17 (right). The entropy is low for certain stretches of time, indicating that the dynamic chain is formed, but also goes up again, showing that the chain gets lost sometimes. These results show that the self-organized behavior works in the presence of obstacles, but that it has difficulties when the environment becomes too complex. In such environments, the

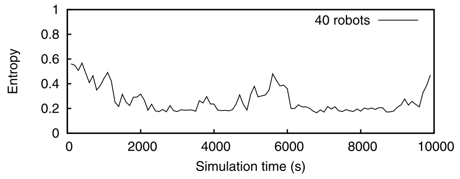
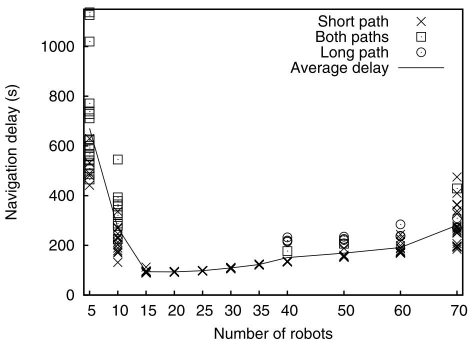


Fig. 19 Evolution of the hierarchic social entropy $S(R)$ over the course of an example test run for 40 robots in the environment of Fig. 17 (right)



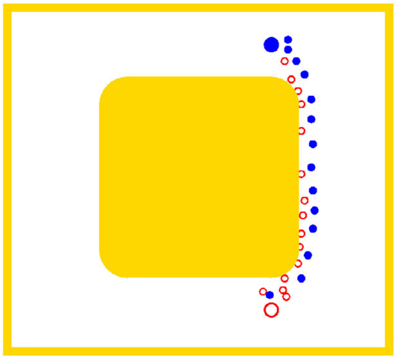


Fig. 20 Shortest path finding performance in the cluttered environment of Fig. 8. The *left figure* shows the navigation delay versus number of robots for each individual test, as well as the average per swarm size (25 tests per swarm size). The choice of path in each test is shown by the *point symbols*. The *right figure* shows a snapshot of the dynamic chain formation observed during one of the simulation experiments

algorithm still works, but it loses the advantage obtained through self-organization, and the performance becomes comparable to that obtained in single robot navigation.

4.3 Shortest path finding

As in the case of single robot navigation, we investigate the behavior of the system in case two paths of different length are available between nest and food source. We use the environment of Fig. 8, where we now place a nest and a food source in the locations that were previously used for searcher and target. The two locations are connected by a short path of length $d_s = 12$ and a long path of length $d_l = 24$. We vary the swarm size from five to 70 robots, and perform 25 tests of 5000 s for each size. We measure the average time a robot needs to navigate between the target locations. We also observe which path each robot takes to reach its target: the short or the long one. We combine this per test run, to calculate the percentage of robots using the short path, p_s. If $p_s > 90\ \%$, we say the swarm uses the short path, if $p_s < 10\ \%$ it uses the long path, and otherwise it uses both.

Figure 20 shows the result of each individual test, as well as the average per swarm size. As in the case of single robot navigation, the robots have a preference for the short path. Also, this preference leads to efficient navigation, as those runs that use the short path usually experience a lower navigation delay (with exception for swarm size 70, where congestion starts to play a major role).

It is interesting to observe the evolution of the preference for the short path for increasing swarm sizes. For swarm size 5, the preference for the short path is rather modest. For 10 robots, the preference is already much stronger, but it is starting from swarm size 15 that the results start to look different: in all runs, all robots always use the short path, and navigation delay is very low and equal over all runs. This *highly efficient navigation behavior* is due to the self-organized formation of the dynamic chain. On the one hand, we observe here the same improvement of navigation efficiency as in uncluttered environments (see Sect. 4.1). On the other hand, there is also a second effect, namely that the dynamic chain makes the collective navigation lock onto one of the paths: once the swarm forms the chain on one path, it will normally not change to the other path anymore. This means that the normal preference for the short path (as observed in single robot navigation) is further reinforced by the chain formation, such that the short path emerges as a stable solution chosen by the swarm for navigation. Between 15 and 30 robots, there are enough robots to form the chain over the short path, but not over the long path. This makes the swarm always choose the short path. Starting from 35 robots, the chain can also be formed over the long path (verified in separate tests not shown here), and we start to observe this from swarm size 40. While the robots' general preference for the short path normally makes the chain form there, fluctuations due to the robots' random initial distribution, or due to collisions and congestion, let the chain occasionally choose the long path. Such amplification of fluctuations (where one of several system-level states are chosen based on small differences in the initial states of the system components) is a typical phenomenon observed in self-organizing systems in nature (Prigogine and Stengers 1984). We also conducted tests placing the targets at different locations, so as to reduce the difference between d_s and d_l (swarm size 50). This led to proportional changes in the number of runs choosing the short path.

5 Implementation on real robots

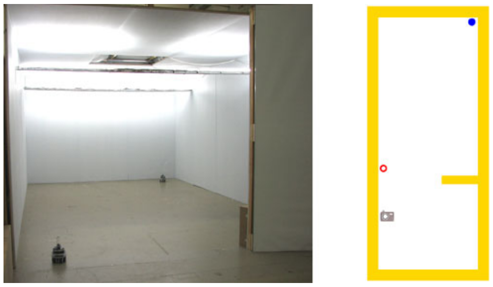
We implemented the communication-based navigation system on real foot-bots (Bonani et al. 2010). Since this is the robot used as model in the simulation experiments, the robot characteristics (IrRB capacity, robot speed, etc.) are the same as described in Sect. 3.

In a first experiment, we used an arena of 10×4 m^2, which is largely uncluttered, except for a wall of 1.4 m on the side. Figure 21 shows a photograph of the arena, as well as an image of how it was reproduced in simulation. We placed a source and target robot in this arena, in the locations of the two robots shown in the figure. Due to the small size of the arena, we limited the communication range of the IrRB system to 2.5 m. We carried out tests similar to the ones reported in Fig. 12: we compare single robot navigation (1 searcher, all other robots perform random movements) and collective navigation (all robots are searchers) in tests with increasing swarm sizes (from one moving robot, up to 10). For each swarm size, we run one single long experiment of 30 minutes, in which the searching robot(s) go back and forth many times. We also reproduce the same experiments in simulation. We report the average time needed for navigation between the source and target. The results are shown in Fig. 22. Both in reality and in simulation, the data show the same trend as in the earlier results of Fig. 12: navigation delay improves and gets more stable (lower standard deviation) with increasing numbers of robots, but for collective navigation, there is a faster improvement thanks to the chain formation. This chain formation was also observed visually by us.

Moreover, although there are some quantitative differences between simulation and real robots results, the trends are qualitatively the same (both for the average and the standard deviation).

In a second experiment, we did tests similar to the ones of Sect. 4.3. We used an arena of 3.10×4.35 m^2, with in the middle an obstacle of 0.75×1.75 m^2. The target and source were placed on either side of the obstacle, at about two thirds along the long edge of the

Fig. 21 Arena used in the real robot experiments: photograph (*left*) and as reproduced in simulation (*right*). The photograph was taken from the position of the camera icon in the right image. In this image, the *circle* and the *disk symbols* indicate, respectively, the position of food and nest robots



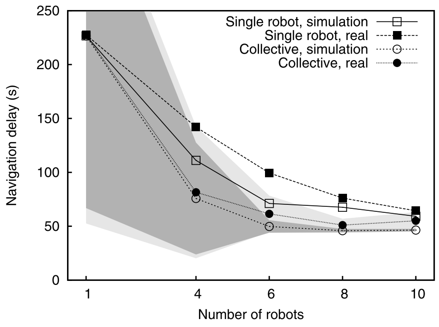


Fig. 22 Experimental results for single and collective navigation using real robots in the scenario of Fig. 21 (left). The average navigation delay is reported for an increasing number of robots. The results obtained in simulation considering the equivalent scenario of Fig. 21 (right) are also reported to show the good correspondence between simulation and real robot behavior. For the collective navigation scenarios, we also show the standard deviation, both for simulation and for real robots

arena, such that a long and a short path were available among them. Figure 23 shows a photograph of the arena, as well as an image of how it was reproduced in simulation. Due to the small size of the arena, we restricted the communication range of the robots to 1.5 m. We ran tests with increasing numbers of moving robots, from one up to eight, for both single robot navigation and collective navigation, and reproduced the same tests in simulation. Each test lasted 40 minutes, but for the collective navigation, we split this up into 4 times 10 minutes. This is because the chain formation makes the robots' choice for either the short or the long path stable for long time, such that consecutive trips between source and target cannot be considered as independent samples; in single robot navigation, on the other hand, the correlation between consecutive trips of the searching robot is limited, such that all trip times gathered during a single run of 40 minutes give enough independent test samples. The results are shown in Fig. 24. We report the average delay needed to move between source and target, and the fraction of robots following the short path. The correspondence between simulation and real robots is good, qualitatively, although quantitatively there are some differences. The trip time results show that the difference between single robot and collective navigation is limited in this case, due to the small size of the arena which causes more congestion. This also results in a high standard deviation on the navigation times. On the other hand, the choice for the shortest path shows a strong difference between the two navigation scenarios: while single robot navigation leads to a preference for the short path that increases linearly with the number of robots, collective navigation has a faster increasing preference for the short path, due to the chain formation (compare to Fig. 20).

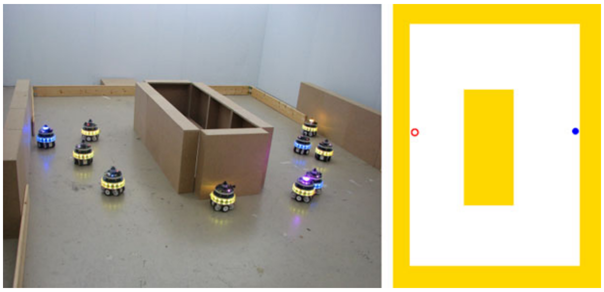
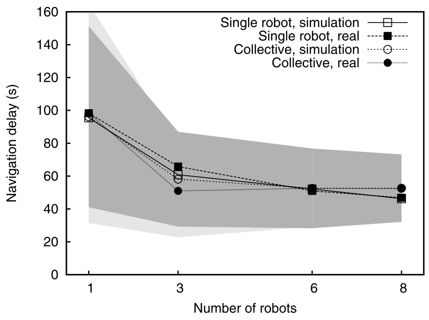


Fig. 23 Arena used in the real robot experiments with obstacle: photograph (*left*) and as reproduced in simulation (*right*). The circle and the *disk symbols in the right image* indicate the position of the food and the nest robots



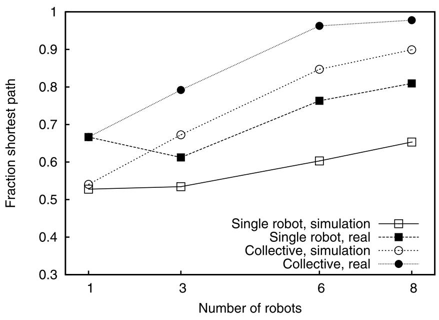


Fig. 24 Experimental results for single and collective navigation using real robots in the cluttered scenario of Fig. 23 (left). In the *left figure*, the observed average delay for the navigation between source and target is reported for an increasing number of robots. For collective navigation, we also show the standard deviation (both for simulation and for real robots). The *right figure* shows the fraction of robots following the short path. The results obtained in simulation considering the equivalent scenario of Fig. 23 (right) are also reported to show the good correspondence between simulation and real robot behavior

Finally, we point out that in previous work (Ducatelle et al. 2009), we implemented the navigation algorithm on e-puck robots (Mondada et al. 2009), fitted with an IrRB communication board (Gutiérrez et al. 2008). The capacities of these robots and their IrRB system are limited compared to those of the foot-bots: each robot could send only 2 bytes per second, with significant packet loss, and very noisy range and bearing estimates. Nevertheless, the navigation system worked fairly well. We refer to Ducatelle et al. (2009), where we report results from these tests. Finally, in some other tests using the foot-bot robots, reported in Ducatelle et al. (2011b), we observed frequent robot failures, and we tested adding robots to and removing robots from the swarm, with the navigation algorithm adapting easily to such changes. All these tests show the general robustness, adaptivity and scalability of the algorithm.

Videos of these experiments as well as of similar experiments used for the paper (Ducatelle et al. 2011b) can be seen in on-line supporting material at: http://www.idsia.ch/~gianni/SwarmNavigation/videos.html. Moreover, a general illustration of the behavior of the algorithm, both in simulation and using real robots, can be found in the video `cooperative-navigation.mp4` included in the web site of the publisher as Electronic Supplementary Material.

6 Related work

In this paper, we have presented an algorithm for communication based cooperative navigation in swarm robotics. The works closest related to ours are situated in the areas of communication-based navigation, and cooperative navigation in swarm robotics.

Several works are related to ours because of the way they use communication to guide navigation. One setup that has been studied extensively over the past few years is to fit the environment with a network of wireless communication nodes, which guide a single robot to a target (Batalin and Sukhatme 2007; O'Hara et al. 2008). The communication nodes may be wireless sensor nodes, which sense the local environment and take this sensed information into account when planning a path (Li and Rus 2005), or nodes without sensors, which use only communication for path planning (O'Hara and Balch 2004). Many of these approaches use communication links to define obstacle-free paths, e.g., using infrared communication (O'Hara et al. 2008), so that they can use network routing algorithms to define navigation paths. An important difference with our approach is that most of these works assume that the communication network that guides the mobile robot is static and embedded in the environment; they do not foresee the possibility that mobile robots guide each other's navigation. Some works do use mobile robots, e.g., to deploy the static communication nodes (Batalin and Sukhatme 2004), or to fill gaps in the sensor network (Witkowski et al. 2008). The closest approach to ours is Sgorbissa and Arkin (2003), where a navigating robot gets support to move around obstacles from a few mobile explorer robots, using line-of-sight communication. Different from our work, however, these few explorer robots are dedicated to support the single robot's navigation task. The authors do not consider the possibility that a whole swarm of mobile robots guide each other's navigation, where each robot may be involved in a task of its own and is not dedicated to support the navigation of the others.

Within the context of swarm robotics, most work on cooperative navigation is based on indirect stigmergic communication (Werger and Matarić 1996; Wodrich and Bilchev 1997; Sharpe and Webb 1999; Russell 1999; Sugawara et al. 2004; Garnier et al. 2007; Fujisawa et al. 2008; Mayet et al. 2010; Nouyan et al. 2008, 2009 Ducatelle et al. 2011a), rather than on direct communication as in our algorithm. This approach is typically inspired by the behavior of certain types of ants, where individual ants mark their paths using a chemical substance, called pheromone, and follow these pheromone trails to find their way between the nest and a food source (Bonabeau et al. 1999). The joint pheromone laying and following actions of the ants of a colony reinforce the most efficient paths, and lets the swarm as a whole self-organize to find shortest paths (Deneubourg et al. 1990). An important problem for the application of such approaches in robotics is how to physically implement pheromone. A common solution is to mark the trail with a chain of robots (Werger and Matarić 1996; Nouyan et al. 2009). Compared to our system, this has the disadvantage that some of the robots remain static and cannot take part in navigation. Moreover, the system is vulnerable to failures of robots in the chain, making it less robust. Other approaches include the use of alcohol (Russell 1999; Fujisawa et al. 2008), phosphorescent paint (Mayet et al. 2010), or light encoding of pheromone using an overhead projector (Garnier et al. 2007; Sugawara et al. 2004), which are interesting, but in practice might be rather hard to detect and follow reliably or to implement. A general disadvantage of all these pheromone-inspired swarm navigation algorithms is that they crucially assume that all robots move between two targets. Our algorithm can also work in this situation, with properties that are similar to other swarm navigation methods; in particular, it lets the swarm self-organize, and displays emergent shortest path finding behavior, as shown in Sect. 4. However, it is also very general and usable in a wide range of different situations. We have illustrated the single robot navigation task in Sect. 3, but many other different scenarios could easily be addressed with this algorithm. One work that is somewhat similar to ours in the context of the single robot navigation task, is *pheromone robotics* (Payton et al. 2001), where robots spread out over an area and indicate the direction to a goal robot using infrared communication. Compared to our work, however, this approach requires robots to adapt their movements to cooperate in the search for the target, and it cannot deal with situations of sparse robot density (it requires the robots of the swarm to form a connected network).

Finally, there are a number of cooperative swarm navigation algorithms that do not implement pheromone-based navigation. Vaughan et al. (2002), propose a method based on direct communication, partially inspired by the bee waggle dance: robots inform each other about the way to a target by exchanging a list of landmarks, in the form of waypoint coordinates. Like pheromone-based methods, however, also this approach assumes that all robots of the swarm navigate back and forth between two targets. Also in Gutiérrez et al. (2010), robots use direct communication to help each other navigate between a nest and a food source. Here, the robots exchange the estimated position of targets (nest or food source), and a robot searching a target can move directly towards the indicated location. However, since the only navigation information used are target locations, the method would not be able to indicate obstacle-free paths in cluttered environments. Sperati et al. (2011) address the collective navigation problem with neuro-evolution. Interestingly, they find a swarm-level behavior that is similar to our dynamic chain, though based on very different individual robot behavior (using visual feedback, robots turn around in local dynamic chains; these chains merge and grow and may eventually include the targets). However, this behavior was not designed to generalize to scenarios that are radically different from the one for which it was developed, namely a collective navigation scenario in an uncluttered environment. Finally, Schmickl and Crailsheim (2008) propose a navigation method inspired by trophallaxis, which is the behavior of social insects to pass food to each other. In this method, the food corresponds to navigation information, which is exchanged through local direct communication. The authors evaluate their method in the context of a foraging task performed by a large swarm of simulated robots. It is not clear whether this method would be applicable in other contexts, such as the single robot navigation task, or in small swarms, and whether it would be usable on real robots.

7 Conclusions and future work

We have presented a navigation system for robotic swarms. It consists of a simple and flexible algorithm that can be used in different contexts in which robots need to collectively find navigation paths toward one or more locations in the environment, where the locations can represent sites associated to tasks or events to deal with. We have studied in depth, both in simulation and using real robots, two specific but at the same time paradigmatic scenarios. In the first one, we have shown how the navigation algorithm allows robots of a swarm to guide a single robot to a target location, without the need to adapt their own movements. In the second scenario, we have investigated how the system can be used for collective navigation between two targets, a common task in swarm robotics. We have shown that cooperation improves navigation performance, and that when enough robots are present, the swarm self-organizes into a dynamic structure that supports efficient navigation and is robust to perturbations and robot failures. Moreover, we have shown that collective navigation has a preference for short paths, similar to pheromone mediated navigation in ant colonies. In tests with real robots, we have shown the feasibility of the approach and the good correspondence between simulation and real-world results.

In future work, we aim to investigate the performance of the current system in more complex scenarios. Moreover we will investigate single robot navigation with different, realistic robot movement patterns, and study the dynamic chain behavior in complex cluttered environments. Future work will also include performing more extensive tests with real robots to confirm all results from simulation. After that, we will integrate this system in other scenarios of swarm deployment, e.g., where the swarm performs tasks to support human activities. Many such scenarios require navigation. Moreover, the swarm communication we use for navigation can be extended to carry more information, e.g., for task allocation, planning, etc.

Acknowledgements The research presented in this paper was partially supported by the European Commission via the Future and Emerging Technologies projects SWARMANOID (grant IST-022888), and ASCENS (grant IST-257414), and by the European Research Council via the ERC Advance Grant "E-SWARM: Engineering Swarm Intelligence Systems" (grant 246939).

The work was also supported by the Swiss National Science Foundation through the National Centre of Competence in Research (NCCR) Robotics.

Marco Dorigo and Rehan O'Grady acknowledge support from the Belgian F.R.S.-FNRS, of which they are a research director and a postdoctoral researcher, respectively.

References

Balch, T. (2000). Hierarchic social entropy: an information theoretic measure of robot group diversity. *Autonomous Robots*, *8*(3), 209–238.

Baldassarre, G. (2008). Self-organization as phase transition in decentralized groups of robots: a study based on Boltzmann entropy. In L. Jain, X. Wu, & M. Prokopenko (Eds.), *Advanced information and knowledge processing*. *Advances in applied self-organizing systems* (pp. 127–146). London: Springer.

Baldassarre, G., Parisi, D., & Nolfi, S. (2004). Measuring coordination as entropy decrease in groups of linked simulated robots. In *Proceedings of the fifth international conference on complex systems (ICCS)* (pp. 1–14). Boston: The New England Complex Systems Institute.

Batalin, M., & Sukhatme, G. (2004). Coverage, exploration and deployment by a mobile robot and communication network. *Telecommunication Systems Journal*, *26*(2), 181–196. Special Issue on Wireless Sensor Networks.

Batalin, M., & Sukhatme, G. (2007). The design and analysis of an efficient local algorithm for coverage and exploration based on sensor network deployment. *IEEE Transactions on Robotics*, *23*(4), 661–675.

Bettstetter, C. (2001). Mobility modeling in wireless networks: categorization, smooth movement, and border effects. *ACM SIGMOBILE Mobile Computations and Communications Review*, *5*(3), 55–66.

Bettstetter, C., Hartenstein, H., & Pérez-Costa, X. (2004). Stochastic properties of the random waypoint mobility model. *Wireless Networks*, *10*(5), 555–567.

Blažević, L., Giordano, S., & Le Boudec, J.-Y. (2002). Self organized terminode routing. *Cluster Computing*, *5*(2), 205–218.

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. New York: Oxford University Press.

Bonani, M., Longchamp, V., Magnenat, S., Rétornaz, P., Burnier, D., Roulet, G., Vaussard, F., Bleuler, H., & Mondada, F. (2010). The marXbot, a miniature mobile robot opening new perspectives for the collective-robotic research. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4187–4193). Washington: IEEE Computer Society.

De Wolf, T., & Holvoet, T. (2005). Emergence vs. self-organisation: different concepts but promising when combined. In *LNCS: Vol. 3464*. *Engineering self-organising systems* (pp. 77–91). Berlin: Springer.

Deneubourg, J.-L., Aron, S., Goss, S., & Pasteels, J.-M. (1990). The self-organizing exploratory pattern of the Argentine ant. *Journal of Insect Behavior*, *3*, 159–168.

Di Caro, G. A., Ducatelle, F., & Gambardella, L. M. (2005). AntHocNet: an adaptive nature-inspired algorithm for routing in mobile ad hoc networks. *European Transactions on Telecommunications*, *16*(5), 443–455.

Dorigo, M., & Sahin, E. (2004). Guest editorial: swarm robotics. *Autonomous Robotics*, *17*(2–3), 111–113.

Dorigo, M., Floreano, D., Gambardella, L. M., Mondada, F., Nolfi, S., Baaboura, T., Birattari, M., Bonani, M., Brambilla, M., Brutschy, A., Burnier, D., Campo, A., Christensen, A. L., Decugnière, A., Di Caro, G. A., Ducatelle, F., Ferrante, E., Förster, A., Gonzales, J. M., Guzzi, J., Longchamp, V., Magnenat, S., Mathews, N., de Oca, M. M., O'Grady, R., Pinciroli, C., Pini, G., Rétornaz, P., Roberts, J., Sperati, V., Stirling, T., Stranieri, A., Stützle, T., Trianni, V., Tuci, E., Turgut, A. E., & Vaussard, F. (2013). Swarmanoid: a novel concept for the study of heterogeneous robotic swarms. *IEEE Robotics & Automation Magazine*, *20*(4).

Dousse, O., Thiran, P., & Hasler, M. (2002). Connectivity in ad-hoc and hybrid networks. In *Proceedings of the 21th IEEE conference on computer communications (INFOCOM)* (pp. 1079–1088). New York: IEEE Press.

Dubois-Ferriere, H., Grossglauser, M., & Vetterli, M. (2003). Age matters: efficient route discovery in mobile ad hoc networks using encounter ages. In *Proceedings of the 4th ACM international symposium on mobile ad hoc networking & computing (MobiHoc)* (pp. 257–266). New York: ACM.

Ducatelle, F., Foerster, A., Di Caro, G. A., & Gambardella, L. M. (2009). Supporting navigation in multi-robot systems through delay tolerant network communication. In *Proc. of the IFAC workshop on networked robotics (NetRob)* (pp. 25–30). Philadelphia: Elsevier.

Ducatelle, F., Di Caro, G. A., Pinciroli, C., & Gambardella, L. M. (2011a). Self-organized cooperation between robotic swarms. *Swarm Intelligence*, *5*(2), 73–96.

Ducatelle, F., Di Caro, G. A., Pinciroli, C., Mondada, F., & Gambardella, L. M. (2011b). Communication assisted navigation in robotic swarms: self-organization and cooperation. In *Proceedings of the 24th IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 4981–4988). Washington: IEEE Computer Society.

Durrant-Whyte, H., & Bailey, T. (2006). Simultaneous localisation and mapping (SLAM): part I. *IEEE Robotics & Automation Magazine*, *13*(2), 99–110.

El-Shehawey, M. (2009). On the gambler's ruin problem for a finite Markov chain. *Statistics & Probability Letters*, *79*(14), 1590–1595.

Fall, K. (2003). A delay-tolerant network architecture for challenged internets. In *Proceedings of the ACM conference on applications, technologies, architectures, and protocols for computer communications, SIGCOMM* (pp. 27–34). New York: ACM.

Fujisawa, R., Dobata, S., Kubota, D., Imamura, H., & Matsuno, F. (2008). Dependency by concentration of pheromone trail for multiple robots. In *LNCS: Vol. 4217. Proceedings of ANTS 2008, 6th international workshop on ant algorithms and swarm intelligence* (pp. 283–290). Berlin: Springer.

Garnier, S., Tache, F., Combe, M., Grimal, A., & Theraulaz, G. (2007). Alice in pheromone land: an experimental setup for the study of ant-like robots. In *Proc. of the IEEE swarm intelligence symp. (SIS)* (pp. 37–44). Washington: IEEE Computer Society.

Groenevelt, R., Nain, P., & Koole, G. (2005). The message delay in mobile ad hoc networks. *Performance Evaluation*, *62*(1–4), 210–228.

Grossglauser, M., & Vetterli, M. (2006). Locating mobile nodes with ease: learning efficient routes from encounter histories alone. *IEEE/ACM Transactions on Networking*, *14*(3), 457–469.

Gutiérrez, A., Campo, A., Dorigo, M., Amor, D., Magdalena, L., & Monasterio-Huelin, F. (2008). An open localization and local communication embodied sensor. *Sensors*, *8*(11), 7545–7563.

Gutiérrez, A., Campo, A., Monasterio-Huelin, F., Magdalena, L., & Dorigo, M. (2010). Collective decision-making based on social odometry. *Neural Computing & Applications*, *19*(6), 807–823.

Jacquet, P., Mans, B., & Rodolakis, G. (2010). Information propagation speed in mobile and delay tolerant networks. *IEEE Transactions on Information Theory*, *56*(10), 5001–5015.

Johnson, D., & Maltz, D. (1996). Dynamic source routing in ad hoc wireless networks. In T. Imielinski & H. Korth (Eds.), *Mobile computing* (pp. 153–181). Dordrecht: Kluwer Academic.

Karlsson, G., Almeroth, K., Fall, K., May, M., Yates, R., & Lea, C.-T. (Eds.), (2008). Special issue on Delay and disruption tolerant wireless communication. *IEEE Journal on Selected Areas in Communications*, *26*(5), 745–840.

Klein, D., Hespanha, J., & Madhow, U. (2010). A reaction-diffusion model for epidemic routing in sparsely connected manets. In *Proceedings of the 29th IEEE international conference on computer communications (INFOCOM)* (pp. 884–892). Piscataway: IEEE Press.

Kuiper, F., & Fisher, L. (1975). A Monte Carlo comparison for six clustering procedures. *Biometrics*, *31*, 777–784.

Li, Q., & Rus, D. (2005). Navigation protocols in sensor networks. *Transactions on Sensor Networks*, *1*(1), 3–35.

Mayet, R., Roberz, J., Schmickl, T., & Crailsheim, K. (2010). Antbots: a feasible visual emulation of pheromone trails for swarm robots. In *Proceedings of the 7th international conference on swarm intelligence (ANTS)* (pp. 84–94).

Mirats Tur, J., Zinggerling, C., & Corominas-Murtra, A. (2009). Geographical information systems for map based navigation in urban environments. *Robotics and Autonomous Systems*, *57*(9), 922–930.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.-C., Floreano, D., & Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions*, Instituto Politécnico de Castelo Branco (IPCB), Portugal (Vol. 1, pp. 59–65).

Nain, P., Towsley, D., Benyuan, L., & Zhen, L. (2005). Properties of random direction models. In *Proceedings of the 24th IEEE conference on computer communications (INFOCOM)* (Vol. 3, pp. 1897–1907).

Nouyan, S., Campo, A., & Dorigo, M. (2008). Path formation in a robot swarm: self-organized strategies to find your way home. *Swarm Intelligence*, *2*(1), 1–23.

Nouyan, S., Gross, R., Bonani, M., Mondada, F., & Dorigo, M. (2009). Teamwork in self-organized robot colonies. *IEEE Transactions on Evolutionary Computation*, *13*(4), 695–711.

O'Hara, K., & Balch, T. (2004). Pervasive sensor-less networks for cooperative multi-robot tasks. In *Proc. of the 7th int. symp. on distributed autonomous robot systems (DARS)* (pp. 305–314). Tokyo: Springer.

O'Hara, K., Walker, D., & Balch, T. (2008). Physical path planning using a pervasive embedded network. *IEEE Transactions on Robotics*, *24*(3), 741–746.

Payton, D., Daily, M., Estowski, R., Howard, M., & Lee, C. (2001). Pheromone robotics. *Autonomous Robots*, *11*(3), 319–324.

Perkins, C., & Bhagwat, P. (1994). Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *ACM SIGCOMM conference on communications architectures, protocols and applications* (pp. 234–244). New York: ACM.

Perkins, C. E., & Royer, E. M. (1999). Ad-hoc on-demand distance vector routing. In *Proc. of the 2nd IEEE workshop on mobile computing systems and applications (WMCSA)* (pp. 90–100). Washington: IEEE Computer Society.

Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Di Caro, G., Ducatelle, F., Birattari, M., Gambardella, L. M., & Dorigo, M. (2012). ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence*, *6*(4), 271–295.

Prigogine, I., & Stengers, I. (1984). *Order out of chaos*. New York: Bantam.

Pugh, J., & Martinoli, A. (2006). Relative localization and communication module for small-scale multi-robot systems. In *Proc. of the IEEE int. conf. on robotics and automation (ICRA)* (pp. 188–193). Washington: IEEE Computer Society.

Roberts, J., Stirling, T., Zufferey, J., & Floreano, D. (2009). 2.5d infrared range and bearing system for collective robotics. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 3659–3664). Washington: IEEE Computer Society.

Royer, E. M., & Toh, C.-K. (1999). A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, *6*(2), 46–55.

Russell, R. (1999). Ant trails—an example for robots to follow? In *Proceedings of IEEE international conf. on robotics and automation (ICRA)* (pp. 2698–2703). Washington: IEEE Computer Society.

Schmickl, T., & Crailsheim, K. (2008). Trophallaxis within a robotic swarm: bio-inspired communication among robots in a swarm. *Autonomous Robots*, *25*(1–2), 171–188.

Sgorbissa, A., & Arkin, R. C. (2003). Local navigation strategies for a team of robots. *Robotica*, *21*, 461–473.

Shalizi, C., Shalizi, K., & Haslinger, R. (2004). Quantifying self-organization with optimal predictors. *Physical Review Letters*, *93*(11), 118701 (4 pages).

Shannon, C. (1949). *The mathematical theory of communication*. Urbana-Champaign: University of Illinois Press.

Sharpe, T., & Webb, B. (1999). Simulated and situated models of chemical trail following in ants. In *Proc. of the 5th int. conf. on the simulation of adaptive behavior (SAB)* (pp. 195–204). Cambridge: MIT Press.

Sperati, V., Trianni, V., & Nolfi, S. (2011). Self-organized path formation in a swarm of robots. *Swarm Intelligence*, *5*(2), 97–119.

Spyropoulos, T., Psounis, K., & Raghavendra, C. (2004). Single-copy routing in intermittently connected mobile networks. In *Proceedings of the first annual IEEE communications society conference on sensor and ad hoc communications and networks (SECON)* (pp. 235–244).

Spyropoulos, T., Psounis, K., & Raghavendra, C. (2006). Performance analysis of mobility-assisted routing. In *Proc. of the 7th ACM int. symp. on mobile ad hoc networking and computing (MobiHoc)* (pp. 49–60). New York: ACM.

Spyropoulos, T., Psounis, K., & Raghavendra, C. (2008). Efficient routing in intermittently connected mobile networks: the single-copy case. *IEEE/ACM Transactions on Networking*, *16*(1), 63–76.

Sugawara, K., Kazama, T., & Watanabe, T. (2004). Foraging behavior of interacting robots with virtual pheromone. In *Proc. of the IEEE/RSJ int. conf. on intelligent robots and systems (IROS)* (pp. 3074–3079). Washington: IEEE Computer Society.

Vaughan, R., Støy, K., Sukhatme, G., & Matarić, M. (2002). Lost: localization-space trails for robot teams. *IEEE Transactions on Robotics and Automation*, *18*(5), 796–812.

Werger, B. B., & Matarić, M. J. (1996). Robotic food chains: externalization of state and program for minimal-agent foraging. In *Proc. of the 4th int. conf. on the simulation of adaptive behavior (SAB)* (pp. 625–634). Cambridge: MIT Press.

Witkowski, U., El-Habbal, M., Herbrechtsmeier, S., Tanoto, A., Penders, J., Alboul, L., & Gazi, V. (2008). Ad-hoc network communication infrastructure for multi-robot systems in disaster scenarios. In *Proceedings of the IARP/EURON workshop on robotics for risky interventions and surveillance of the environment (RISE)* (published online at http://www.robot.uji.es/research/events/rise08).

Wodrich, M., & Bilchev, G. (1997). Cooperative distributed search: the ants' way. *Control and Cybernetics*, *26*, 413–446.

Zhang, X., Neglia, G., Kurose, J., & Towsley, D. (2007). Performance modeling of epidemic routing. *Computer Networks*, *51*(10), 2867–2891.