Cooperative Self-Organization in a Heterogeneous Swarm Robotic System

Frederick Ducatelle
Gianni A. Di Caro
Luca M. Gambardella

Technical Report No. IDSIA-01-10

IDSIA / USI-SUPSI
Dalle Molle Institute for Artificial Intelligence
Galleria 2, 6928 Manno, Switzerland

Cooperative Self-Organization in a Heterogeneous Swarm Robotic System

Frederick Ducatelle [*]
Gianni A. Di Caro
Luca M. Gambardella

Abstract

We study how a swarm robotic system consisting of two different types of robots can solve a foraging task. The first type of robots are small wheeled robots, called foot-bots, and the second type are flying robots that can attach to the ceiling, called eye-bots. While the foot-bots perform the actual foraging, i.e. they move back and forth between a source and a target location, the eye-bots are deployed in stationary positions against the ceiling, with the goal of guiding the foot-bots. The key component of our approach is a process of mutual adaptation, in which foot-bots execute instructions given by eye-bots, and eye-bots observe the behavior of foot-bots to adapt the instructions they give. Through a simulation study, we show that this process allows the system to find a path for foraging in a cluttered environment. Moreover, it is able to converge onto the shortest of two paths, and spread over different paths in case of congestion. Since our approach involves mutual adaptation between two sub-swarms of different robots, we refer to it as cooperative self-organization. This is to our knowledge the first work that investigates such a system in swarm robotics.

[*]Corresponding author. This work was partially supported by the SWARMANOID project, funded by the Future and Emerging Technologies programme (IST-FET) of the European Commission under grant IST-022888. The information provided is the sole responsibility of the authors and does not reflect the European Commissions opinion. The European Commission is not responsible for any use that might be made of data appearing in this publication.

1 Introduction

Swarm robotics is the study of robotic systems consisting of a large group of relatively small and simple robots that interact and cooperate with each other in order to jointly solve tasks that are outside their own individual capabilities. The behavior of the collective is expected to emerge from the interactions between the individual robots, a process referred to as self-organization. The robots that make up a swarm robotic system are usually homogeneous; heterogeneous swarm robotic systems are considered if they consist of separate groups (sub-swarms) of homogeneous robots [5]. In this paper we study a system consisting of two sub-swarms of robots with very distinct capabilities, and we investigate how they can self-organize to jointly solve a foraging task.

We consider the following problem setup. A swarm of wheeled robots, called foot-bots, is deployed in an indoor environment to solve a foraging task, i.e., they need to go back and forth between a source and a target location to transport objects. For the navigation between the two locations, they are assisted by a swarm of flying robots that can attach to the ceiling, called eye-bots. These are deployed beforehand (e.g. using the algorithm described in [20]) and form a stationary grid on the ceiling between the source and target location. From their position on the ceiling they give directional instructions to the foot-bots on the ground, to guide them towards the source or the target location. The difficulty of the problem lies in the fact that the topology of the terrain is different on the ceiling and on the ground. This is a normal situation in any indoor environment, where one can expect to find tables, chairs, cupboards and other obstacles that can obstruct the way for foot-bots but not for eye-bots. This means that eye-bots cannot rely on their own sensor feedback (e.g., distance scanner) to find a path between the source and target location and derive instructions to give to foot-bots.

To deal with this situation, we use an adaptive solution, in which eye-bots start by giving random instructions to foot-bots, and foot-bots give feedback about their behavior and experiences, so that eye-bots can adapt the instructions they give. Through this process, the heterogeneous system of eye-bots and foot-bots is able to cooperatively find paths through the environment. Moreover, as we will show in the rest of this paper, it is capable of finding shortest paths and of spreading over multiple paths in case of congestion. Since the behavior of the system as a whole does not result from the interactions within each swarm, but rather from the interactions between the members of the two swarms, we refer to it as *cooperative self-organization*. To the best of our knowledge, this is the first example of such a system in the swarm robotics literature.

The rest of this paper is organized as follows. First, we give details about the robots and the problem setup. Then, we describe the algorithm. After that, we present experimental results that show that the robots are able to find a path through a cluttered environment. Next, we discuss and investigate how the system can find shortest paths, and how it can spread out over multiple paths in case of congestion. Finally, we describe related work.

2 Problem description

In this section, we first present the robots that are used in this work, and then we describe the problem setup.

2.1 The robots

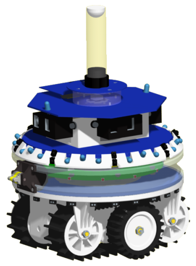
The foot-bot and the eye-bot robots were both developed as part of the EU-funded project Swarmanoid[1]. The foot-bot is shown in Figure 1(a). It is about 15 cm wide and long and 20 cm high. It moves on the ground using Treels, which are a combination of tracks and wheels. It has two cameras, one omnidirectional to see other foot-bots and one pointing up to see eye-bots. Foot-bots can communicate with each other and with eye-bots through visual signaling, using the 256 color LED ring that is placed around their body and the powerful LED beacon they have centrally on top. Moreover, they can exchange wireless messages locally at low bandwidth using a 2.5D infrared range and bearing (IrRB) system [16], which also provides them with relative positional information about each other. The eye-bot is shown in Figure 1(b). It is a flying robot with a diameter of 35 cm and a height of 45 cm. It uses a powerful co-axial central rotor system for upward thrust and four side thrusters for stability control. It can attach to the ceiling using a magnet (the design assumes the presence of ferromagnetic ceilings), which allows it to save energy. The eye-bot has a pan-and-tilt camera which can be pointed in any direction below or around it. Like the foot-bot, it can communicate with visual signals using a multi-color LED ring that is placed all around its body, or with wireless messages using the IrRB system. Details about both robots can be found in [22].

2.2 The scenario

The eye-bots and foot-bots are placed in an indoor arena such as the one shown in Figure 2.

[1]`http://www.swarmanoid.org`

(*a*)



(*b*)

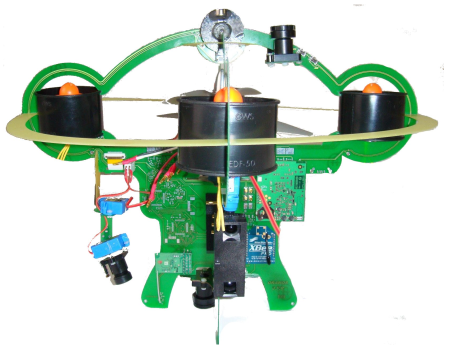


Figure 1: Swarmanoid robots: (a) the Footbot (CAD draw) and (b) the Eyebot (prototype).

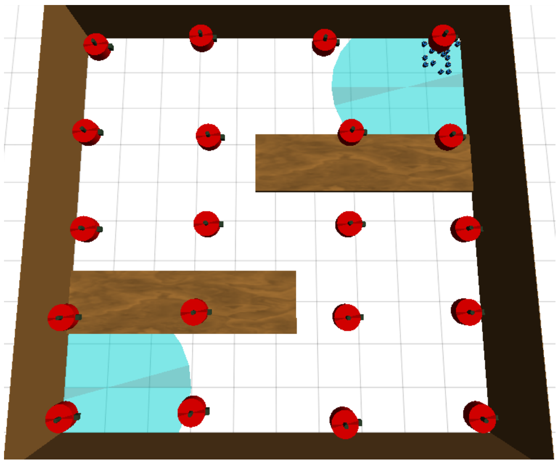


Figure 2: Example problem scenario. The eye-bots form a grid covering the arena. The foot-bots are deployed in the start location at the top right of the arena. The target location is at the bottom left.

The task of the foot-bots is to go back and forth between a source location (top right in the figure) and a target location (bottom left in the figure). The role of the eye-bots is to support the foot-bots in this task, by giving directional instructions. The eye-bots are attached to the ceiling in a grid formation that covers the area between the source and the target, as shown in the figure. We do not study how they can obtain this formation; we refer to [20], where the authors describe how the eye-bots can search an environment and form a connected grid between source and target using the IrRB system. We assume the eye-bots have formed such a grid and investigate how they can guide foot-bots between the two locations.

The main difficulty lies in dealing with obstacles. While the walls surrounding the arena reach from the floor to the ceiling, and can therefore easily be sensed by both foot-bots and eye-bots, other obstacles are lower (e.g., the two blocks in the middle of the arena in Figure 2), so that they block the way for foot-bots, but not for eye-bots. In human-made environments, this situation is very common, as most objects, such as, for example, tables, cupboards or sofas, are of limited height. This means that eye-bots cannot rely on their sensor feedback, or on infrared communication in the connected grid between them, to find a path between the source and target location for foot-bots. A possible solution could be to get camera images from the different eye-bots, join them, and perform image recognition on them, to create a map of the environment in which a path can be calculated. However, this would violate basic principles of swarm robotics related to distributed operation, scalability and the use of simple interaction patterns. In what follows, we solve the problem using an adaptive self-organized approach.

3 Cooperative self-organized path finding

In this section we describe the algorithm used by eye-bots and foot-bots to cooperatively find a path through the cluttered environment. We first give an overview of the working of the system. Then, we explain how eye-bots give directions to foot-bots, give details about foot-bot behavior, and describe how eye-bots update and use their policies.

3.1 General description

The general idea behind the approach is that foot-bots execute directional instructions they receive from eye-bots, and eye-bots observe foot-bot behavior and feedback in order to adapt the instructions they give.

We let eye-bots send foot-bots in a finite number of discrete directions. We use 12 different directions, so one direction every $\pi/6$ radians. The preference to send foot-bots in each of these directions is stored in a policy, which is implemented as an array of floating point numbers. Each eye-bot maintains two different policies: policy P_t for the target and policy P_s for the source. At each time step (of 0.1 s), the eye-bot draws two directions randomly, d_t for the target and d_s for the source, whereby the probability distributions to draw directions are based on the two policies. These directions are then broadcast locally to nearby foot-bots.

The eye-bots update P_t and P_s at each time step, based on the observed behavior of each foot-bot within their field of view. They consider three different aspects of foot-bot behavior: the goal of the foot-bot (whether it is going to the target or the source), the direction it is coming from d_f (relative to the eye-bot's orientation), and whether or not the foot-bot is performing obstacle avoidance. When an eye-bot observes a foot-bot that is going towards the target, it assumes that the foot-bot is coming from the source, so it increases the policy P_s for direction d_f, and decreases the policy P_t for that same direction. When the eye-bot observes a foot-bot performing obstacle avoidance, it decreases both policies P^s and P^t for the direction in which it sees the foot-bot, assuming that direction is blocked by obstacles.

To be able to derive the necessary information about the foot-bot behavior, the eye-bot requires feedback from the foot-bot. This is given in the form of light signals. The foot-bot simultaneously switches on its LED beacon on top and one LED in front, in order to clearly show the eye-bot its movement direction. The color of the front LED is used to indicate whether the foot-bot is going towards the source or the target, and the color of the LED beacon to show whether it is doing obstacle avoidance or not.

3.2 Giving directional instructions

Eye-bots give directional instructions to foot-bots using a combination of visual signals with LEDs and wireless communication with the IrRB system. They switch on a red LED in front and a blue LED in the back, in order to show foot-bots a reference direction d_0. Then, they broadcast the two chosen directions d_s and d_t using wireless communication over the IrRB system. IrRB communication from eye-bots to foot-bots is focused in a cone, so that only foot-bots underneath the eye-bot can receive its messages.

In order to get directions, a foot-bot moves under an eye-bot. It uses its upward camera to define d_0, and extracts direction d_s or d_t (depending on whether the foot-bot's goal is the source or the target) from the received wireless message. The foot-bot interprets d_s or d_t as a relative direction with respect to d_0, in order to derive a new travel direction d_n. It first turns into that direction, and then moves forward for a default distance (2.5 meters, which is enough to get out of the field of view of the eye-bot it received the message from), or until it arrives under a different eye-bot. If after the default distance no other eye-bot has been reached, the foot-bot uses its upward camera to define the direction to the closest eye-bot, and moves there. If no eye-bot is seen, the foot-bot starts a random movement: repeatedly make a random turn and move forward for a random distance.

This communication scheme is scalable for the number of foot-bots and eye-bots, since wireless communication is limited to one message per time step broadcast locally by each eye-bot. All other communication is via light signals.

3.3 Foot-bot behavior

Foot-bot movements are guided by the instructions of eye-bots, as outlined above in Section 3.2. However, Foot-bots have a preference to move forward, in order to direct the exploration away from the source. This preference is implemented as follows. When a foot-bot receives from an eye-bot eb_1 a travel direction that is forward (i.e., between $-\pi/2$ and $\pi/2$) with respect to the travel direction received from the previous eye-bot eb_0, the foot-bot follows the given direction and does not consider other directions received in subsequent time steps from eb_1. If, however, the received travel direction is backward, the foot-bot follows the direction but simultaneously keeps listening for other instructions from eb_1. If eb_1 has a strong preference for that backward direction, it will send the same direction to the foot-bot again in the next time steps, so that the foot-bot keeps turning back. However, if one of the subsequent directions sent by eb_1 is forward, the foot-bot will use that without listening to other directions, and will not turn back. Besides this, foot-bots have a low-level obstacle avoidance behavior, which makes them turn away smoothly from obstacles that are observed using infrared proximity sensors.

Foot-bots manipulate their LED signals depending on their behavior. Besides changing the color of the front LED and the LED beacon, as described before, they also switch off the front LED in certain occasions. This way, eye-bots can see where they are and whether they are doing obstacle avoidance (through the color of the LED beacon, which is not switched off), but not the direction they are coming from, d_f. As a consequence, eye-bots cannot update their policy for d_f. Foot-bots do this whenever their movement direction is not representative for the general direction they are following from source to target: when they are performing obstacle avoidance, when they are following an instruction that sends them backward, or when they have not received any eye-bot instruction (e.g., because they did not observe an eye-bot). The goal is to reduce noise in eye-bot policies.

3.4 Updating policies and drawing directions

As described in Section 3.1, an eye-bot increases its policy for the source P_s in the direction that foot-bots going towards the target come from, and decreases it in the direction that foot-bots going towards the source come from. To update the policy for the target P_t, the inverse rule is followed. Both P_s and P_t are decreased in the direction that robots doing obstacle avoidance are observed. To update a policy P in a given direction d, the eye-bot first calculates the discrete policy index i corresponding to d. Then, policy increases are performed using an additive constant c_a (set to 0.5), as shown in equation 1, while policy decreases are performed using a multiplicative constant c_m (set to 0.975), as shown in equation 2. All policy entries are initialized using a small constant (set to 1/12).

$$P[i] \leftarrow P[i] + c_a \tag{1}$$

$$P[i] \leftarrow P[i] * c_m \tag{2}$$

Eye-bots draw directions from the policies using an ϵ-soft rule. With a constant probability of ϵ (set to 0.5), the direction with the highest preference is chosen. Otherwise, a direction is chosen randomly, from a probability distribu-

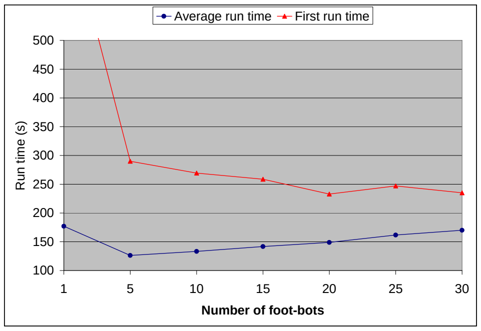


Figure 3: The time needed by foot-bots to go between source and target. We report the time of the first foot-bot that reaches the target, and the average time over all completed runs of all foot-bots. We vary the number of foot-bots from 1 up to 30.

tion that is proportional to the relative preferences of directions in the policy.

4 Finding paths in a cluttered environment

In this section, we investigate whether our system is able to find a path in a cluttered environment, and whether it can efficiently solve the foraging task. We first present the simulation setup, and then experimental results.

4.1 Simulation setup

All tests presented in this section and in the rest of this paper are done with the Swarmanoid simulator [23], which was developed as part of the Swarmanoid project. It uses the Open Dynamics Engine library [19] for the calculation of physical movements and collisions of the robots and their environment. The simulator contains precise models of the foot-bot and eye-bots robots.

All experiments last 2000 seconds. We carry out 30 independent runs for each test.

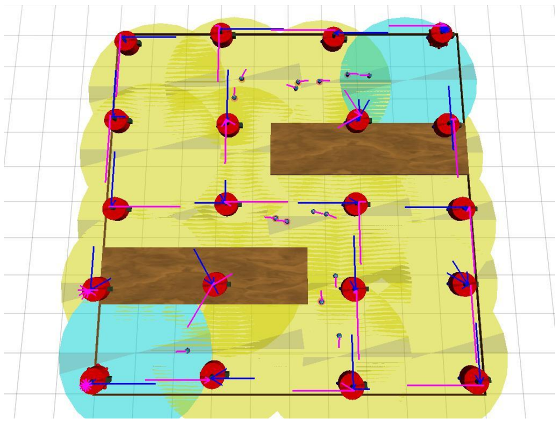


Figure 4: Snapshot of the system's foraging behavior after 500 s in an experiment with 15 foot-bots.

4.2 Experimental results

We study the scenario of Figure 2. We vary the number of foot-bots from 1 up to 30. We measure the time from the start of the experiment until the first foot-bot reaches the target, which we refer to as t_1, and the average time needed by foot-bots to complete any of their runs between source and target, which we refer to as t_a. The results are shown in Figure 3.

The first foot-bot cannot rely on experiences of other foot-bots to find its way to the target, and hence performs random exploration. Once the first foot-bot has reached the target, it can for its way back profit from stored information in the eye-bot policies. A comparison between t_a and t_1 shows that on average, the foot-bots need much less time for each run than what the first foot-bot needed to find the target. This shows that the system can provide the basic functionality it was programmed for: it is able to learn a path from experience and guide foot-bots between source and target.

For increasing numbers of foot-bots, t_a first decreases, and then increases. The decrease is mainly due to the fact that multiple foot-bots searching in parallel can explore the environment more efficiently, so the system saves time in the exploration phase, when the foot-bots need to find the target for the first time. This is shown by the large decrease in t_1 for increasing numbers of foot-bots: with more foot-bots, the target is found faster. The increase in t_a for higher numbers of foot-bots is due to increased congestion.

Finally, as an example, we show in Figure 4 a snapshot of the system's state after 500 s in an experiment with 15 foot-bots. The lines above the eye-bots show the relative preferences for the different directions in each of the two policies (P_s in blue and P_t in pink). The pink line on each foot-bot shows the way it is heading. The colored circle under each eye-bot shows the area within which it is visible for foot-bots below. From the figure, it can be seen that the policies have converged to show an efficient path between source and target, and that foot-bots align on this path. The eye-bots that are situated above either of the two obstacles are never reached by any foot-bot, so their policies do not affect foot-bot behavior.

5 Shortest path finding

In this section, we investigate in how far our approach is able to pick the shortest among multiple paths. We first comment on the similarity between our system and the behavior of ants in nature. Then, we present experimental results about shortest path finding.

5.1 Ants and pheromone

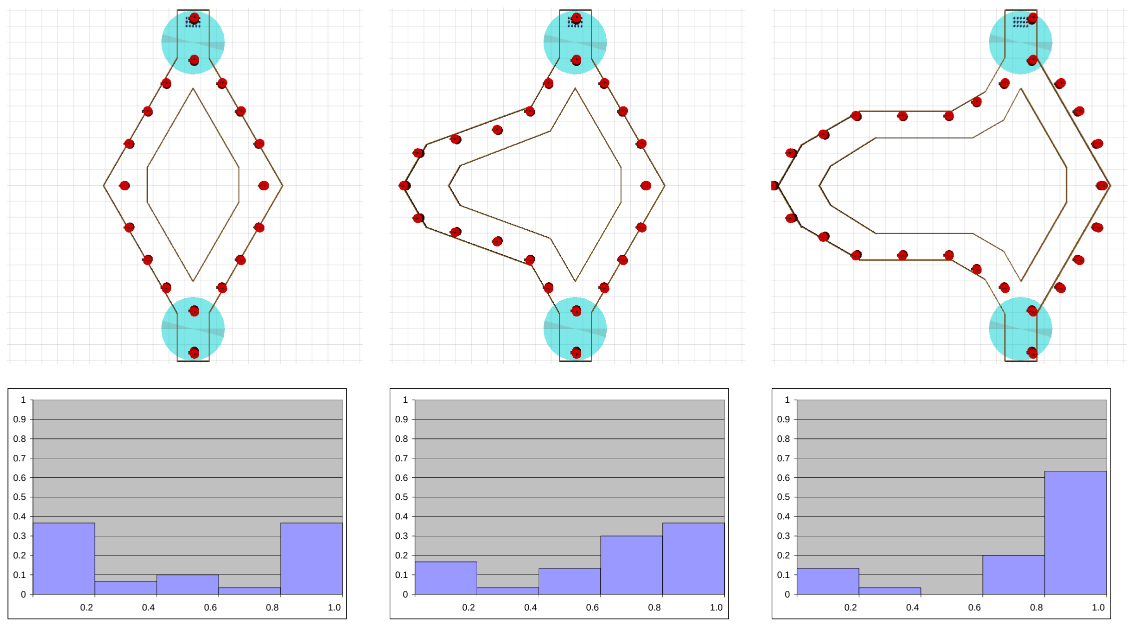
It is interesting to look at our system through the eyes of the involved robots. So far, we have mostly described the system from an eye-bot point of view. In this view, eye-bots store and update policies in order to learn a path between two locations, and they use foot-bots as sampling agents to get feedback about the quality of their policies. It is, however, also possible to consider the system from a foot-bot point of view. In this view, foot-bots try different paths between the two locations, and the role of eye-bots is to store past foot-bot experiences and communicate them to other foot-bots. Seen this way, eye-bots play the role of stigmergic communication points for foot-bots in the environment.

When considering the foot-bot point of view, it is easy to see the similarities with the stigmergic path-finding process of ants in nature [4]. That process is based on the use of a chemical substance called pheromone: while going between their nest and a food source, ants deposit pheromone, and they also preferentially go in directions where they sense higher pheromone intensities. This way, ants follow and reinforce paths indicated by other ants. In our system, the policies stored by the eye-bots play the role of artificial pheromone. The preferences for different directions are updated based on foot-bot movements, and foot-bots are also sent in directions that have higher preference. The system can therefore be seen as an example of a practically feasible application of ant-inspired path finding in swarm robotics.

An interesting aspect is that the pheromone following behavior of ants is not only able to find a path between the nest and a food source, but also to converge onto the shortest path [9]. This is because the shortest path can be completed faster and more frequently by ants, and therefore receives more pheromone, which in turn attracts more ants. In what follows, we investigate experimentally whether our system is similarly capable of finding shortest paths.

5.2 Experimental results

We used the test scenarios shown in Figure 5. The source and target locations are connected by two corridors. We vary the ratio $r = l_l/l_r$, where l_l is the length of the left corridor and l_r is the length of the right corridor: we use $r = 1$, $r = 1.5$ and $r = 2$. This setup is similar to the



(a) (b) (c)

Figure 5: Double bridge experiments with varying ratio r between bridge lengths: (a) $r = 1$, (b) $r = 1.5$ and (c) $r = 2$. We show a histogram of the distribution of the ratio r_c (number of foot-bots observed on the right branch versus total number of observed foot-bots) over 30 independent runs: the x-axis shows r_c, discretized into 5 intervals, while the y-axis shows which fraction of the 30 runs falls into each interval.

one used with real ants in [9]. Like in [9], we place the corridors at an angle of $\pi/6$ radians with respect to the source and target area, to avoid that foot-bots double back into the other corridor. We use 15 foot-bots, which we deploy one by one with a time interval of 50 s between each foot-bot.

We gather statistics in the second half of each experiment (i.e., after 1000 s), when all foot-bots have been deployed and the system has had time to explore the area and settle on a path. We count over all remaining time steps how many foot-bots can be found in the right corridor, c_r, and how many in the left corridor, c_l. We calculate the ratio $r_c = c_r/(c_r + c_l)$, which is near 1 or near 0 if the foot-bots have converged onto respectively the right or the left corridor, and near 0.5 if they use both corridors. In Figure 5, we show a histogram summarizing the values of r_c measured in 30 different test runs (on the x-axis the values of r_c discretized into 5 intervals, on the y-axis the fraction of the 30 runs that falls into each of the different intervals).

It can be seen that in the case of equal corridors ($l_r = l_l$), the foot-bots converge onto one of them, which can be either the left or the right. In a few cases, both corridors are used equally. This behavior is the same as for ants [9], as well as for mathematical models of ant behavior [11]. When the corridors are of different length, the foot-bots converge more often onto the shortest corridor, and this effect gets stronger as the difference between the corridors increases. This indicates that our self-organized system is able to find the shortest path.

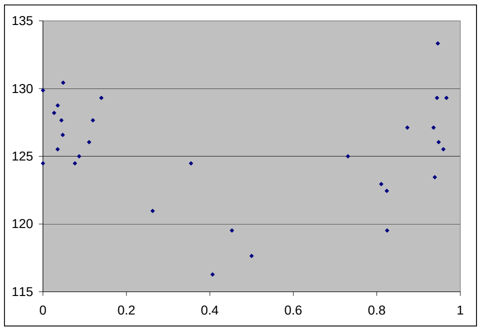


Figure 6: For each of the 30 independent test runs of the experiments with equal corridors of Figure 5(a), we plot the ratio r_c (x-axis) versus the average time t_a needed by foot-bots to go between source and target (y-axis).

Finally, we want to point out that this shortest path finding behavior requires the use of multiple foot-bots. When testing the scenario of Figure 5(b) with 10 or less foot-bots we found that the system converged onto both corridors equally likely.

6 Self-organized spreading

In this section, we study the problem of congestion. We first describe the problem in detail, and explain how ants in nature handle it. Then, we discuss how our system deals with congestion.

6.1 Congestion and ants

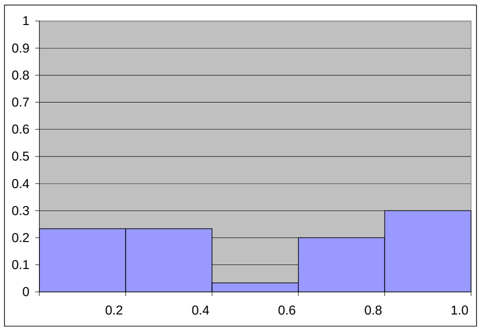
Congestion is an important issue in multi-robot systems. In Section 4.2, we explained that the time required by foot-bots to go between source and target increases for increasing numbers of foot-bots, due to congestion. One way to handle congestion is to spread robot traffic over multiple paths, increasing the maximum throughput of the system. Especially in the environments of Figure 5, where two distinct paths are available, this can be a good solution. We illustrate the effect of traffic spreading in Figure 6. For each of the 30 independent test runs of the experiments with equal corridors of Figure 5(a), we plot the ratio r_c versus the average time t_a needed by foot-bots to go between source and target. One can clearly see that the few runs in which foot-bot traffic was spread over both corridors (r_c close to 0.5) obtained a better performance in terms of t_a. This shows that traffic spreading can alleviate the effects of congestion. The question is how spreading can be obtained automatically in a distributed self-organized system.

Interestingly, ants in nature are capable of automatic traffic spreading. When two paths of equal length are available, they converge onto one of them when ant traffic is low, and spread over both when ant traffic is high [6]. The mechanism behind this phenomenon is based on direct interactions between ants: in crowded conditions, ants were found to physically push each other onto different paths.

Since robots, like ants, are embodied agents, physical interactions play an important role in their behavior. These interactions are expected to increase in crowded conditions. A mechanism of automatic traffic spreading similar to that of ants could therefore be used in our swarm robotic system.

6.2 Foot-bot traffic spreading

Surprisingly, experiments showed that our self-organized path finding behavior, without any modifications, is capable of foot-bot traffic spreading. We used the scenario with equal corridors of Figure 5(a), with increased numbers of foot-bots. In Figure 7, we show the distribution of the ratio r_c for tests with 20 robots (Figure 7(a)) and 25 robots (Figure 7(b)). Compared to the distribution for 15 foot-bots shown in Figure 5(a), higher levels of congestion increases the number of test runs with intermediate levels of r_c, indicating that the system converges less often onto a single corridor, but rather spreads the traffic over both.

(*a*)


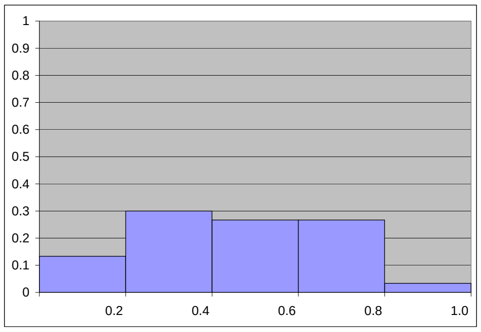
(*b*)


Figure 7: We perform tests in the setup of Figure 5(a) with (a) 20 foot-bots and (b) 25 foot-bots. We show a histogram with the distribution of the ratio r_c over 30 independent runs.

A visual investigation of the system's behavior revealed that traffic spreading is indeed caused by physical interactions between foot-bots. However, the main driving factor is not so much the fact that foot-bots push each other into a different corridor, but rather that they execute obstacle avoidance behavior to steer away from each other. As pointed out in Section 3, eye-bots observe where foot-bots are performing obstacle avoidance, and reduce their policies in those directions, assuming there is no way through there. While we intended this mechanism in the first place to let the system learn quickly about obstacles in the environment, it also causes the foot-bots to steer away from congested areas where other foot-bots are spending a lot of time avoiding each other. This way, automatic foot-bot traffic spreading emerges from our self-organized path finding behavior.

As a final remark, we point out that in case of foot-bot traffic spreading, foot-bots use both corridors in both directions (source to target and target to source). Given that eye-bots derive the direction towards the source from the movements of foot-bots going towards the target (i.e., they derive the path in one direction from movements in the opposite direction), it is impossible for our system to learn a separate path to go and to come back. Also foraging ants are not able to separate opposite flow directions [6], unlike for example pedestrians [10].

7 Related work

While we know of no other research that deals with the same problem we are addressing here, quite a lot of existing work is related to different aspects of our work.

In terms of problem setup, the closest related work in our opinion is the field dedicated to the use of embedded sensor networks to support robot navigation [2, 12, 13, 25]. The general idea behind such systems is to place sensor nodes in the environment and let them cooperate to guide a single mobile robot to a target. Some work considers the use of robots to place the sensors, or even to play the role of sensors [1, 18, 26], which becomes similar to our setup, where eye-bots can be considered to be guiding sensor nodes. An important difference with our work is the central role of network communication in these systems. The sensor nodes calculate the shortest path through the network using a routing algorithm such as Bellman-Ford routing [3], and use this to guide the robot. Two issues arise here. First, they assume that communication links between sensor nodes can be followed by robots. This ignores the case when a path is blocked for robot navigation, but not for sensor communication. This relates directly to the problem we address in this paper, the fact that obstacles are not the same for eye-bots and foot-bots (see Section 2.2). Second, they require communication links between sensor nodes along all paths that are navigable for robots. This is the complementary problem: they ignore the case when a path is blocked for communication between sensor nodes, but not for navigation of robots. In the case of our robots, one could think of a door opening, which does not usually reach all the way to the ceiling: it would let foot-bots through, but not the infrared communication between eye-bots. Our system works also when confronted with this problem, since it does not require communication between eye-bots.

In terms of the solution method, we know of no other work that studies cooperative self-organization between two robot swarms. If we make abstraction of the adaptive behavior of the eye-bots, and consider them as no more than stigmergic communication points for foot-bots, our system can be seen as a particular implementation of ant-inspired pheromone based foraging (as described in Section 5.1). A number of works in swarm robotics have investigated this approach [7, 8, 14, 17, 21, 24, 27]. All these papers are concerned with path finding and shortest path finding, but none of them addresses automatic traffic spreading.

A difficult issue in pheromone based robotic systems is how to implement pheromone. Some authors ignore the problem [14], while others use practically infeasible or unrealistic solutions, such as light projections [8, 21] (a central computer follows robot movements with an overhead camera, calculates pheromone trails, and communicates them to the robots using light projections) or a map in a shared memory [24], assuming that the issue of pheromone implementation will be solved somehow in the future. Other authors experiment with chemical pheromone traces, e.g. using alcohol [7, 17]. In our system, pheromone takes the form of numerical values stored on board of robots, which can be considered a practically feasible implementation of pheromone. A similar approach was followed in pheromone robotics [15], where robots spread out to find a target and communicate among them to create a vector field pointing to the target. However, the use of network communication (rather than adaptation) to find paths makes this work more similar to the sensor network navigation systems described earlier than to our work. Moreover, our approach uses two different robot swarms. Since the robots carrying the pheromone can be mobile and can be deployed separately from the robots following the pheromone, our system is highly flexible.

8 Conclusions

We have described a swarm robotic system that solves a foraging task. It relies on an adaptive process involving two swarms of different types of robots, which we call cooperative self-organization. To our knowledge this is the first example of such a system in the literature. We show that our approach is able to find paths in a cluttered environment, to find shortest paths, and to spread robot traffic in case of congestion. The system also shows a practically feasible approach to implement pheromone in swarm robotics.

In future work, we will develop this system in two different directions. First, we will investigate the movement of eye-bots. So far, we have used the eye-bots in a stationary setup. The goal is to let them also move and adapt their position based on foot-bot feedback. This would allow them to search the best locations to give their instructions. A second research direction is the use of explicit feedback about foot-bot experiences. In the current work, feedback is implicit: eye-bot adapt their policies for all observed foot-bot movements, and the ability to find the shortest path arises from the fact that shorter paths can be completed faster and more often by foot-bots. If foot-bots communicate explicit information about the quality of the path they have followed, learning can be more precise. Such feedback is equivalent to a reinforcement signal, so that the system would implement a form of stigmergic reinforcement learning.

References

[1] M. Batalin and G. Sukhatme. Coverage, exploration and delpoyment by a mobile robot and communication network. In *Proc. of the International Workshop on Information Processing in Sensor Networks*, 2003.

[2] M. Batalin, G. Sukhatme, and M. Hattig. Mobile robot navigation using a sensor network. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, May 2004.

[3] R. Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16(1):87–90, 1958.

[4] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems.* Oxford University Press, 1999.

[5] M. Dorigo and E. Sahin. Guest editorial: Swarm robotics. *Autonomous Robotics*, 17(2–3), 2004.

[6] A. Dussutour, V. Fourcassié, D. Helbing, and J.-L. Denebourg. Optimal traffic organization in ants under crowded conditions. *Nature*, 428:70–73, March 2004.

[7] R. Fujisawa, S. Dobata, D. Kubota, H. Imamura, and F. Matsuno. Dependency by concentration of pheromone trail for multiple robots. In *Proceedings of the 6th International Conference on Ant Colony Optimization and Swarm Intelligence (ANTS)*, 2008.

[8] S. Garnier, F. Tache, M. Combe, A. Grimal, and G. Theraulaz. Alice in pheromone land: An experimental setup for the study of ant-like robots. In *Proceedings of the IEEE Swarm Intelligence Symposium (SIS)*, April 2007.

[9] S. Goss, S. Aron, J. L. Deneubourg, and J. M. Pasteels. Self-organized shortcuts in the Argentine ant. *Naturwissenschaften*, 76:579–581, 1989.

[10] D. Helbing. Traffic and related self-driven many-particle systems. *Reviews of Modern Physics*, 73, October 2001.

[11] A. M. Makowski. The binary bridge selection problem: Stochastic approximations and the convergence of a learning algorithm. In *Proceedings of the 6th International Conference on Ant Colony Optimization and Swarm Intelligence (ANTS)*, 2008.

[12] K. O'Hara and T. Balch. Pervasive sensorless networks for cooperative multi-robot tasks. In *Proceedings of the Seventh International Symposium on Distributed Autonomous Robot Systems (DARS-04)*, 2004.

[13] K. O'Hara, V. Bigio, S. Whitt, D. Walker, and T. Balch. Evaluation of a large scale pervasive embedded network for robot path planning. In *Proceedings 2006 IEEE International Conference on Robotics and Automation (ICRA)*, May 2006.

[14] L. Panait and S. Luke. Ant foraging revisited. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9)*, 2004.

[15] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee. Pheromone robotics. *Autonomous Robots*, 11(3), November 2001.

[16] J. Roberts, T. Stirling, J. Zufferey, and D. Floreano. 2.5d infrared range and bearing system for collective robotics. In IEEE, editor, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-2009)*, St. Louis, October 2009.

[17] T. Sharpe and B. Webb. Simulated and situated models of chemical trail following in ants. In *Proceedings of the 5th International Conference on the Simulation of Adaptive Behavior*, 1998.

[18] T. Sit, Z. Liu, M. Ang Jr., and W. Seah. Multi-robot mobility enhanced hop-count based localization in ad hoc networks. *Robotics and Autonomous Systems*, 55(3):244–252, March 2007.

[19] R. Smith. *Open Dynamics Engine v0.5 User Guide*, 2006.

[20] T. Stirling, S. Wischmann, and D. Floreano. Energy-efficient indoor search by swarms of simulated flying robots without global information. *Swarm Intelligence*, 2010. To appear.

[21] K. Sugawara, T. Kazama, and T. Watanabe. Foraging behavior of interacting robots with virtual pheromone. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3074–3079, October 2004.

[22] Swarmanoid. Final swarmanoid hardware. Internal Deliverable D13 of IST-FET Project *Swarmanoid* funded by the European Commission under the FP6 Framework, 2009.

[23] Swarmanoid. Final swarmanoid simulator. Internal Deliverable D12 of IST-FET Project *Swarmanoid* funded by the European Commission under the FP6 Framework, 2009.

[24] R. Vaughan, K. Støy, G. Sukhatme, and M. Mataric. Whistling in the dark: Cooperative trail following in uncertain localization space. In *Proceedings of the Fourth International Conference on Autonomous Agents*. ACM Press, 2000.

[25] C. Vigorito. Distributed path planning for mobile robots using a swarm of interacting reinforcement learners. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2007.

[26] U. Witkowski, M. El-Habbal, S. Herbrechtsmeier, A. Tanoto, J. Penders, L. Alboul, and V. Gazi. Ad-hoc network communication infrastructure for multi-robot systems in disaster scenarios. In *Proceedings of the International Workshop on Robotics for Risky Interventions and Surveillance of the Environment*, 2008.

[27] M. Wodrich and G. Bilchev. Cooperative distributed search: The ants' way. *Control and Cybernetics*, 26, 1997.