Contents lists available at SciVerse ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Collective energy homeostasis in a large-scale microrobotic swarm

Serge Kernbach *, Olga Kernbach

Institute of Parallel and Distributed High-Performance Systems, University of Stuttgart, Universitätsstr. 38, 70569 Stuttgart, Germany

ARTICLE INFO

Article history:
Received 21 September 2006
Received in revised form
28 July 2011
Accepted 4 August 2011
Available online 19 August 2011

Keywords:
Energy
Energy efficiency
Robotics
Swarm
Self-sufficiency

ABSTRACT

This paper introduces an approach that allows swarm robots to maintain their individual and collective energetic homeostasis. The on-board recharging electronics and intelligent docking stations enable the robots to perform autonomous recharging from low energy states. The procedure of collective decision-making increases collective efficiency by preventing bottlenecks at docking stations and the energetic death of low-energy robots. These hardware and behavioral mechanisms are implemented in a swarm of real microrobots, and several analogies to self-regulating biological strategies are found.

1. Introduction

Homeostasis is one of the distinctive properties of living organisms [1]. The evolutionary sensor–actuator development of living organisms and their behavioral and reproduction strategies depend on certain parameters of their internal regulatory mechanisms [2]. The most important of these is energy balance and, closely connected to this, foraging behavior and strategies [3].

Homeostasis in technical systems differs from biological organisms primarily in the non-autonomy of the energy balance [4]. Technical systems, such as mobile robots, depend on human participation in their energy supply. The energy problem is especially challenging in microrobotic systems [5]; state of the art solutions for autonomous recharging in robots are exemplified by on-board electronics, sensors and power stations (see for example [6] or [7]); alternative approaches are represented by, for example, microbial fuel cells [8] and energy harvesting [9]. Work has been published concerning models of robot foraging [10], bio-inspired energy harvesting strategies [11] and more generally on foraging and scalability; see for example [12].

When autonomous robots have energetic homeostasis, their behavior does not depend solely on behavioral goals defined by a designer [13]. Such robots monitor their energy state, save energy by choosing optimal behavior and autonomously seek recharging stations. In this, we perceive several analogies with biological organisms [14]. To some extent, robots can "feel" hungry, can look for a food source and can exercise certain degrees of behavioral freedom to avoid energetic death. In this way, these biological concepts are re-embodied in robotics [15]. This problem becomes especially hard when many robots perform cooperative energy foraging. Depending on the current energy level in the swarm, these robots can collectively choose different foraging strategies: to allow the robots with the lowest energy to recharge first, in case they do not survive; an altruistic strategy in which all robots recharge for a short time and thus maximize the common energetic level; or an egoistic strategy of maximum individual recharging and competition for resources. These are analogous with the behavioral strategies of animals in regions of distributed food resources [16].

In this paper, we demonstrate autonomous recharging in a microrobotic swarm [17] and investigate the swarm's collective energy homeostasis. This includes the development of relevant hardware and software components for individual robots. The technological constraints of the running and recharging times imposed by the docking and recharging processes define the behavioral strategies for all the robots. The docking station is equipped with a communication system compatible with the robots' [18] and allows the robots to sense the availability of energy. To regulate collective behavior, we implement a procedure for collective decision-making. Since the microrobots have limited computation and communication capabilities, their collective decision-making is based on randomly-coupled map lattices [19], which do not demand sophisticated computation and communication resources. The mechanism used either changes the priority and duration of individual recharging or adjusts individual duty cycles, allowing the robot swarm to adapt to the energetic conditions in the environment. This self-regulation mechanism

* Corresponding author.
E-mail addresses: Serge.Kernbach@ipvs.uni-stuttgart.de (S. Kernbach), Olga.Kernbach@ipvs.uni-stuttgart.de (O. Kernbach).

0921-8890/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2011.08.001

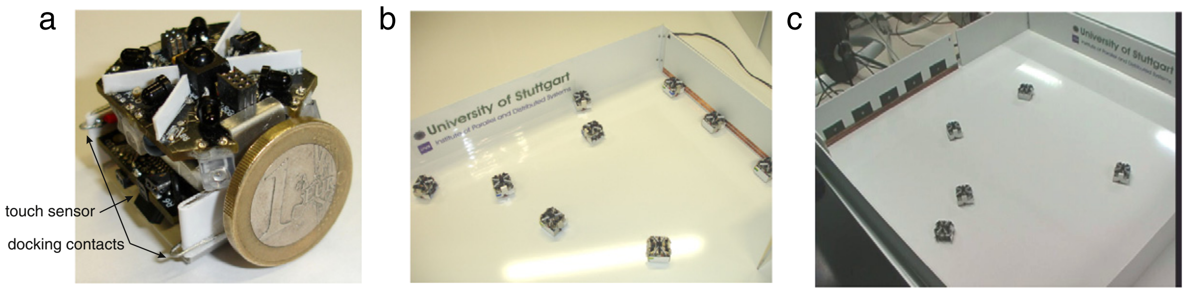


Fig. 1. (a) The "Jasmine" microrobot's recharging contacts. The touch sensor on the front of the robot is also shown. (b) The first docking station, at which each robot can become a communication point. (c) The second docking station, with integrated communication points.

is highly-scalable with regard to swarm density, and introduces the so-called "tech-inspired" approach, in contrast to previously published bio-inspired approaches, for example [11,20].

The paper is organized thus: in Section 2, we describe the hardware components of the robot and the technological constraints imposed on its behavior; Section 3 deals with energetic homeostasis and the software components; Section 4 treats the suggested collective strategy that allows the robots to cooperate in foraging; Section 5 demonstrates the implementation and the experiments performed; and finally, Section 6 offers conclusions from this work.

2. On-board recharging hardware

The "Jasmine" microrobot, shown in Fig. 1(a), measures $30 \times 30 \times 20$ mm^3 in size and has two small DC motors with an integrated planetary gearbox. The microrobot has two circuit boards, the motor board and the main board, which communicate via a 200 kHz I2C interface. The main board holds an ATmega 168 microcontroller, six (60° opening angle) IR channels (used for proximity sensing and communication) and one IR geometry-perception-channel (15° opening angle). The sensing area covers a 360° rose-like area with maximum and minimum ranges of 200 mm and 100 mm respectively [17]. The physical communication range can be decreased through a change of sub-modulation frequency. The main board also supports remote control, differential light sensing, energy management, ZigBee communication and is primarily used for the behavioral control of the robot and for upper extension boards. The motor board has an ATmega 88 microcontroller and is used for motor control, the odometrical system [21], energy control, touch (short-range reflective IR sensor), and color sensing; it also provides another four channels for further sensors/actuators.

To make the microrobots capable of autonomous recharging, four components are required: (a) internal energy sensors, monitoring the energy level of the Li–Po accumulator; (b) recharging circuitry for the Li–Po process; (c) reliable connection with the docking station with low electrical resistance; (d) communication with the docking station. The internal energy sensor is implemented as a resistive voltage divider with a coefficient of 0.55, connected directly to the Li–Po accumulator. The overall resistance is 726k (402k + 324k), and the continuous drain current is about 5 μA. The ADC conversion takes about 64 μs, so monitoring of the energy level can be achieved relatively quickly.

The microrobot uses a single-cell 4.2 V Lithium–Polymer accumulator with a capacity of 250 mAh. The robot consumes about 200 mA when moving and sensing, about 20 mA when sensing only (communicating) and about 10 mA when listening only. Thus, the running time of the robot is approximately 1.25 h. In its optimal working mode, the Li–Po accumulator discharges only 75%–80% of its capacity. The accumulator reaches critical level when the voltage drops to less than 3 V. At this point, the internal power regulator is not able to stabilize the voltage fluctuations and the microcontroller can spontaneously reboot. The recharging current is 1C (250 mA), and full recharging takes about 90 min Partial recharging is almost equal to discharging (that is, 15 min motion requires about 15 min recharging). To control the recharging process, we use the linear Li–Ion/Li–Po battery charger LTC4054-4.2 in a small ThinSOT package.

For autonomous recharging, we developed a simple and reliable solution for the docking station [22]: two 0.4 mm silver-plated wires glued to the front of the robot. The connectors are installed at different heights, see Fig. 1(b) or Fig. 1(c). The docking station comprises a wall to which are attached two strips of copper, 0.2 mm thick by 5 mm wide. Both copper strips are connected to the power supply, which can provide 5 V and 3 A current. The length of the docking station is chosen to allow the simultaneous recharging of 5–10 robots, see Fig. 1(b). Many such docking stations can be placed together, see Fig. 9. To connect to the docking station, a robot has to move to this wall (after receiving the docking signal), until it receives a positive signal from the touch sensor. Then, the robot turns slightly on its wheels, producing a small mechanical strain to maintain reliable mechanical contact. The resistance of such a contact is less than 0.1 Ω (measured statistically).

2.1. Communication with the docking station and docking approach

The development of the docking station and the docking approach has been addressed in several publications, such as [11,22–24,20]. After testing many solutions, we eventually used two setups that demonstrated the best reliability for docking. In the setup shown in Fig. 1(b), every docked robot can became a communication point to guide other robots to the station. The initial communication point is represented by a robot that remains at the left boundary of the docking station and does not move. When a robot approaches the station, it sends a request signal to the station and listens for an acknowledgment. If it receives an acknowledgment, it moves straight to the communication point and sends the request again. No acknowledgment means a free slot exists. The direction of the free slot is given by the receiving sensor, that is, if it is the first sensor, that means a docking slot exists in front, if the second sensor, the robot needs to rotate 30°, and so on. When necessary, a robot rotates and moves forwards until it receives a touch confirmation from the touch sensor. Through monitoring the voltage on the energy sensor, the robot is made aware of the start of recharging. After a robot is docked to the station, it sends the docking acknowledgment to any requesting robots through its rear transmitter.

The setup in Fig. 1(c) uses integrated communication points (the main boards from the robot) at each docking slot. The docking station continuously sends the availability signal and number of free docking positions. This signal can be received up to 10–15 cm away from the docking station. A robot receiving the docking signal for free slots approaches the docking station and sends

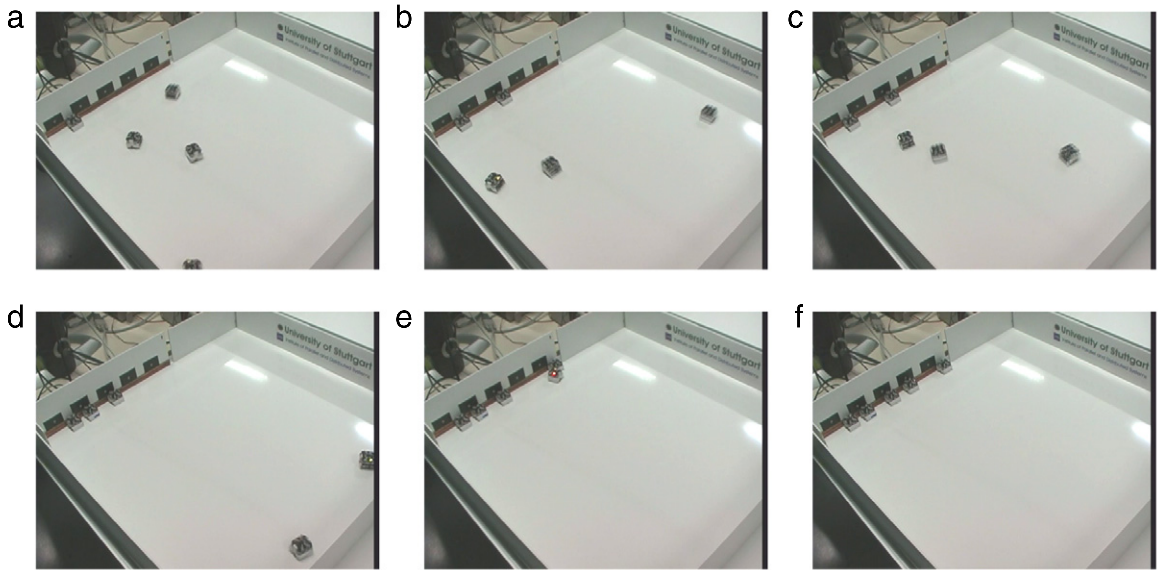


Fig. 2. (a)–(f) A docking run for 5 robots, using the second setup. The images were taken at 10 s intervals.

a recharging request. The docking station decreases the number of free slots by one, and sends a confirmation message to the requesting robot. The robot then starts the docking and recharging process. When recharging is finished, the robot sends a terminating signal to the docking station and moves away, and the docking station increments the number of free slots.

One run for the first setup is shown in Fig. 9 and for the second setup in Fig. 2(a)–(f). Both setups work well and are designed for different purposes. The first setup allows better scalability for the docking approach, however it is less efficient, because the initial communication points (that is, the robots) decrease the recharging capacity of the docking station. The second setup solves this problem, allowing a simpler docking approach. It is more intelligent in terms of knowing the number of free and busy slots but it is more expensive from the hardware point of view. In the experiments in Section 5 we did not require feedback from the intelligent docking station and therefore used the first setup.

3. Energetic homeostasis of individual robots

The energetic homeostasis of a robot is based on the charging-recharging cycle of the Li–Po accumulator[1] and includes five states, shown in Table 1. The robot can manage its energetic behavior, as shown in Fig. 3. First, in a critical state, the robots should stop the collective or individual activity currently being executed. Second, a robot should compare the priority of the currently executed activity Pr($Task$) and the priority of looking for food Pr(S_h). For example, if the priority of the current activity is 0.6, but "hunger" is 0.7, the robot will look for a docking station. Finally, a robot can have a so-called "collective instinct", in which it can recharge only until it reaches the "satisfied state", which takes less time) and frees a slot for another robot to recharge.

Generally, the potential cooperativeness of individual energetic homeostasis can comprise:

- higher priority for collectively executed tasks, for example tasks in which several robots work cooperatively gain maximum priority;

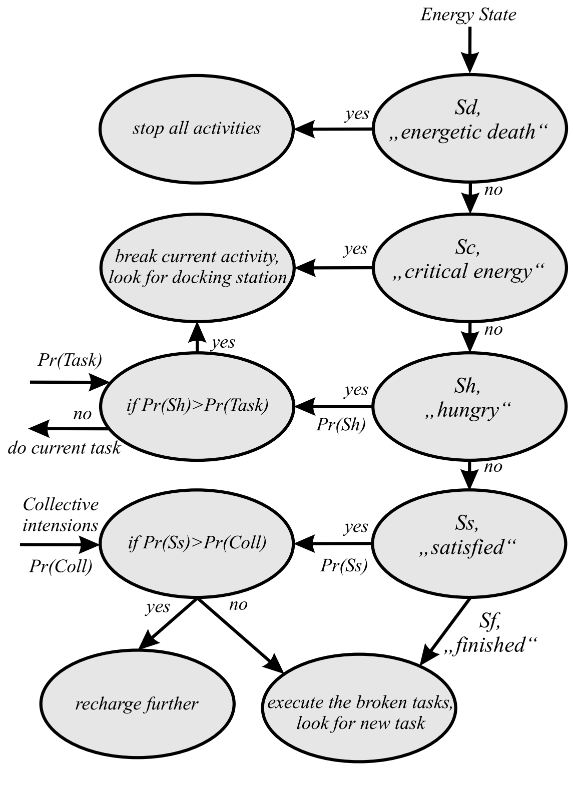


Fig. 3. Structural scheme for energetic homeostasis of the "Jasmine" robots. *Source:* Taken from [11,19].

- activities in which a swarm "permits" express recharging for robots at a critical energy state;
- avoiding full recharging when there are other robots in "hungry" states.

4. Collective strategies in energy foraging

When the robots recharge individually, several undesired effects can appear:

- the docking station can become a "bottleneck" that essentially decreases collective efficiency;

[1] See datasheet for the LTC4054-4.2 standalone linear Li–Ion/Pi–Po battery charger from Linear Technology.

Table 1
Individual energetic states of the microrobot.

S	State	Voltage	ADC value	Comment
S_d	Energetic death	$V < 3.05$	142	The robot should stop and go into stand-by mode. In this state it is not able to react to external stimuli and needs human assistance to recharge
S_c	Critical state	$V < 3.2$	150	The robot should look for the docking station regardless of its current task. It has about 3–5 min to find it before energetic death occurs
S_h	Hungry state	$V < 3.7$	173	The robot can start to look for the docking station when it has no more important tasks to perform. Ideally, when reaching 3.65 V (ADC value 142), the robot should start looking for the docking station.
S_s	Satisfied state	$V < 4.0$	187	The accumulator is not fully recharged (80%–85%), but is charged sufficiently to allow the robot to run again and free the slot for other robots.
S_f	Full state	$V = 4.2$	196	The voltage increases from 4.1 to 4.2 very slowly, (the accumulator is already recharged up to 90%–95%). At this voltage, the circuit stops recharging.

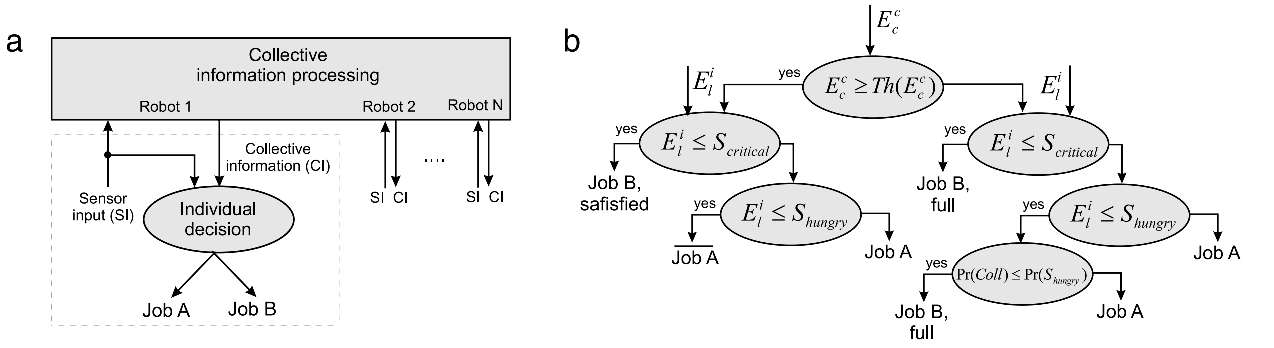


Fig. 4. (a) The structure of decision-making based on collective information processing. (b) The structure of individual decision-making using collectively calculated E_c^c and individual E_l^i. "Job A"—execute the current collective activity, "not Job A"—do not participate in collective activities, "Job B satisfied"—recharge until reaching the satisfied state, "Job B full"—recharge until reaching the fully recharged state.

– robots with a high energy level can occupy the docking station and block low-energy robots, which can then energetically die;
– many robots can "crowd" around the docking station and essentially block the docking approach, increasing the total recharging time and worsening the energetic balance of the whole swarm.

In managing their energetic homeostasis, the robots can make only two possible individual decisions: to execute a current collective task (Job A) or to move to recharging (Job B). A cooperative strategy should enable the correct timing and a correct combination from the individual decisions of all robots.

There are two ways to synchronize individual decisions. First, decision-making can be collective, and all robots must execute that decision. Examples of collective decision-making are bargaining or auctioning, in which the final collective activity is not fixed and must be negotiated [25]. Second, the decision can be made individually, however the information used for this decision is prepared collectively [26]. This decision-making process is usually applied to "switching decisions", where the final collective activity represents a sequence of predefined sub-activities. For energy foraging, the second method is the most suitable; the robots can exchange information about their individual sensor inputs and after the inputs are collectively processed, the robots will receive a signal that will define whether they will do Job A or Job B, see Fig. 4(a). Obviously, the individual state of a robot influences its decision, so that ultimately, collective behavior represents a complex mix between collective "needs" and individual "desires".

We introduce two input values that robots can exchange in collective processing. These are individual energy level E_l^i, representing the digitalized voltage of the accumulator, and individual energy consumption E_c^i, calculated as the energy difference per time unit:

$$E_c^i = \frac{\Delta E_l^i}{\Delta t} = \frac{E_l^i(t_1) - E_l^i(t_2)}{t_2 - t_1}. \quad (1)$$

As well as these individual values, we introduce E_c^c, which represents the corresponding collective value. The value of E_c^c is very useful for estimating the level of collective activity. When E_c^c is high, all the robots move and the whole swarm is very busy; when E_c^c is low, the activity level of the swarm is low.

Individual decision-making takes the form shown in Fig. 4(b). When E_c^c is larger than a given threshold, the swarm is very active. This means that robots with a low energy should look for the docking station and recharge. Recharging need last only until the satisfied state S_s (Job B satisfied) is reached, because full recharging takes more time and other robots can be in a critical energy state. Moreover, these robots should not participate in any collective activities, because they can distort the collective state when they go to recharge (not Job A). When E_c^c is smaller than a given threshold (the swarm is passive), robots at critical energy levels can recharge until the full state S_f (Job B full) is reached. "Hungry" robots should compare the priority of collective tasks with the priority of going to recharge. This collective strategy manages the order of the robots' recharging depending on the collective energy consumption. In the next section, we consider the question of how to calculate the value of E_c^c.

4.1. *Collective calculation of a swarm's energy consumption*

The application of distributed computation with networked algorithms [27] in swarms is a challenging problem because of the limited computational and communication capabilities of microrobots. To calculate collective energy consumption E_c^c, we use the theory of coupled map lattices (CML) [28], well known in nonlinear dynamics. CML allow the derivation of the required distributed computation with a minimum of communication and individual computation. Each robot is required to receive some numerical values from neighboring robots and also transmit its value to its neighbors. We first assume that a robot receives these values only from two neighbor robots, that is, the CML structure is fixed. After that, we extend this approach to a randomly changing structure, that is, when the robots move.

The numerical values that the robots are exchanging represent internal dynamic variables denoted as q_n^i, where n denotes a time step and i the robot's number. When a robot communicates with two neighbors, they create a two-way coupled ring:

$$q_{n+1}^i = aq_n^i + a(q_n^{i-1} + q_n^{i+1}), \quad i = 1, \dots, m, \tag{2}$$

where a is a coefficient, and m is the total number of robots. Our first task is to determine the coefficient a to obtain the required computation. The system (2) is first considered in the low-dimensional case ($m = 4$), then dimension scaling is performed. To determine a, we find the solution of these difference equations using the Jordan normal form approach [29]. Rewriting the system in the form:

$$\underline{\mathbf{q}}_{n+1} = \underline{\underline{\mathbf{A}}}\,\underline{\mathbf{q}}_n, \tag{3}$$

where the matrix $\underline{\underline{\mathbf{A}}}$ is the two-way coupling matrix [30], we introduce new state variables $\underline{\boldsymbol{\xi}}_n$ determined by $\underline{\mathbf{q}}_n = \underline{\underline{\mathbf{V}}}\,\underline{\boldsymbol{\xi}}_n$; thereby the system (3) yields $\underline{\underline{\mathbf{V}}}\,\underline{\boldsymbol{\xi}}_{n+1} = \underline{\underline{\mathbf{A}}}\,\underline{\underline{\mathbf{V}}}\,\underline{\boldsymbol{\xi}}_n$. Multiplying from the left side with the inverse matrix $\underline{\underline{\mathbf{V}}}^{-1}$, we get achieve the diagonalized system:

$$\underline{\boldsymbol{\xi}}_{n+1} = \underline{\underline{\mathbf{J}}}\,\underline{\boldsymbol{\xi}}_n, \tag{4}$$

where the matrices of the eigenvectors $\underline{\underline{\mathbf{V}}}$ and the inverse $\underline{\underline{\mathbf{V}}}^{-1}$ are given by

$$\begin{gathered}\underline{\underline{\mathbf{V}}} = \begin{bmatrix} 1 & -1 & -1 & 0 \\ 1 & 1 & 0 & -1 \\ 1 & -1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}, \\ \underline{\underline{\mathbf{V}}}^{-1} = \frac{1}{4}\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ -2 & 0 & 2 & 0 \\ 0 & -2 & 0 & 2 \end{bmatrix}.\end{gathered} \tag{5}$$

The solution of the system (4) can readily be obtained as

$$\underline{\boldsymbol{\xi}}_n = \underline{\underline{\mathbf{T}}}\,\underline{\boldsymbol{\xi}}_0, \tag{6}$$

where the diagonal matrices $\underline{\underline{\mathbf{J}}}$ and $\underline{\underline{\mathbf{T}}}$ are given by

$$\underline{\underline{\mathbf{J}}} = \begin{bmatrix} 3a & 0 & 0 & 0 \\ 0 & -a & 0 & 0 \\ 0 & 0 & a & 0 \\ 0 & 0 & 0 & a \end{bmatrix}, \quad \underline{\underline{\mathbf{T}}} = \begin{bmatrix} (3a)^n & 0 & 0 & 0 \\ 0 & (-a)^n & 0 & 0 \\ 0 & 0 & a^n & 0 \\ 0 & 0 & 0 & a^n \end{bmatrix}. \tag{7}$$

The solution (6) for the initial state variables $\underline{\mathbf{q}}_n$ has the form:

$$\underline{\mathbf{q}}_n = \underline{\underline{\mathbf{V}}}\,\underline{\underline{\mathbf{T}}}\,\underline{\underline{\mathbf{V}}}^{-1}\underline{\mathbf{q}}_0 \tag{8}$$

or setting $\underline{\mathbf{q}} = \{x, y, z, v\}^T$, we obtain explicitly

$$\begin{bmatrix} x_n \\ y_n \\ z_n \\ v_n \end{bmatrix} = \frac{1}{4}\begin{bmatrix} \beta^+ x_0 + \gamma y_0 + \beta^- z_0 + \gamma v_0 \\ \gamma x_0 + \beta^+ y_0 + \gamma z_0 + \beta^- v_0 \\ \beta^- x_0 + \gamma y_0 + \beta^+ z_0 + \gamma v_0 \\ \gamma x_0 + \beta^- y_0 + \gamma z_0 + \beta^+ v_0 \end{bmatrix}, \tag{9}$$

$$\gamma = \left((3a)^n - (-a)^n\right), \qquad \beta^\pm = \left((3a)^n + (-a)^n \pm 2a^n\right).$$

To obtain the arithmetical mean value

$$q_{n\to\infty}^i = \frac{1}{N}\sum_{j=1}^{N} q_0^j \tag{10}$$

we set $a = 1/3$. Then, in the long time dynamics, the coefficients γ and $\beta^\pm$ will be equal to $\gamma = \beta^\pm = 1$ and each of the state variables of system (2) will be equal to (10).

Dimension scaling. The dimension scaling for the system (2) is based on the properties of the eigenvalues and eigenvectors of a two-way coupling matrix $\underline{\underline{\mathbf{A}}}$ in (3). The eigenvalues of this symmetric band $m \times m$ matrix (see e.g. [30]) are given by

$$\lambda_i = a + 2a\cos\left(\frac{2\pi \mathrm{i}}{m}\right), \quad i = 1, \dots, m. \tag{11}$$

Apparently, the eigenvalues (11) take values maximum of that in absolute magnitude determined by$\lambda_m = 3a$. Setting $a = 1/3$ and moreover $|\lambda_{i=1,\dots,m-1}| < 1$, we get the long time dynamics for the matrix $\underline{\underline{\mathbf{T}}}$, where $T_{[1,1]} = 1$ and $T_{[k\neq 1, l\neq 1]} = 0$. This means the result of $\underline{\underline{\mathbf{V}}}\,\underline{\underline{\mathbf{T}}}\,\underline{\underline{\mathbf{V}}}^{-1}$ in (8) is determined only by the first column of $\underline{\underline{\mathbf{V}}}$ and the first row of $\underline{\underline{\mathbf{V}}}^{-1}$.

Considering the linear problem of eigenvectors for $\underline{\underline{\mathbf{A}}}$, we ascertain that all components of the first column in $\underline{\underline{\mathbf{V}}}$ (eigenvector corresponding to the maximum eigenvalue λ_m) will always be equal to 1 independent of the dimension of (2). However, the difficulty is that we are not always able to analytically determine the inverse matrix of eigenvectors. To get round this problem for the case of $m \leq 100$, we calculate this matrix numerically and thus ascertain $1/m$ for all elements of the first row.

Effect of randomly changing neighbors. In Eq. (2), we assumed that the robots are connected in two-way rings with fixed neighbors. However, in the real world, the robots are moving and so the neighbors in the two-way ring are continuously changing. Generally, this kind of system is denoted as CML with random coupling, and investigated from the viewpoint of oscillator synchronization [31], global stability [32], bifurcations [33], and other properties. We are interested in the observation that CML can be thought of as a 2-dimensional grid, where each node is a fixed robot (agent, equation) and couplings indicate connections between these fixed nodes. When we follow this idea, the moving robots represent changing couplings. However, when we consider the nodes in this CML-grid as "place-holders" for a robot, the coupling means connections between "place-holders" and not between robots. We only require that for each "place-holder" there is a robot, when information update in CML is asynchronous (that is, a robot can connect to and disconnect from a "place-holder" at any time).

Thus, we can understand the structure of Eq. (2). The changing of robots with "place-holders" does not influence the eigenvalues and eigenvectors, but obviously will introduce additional nonlinear effects in the dynamics of CML. There are two main effects that influence these dynamics. First, as seen in swarm robotics, robots often build so-called "clusters", which are sometimes separated from the rest of a swarm [15]. Obviously, this clustering effect can be observed in CML, where we can see a separate sub-CML with its own $q_{n\to\infty}^{local}$. When the robots are moving for a long enough time (primarily defined by swarm density), they will achieve $q_{n\to\infty}^{global}$ as defined by (10). Second, the solution (10) with eigenvalues (11) represents a stable fix point. When L_j clustered, the robots achieved by p-asynchronous updates the stable fix point $q_{p\to\infty}^{local,j}$; they will retain this value so long as they do not enter into a new cluster

$$q_{p\to\infty}^{local,j} = \frac{1}{L_j}\sum_{i=1}^{L_j} q_{0,j}^i, \tag{12}$$

where j is the number of such subclusters, $j = 0 \dots K$ and K is the total number of subclusters. The values of $q_{p\to\infty}^{local,j}$ represent a local averaging of $q_{0,j}$ and generally have a random character, because subclusters are created more or less randomly. Thus, we observe two-step-dynamics with building local fix points $q_{p\to\infty}^{local,j}$ and their averaging into $q_{r\to\infty}^{global}$

$$q_{r\to\infty}^{global} = \frac{1}{K}\sum_{j=1}^{K} q_p^{local,j}, \tag{13}$$

where $r + p = n$ (n from the expression (10)). The value of $q_{r\to\infty}^{global}$ does not depend on the initial value q_0 and represents an averaging of random $q_{p\to\infty}^{local,j}$.

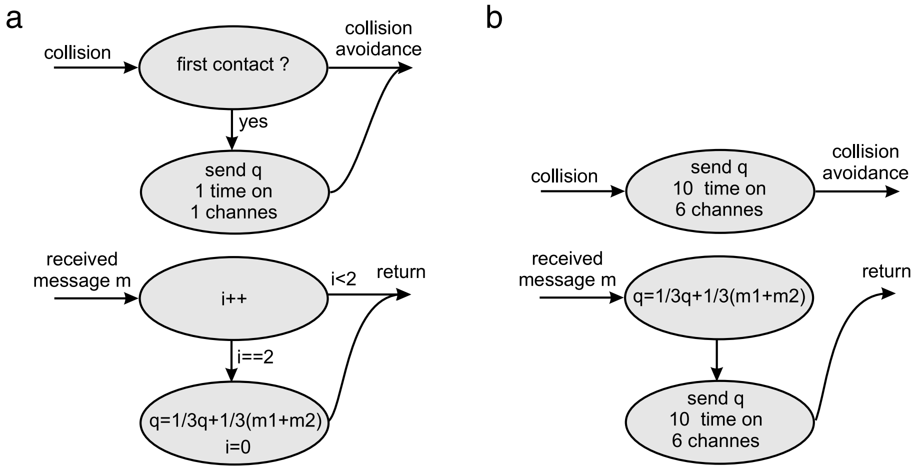


Fig. 5. (a) Slow information update, where the message triggering averaging is sent only once, on collision with another robot. (b) Fast information update, where the messages are sent continuously.

The relationship between p and r in the expressions (12) and (13) defines a dynamics of CML (2) when neighbors are randomly changed. In the case $p > r$, the dynamics are defined by the robots' behavior in clusters, whereas for $p < r$ and especially for $p \ll r$, the final result is (almost) independent of the clustering effect. While both cases have their applications in swarm robotics, in this paper we are more interested in the case when $p < r$ or $p \ll r$. This condition can be achieved when asynchronous information update in CML is slow, so that subclusters cannot be built and the $q^{\text{global}}_{r\to\infty}$ is defined mostly by q_0.

To exemplify these calculations, and before carrying out the real experiments, we simulated this behavior. The simulation of the robots and especially the robot–robot communication is implemented in "Breve" [34] and has a physical character, meaning the simulated behavior matches relatively well with the behavior of the real "Jasmine" robots [35]. For different swarm densities, which also influence the clustering effect, we simulated two cases of 50 and 100 agents in an arena of the same size. For 50 agents we expected stronger clustering effects than for 100 agents, primarily due to lower swarm density.

For both cases we implemented both fast and slow updates of information, so that we had four different situations for analysis. The algorithms for fast and slow information updates are shown in Fig. 5. The main difference between the algorithms is the number of communication messages sent. In slow update, the messages are sent only on the first collision contact. Averaging takes place only when two different messages are received. In fast information update, communication takes place continuously, not only at collision contacts but also at averaging. Moreover, a robot sends messages even when there are no collisions. We expect 3–5 times quicker information propagation in the fast than in the slow case. In the simulation, we collected the q values of all robots, plotted in Fig. 6 depending on the number of information updates performed. Since updates are asynchronous (in the same time step one robot might perform five updates, whereas another might perform 50 updates), these plots show relative time dynamics. We stop the simulation only when all the robots achieve a common value, so that eventually all the robots achieve the fix point.

In Fig. 6(a) and (b) we see the expected clustering effect. The robots build many small clusters, where they archive the stable fix points $q^{local,j}_{p\to\infty}$. The global averaging is first, relatively slow, and second, the inaccuracy of averaging is about 33%. In Fig. 6(c) and (d) we demonstrate the slow updates case. Here, there are fewer clustering effects and the accuracy of averaging is higher; the maximum mistake is about 10%. Note that the slow and fast cases differ in the number of information updates; slow update requiring about 3–5 times less communication. However, the running time (how many times the agents are running) is almost the same for both the 50 and 100 robot-cases, because the number of collision contacts primarily depends on the swarm density, the velocity of motion and other parameters of the robots.

5. Implementation and experiments

As the communication and random motion of robots in simulation and in the real swarm are implemented in different ways, we must first validate the results from the simulation. After each collision, simulated robots rotate through a random angle and move on. To generate randomness in the simulation, we use the discrete uniform distribution bounded on the region [1, 6] with constant probability at each value. Such distribution is also called discrete rectangular distribution [36]. Real robots use environmental conditions for generating randomness, see Fig. 7 and the robots can detect collisions by sensing reflected IR light. The sensing and collision avoidance radii are nonlinear functions of the surfaces, motion velocity and orientation of the robots, and vary between 8 and 15 cm. After each collision, each robot measures the reflected IR light on all six channels. The IR signal can come not only from obstacles but also from other robots, which also avoid collisions. Reflected and directed IR light can be distinguished from each other, which supports recognition of robots and passive objects.

The total level of IR signals on all six channels finally defines the areas with high and a low densities of robots and obstacles. The robot selects a direction of motion with the smallest IR signal. Thus, this environment-dependent random behavior has properties of spatial uniform distribution, that is, the robots tend to be equally distributed over the arena. We expect fewer clustering effects for this behavior than in the simulation.

When simulated robots meet each other, they exchange their values in a very short time. This results in a rapid convergence of Eq. (2). To validate this approach, we tested communication between real robots, using the setup shown in Fig. 7(b). We measured the time needed to establish bi-directional optical communication and to transmit a message from the first to the last robot [37]. The experimental data are shown in Fig. 7(c) and (d) for different distances between robots and different communication protocols. Local communication is stable within the communication range, and the propagation time increases with the number of bits per message and is almost independent of the

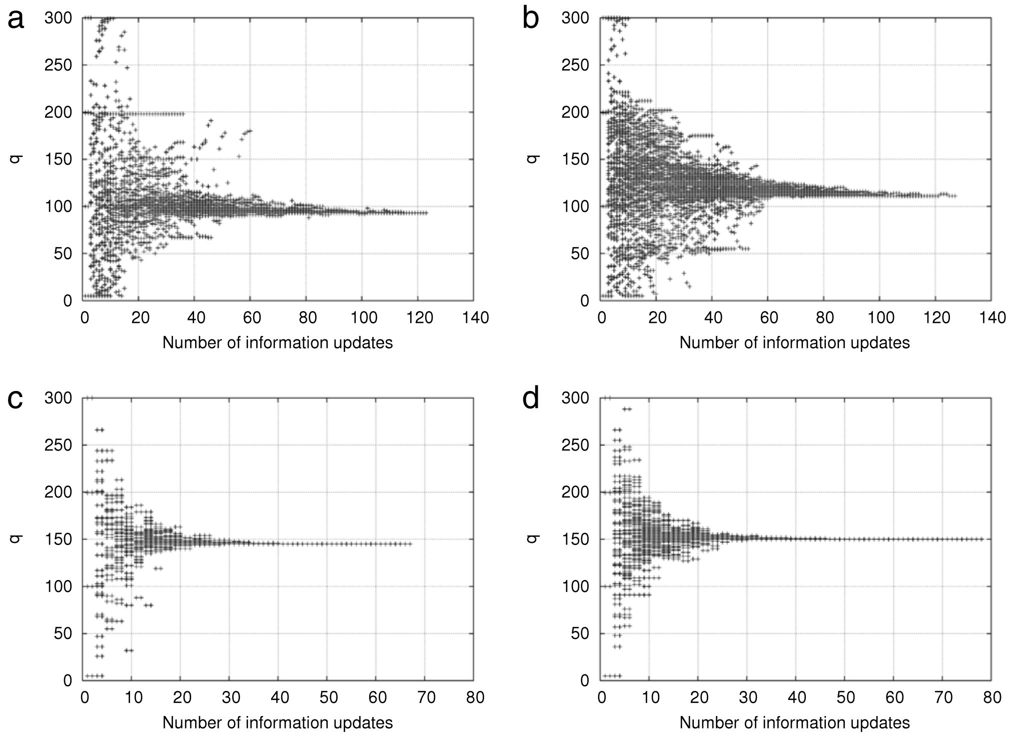


Fig. 6. Dynamics of variable q for all robots in the simulation; the expected mean value is 150. (a) 50 robots in the fast information update case; (b) 100 robots in the fast information update case; (c) 50 robots in the slow information update case; (d) 100 robots in the slow information update case.

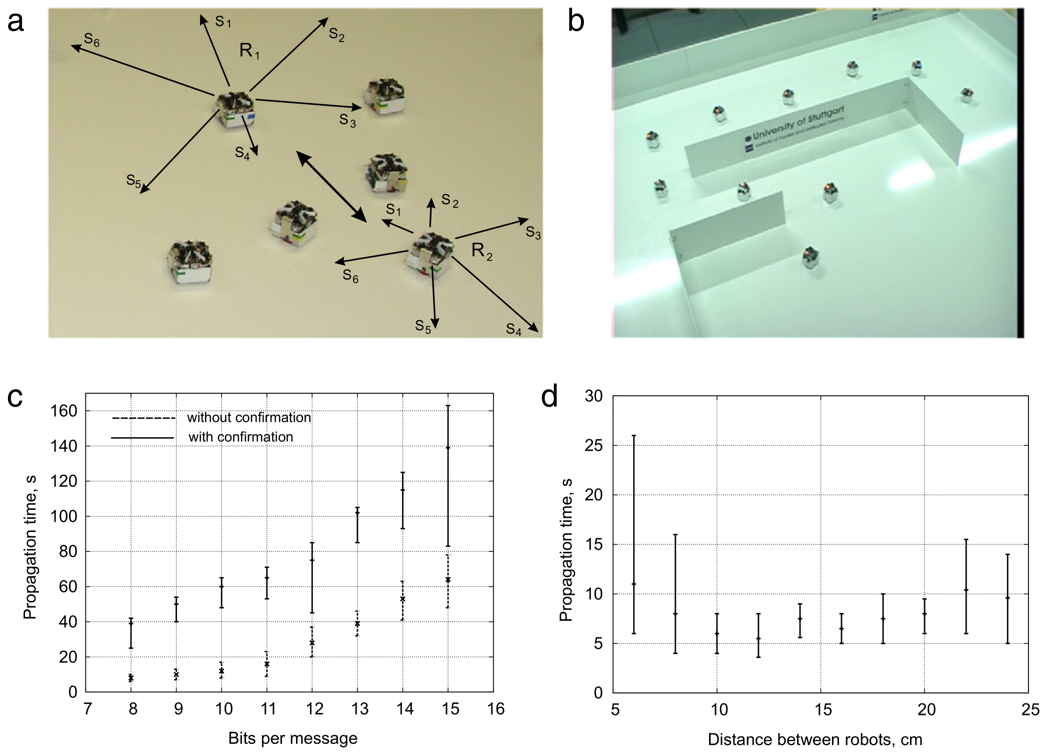
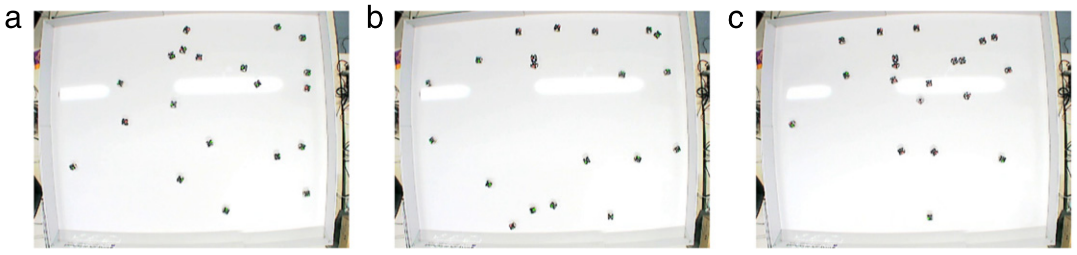
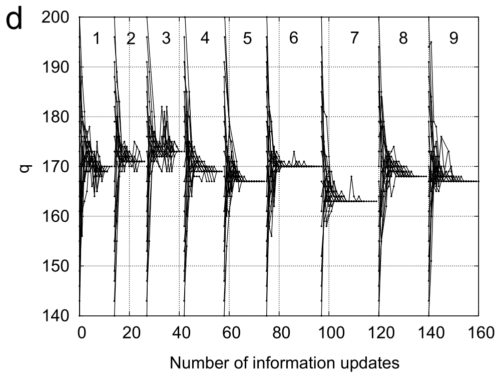


Fig. 7. (a) Collision avoidance and generation of environment-dependent random motion. Two robots R_1, R_2 perform active collision avoidance, with the robots between them representing passive objects; (b) Setup for testing optical communication between robots; (c) Dependence between the number of bits per message and the propagation time for "with confirmation" and "without confirmation" protocols (for 6 robots); (d) Dependence between the propagation time and the distance between robots (for 6 robots, 8 bit package).

distance between robots. When the wall dividing the robots is removed, local communication is distorted by the IR noise from the other robots, which increases the propagation time. This distortion can even halt communication, when the robots move. To increase stability to IR noise, robots meeting each other stop moving and repeatedly send packages until they exchange two bytes of information. We expect this will slightly slow the convergence of Eq. (2).





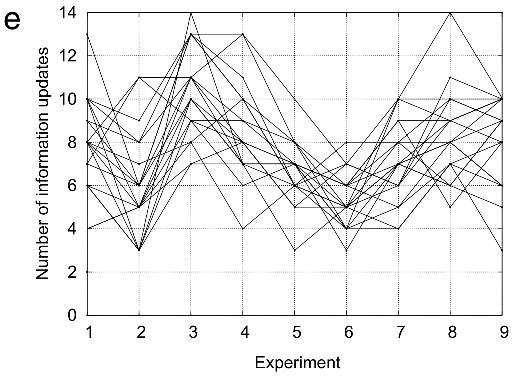


Fig. 8. (a)–(c) One run of 20 robots, images taken at 1 min intervals. (d) Plot of numerical q values from real robots. The averaged values obtained in nine experiments are 169, 171, 173, 169, 167, 170, 163, 168, 167; the average value over all experiments is 168.55, with standard deviation 2.67. (e) Number of information updates in nine experiments before all robots achieve a stable fix point.

We reproduced the simulation in Fig. 5 with 20 real robots in an arena measuring 140 × 115 cm, see Fig. 8(a)–(c). The robots use the random behavior and communication strategy previously described. All the initial values of q_i are mapped to the region [140, 200] of the energetic states from Table 1. We manually set up all the initial values of q into the robots' EEPROM, with intervals of 3; that is, the first robot receives $q_0^{(1)} = 140$, the second robot – $q_0^{(2)} = 143$, …the last robot – $q_0^{20} = 200$. Thus, we expect a value of 170 for long-term dynamics. Internally, q is declared as a float-type variable, however for communication it was coarsened to 8-bit, that is, the robots communicate integers between 0 and 255. To achieve better averaging, each robot accepts only every thirtieth communication update for the averaging procedure. Each experiment lasts around 10 min and is repeated 9 times. At each communication, the robots store the value of q_i in their RAM array. Since all data are asynchronous, each robot writes a marker (the value of 255) in its RAM array before a new run of the experiment. This enables us to synchronize data for plotting. At the end of all the runs we read these values and plot them in Fig. 8(d). The average mean value over all experiments is 168.55 and the maximum error is 4.1%. We expect that by increasing the number of robots, the error level will also increase, but qualitatively it should remain around 10% of the slow update strategy. In Fig. 8(e) we plot the number of information updates before all robots achieved the stable fix point, thus, the maximum difference between the fastest and slowest robots is 11 information updates. Since achieving the fix point is a global event for all robots, this underlies the following synchronization procedure. Experiments 1–4 are triggered manually, however, in experiments 5–9, after achieving the fix point the robots start counting 15 stable values, which are not written in the RAM array. The first robot to achieve the fifteenth value sends 255; every other robot, on receiving 255, sends it twice, resets the value of q from EEPROM and starts a new run. Thus, we observe experiments 5–9 were slightly longer, with highly-visible fix points.

Concluding these validation experiments, we observe a number of qualitative and quantitative differences between simulated and real robots, related to behavior, communication and computation capabilities. Qualitatively, the long-term dynamics of Eq. (2) in real robots is similar to simulated robots, but with two differences. First, we observed fewer clustering effects in real robots than in simulated, resulting in a lower deviation from the expected mean value. Second, the running time of real experiments is longer than simulated experiments, despite the simulator maintaining the same relationship between the speed of robots and the size of the arena. The number of information updates in Figs. 6 and 8 does not reflect their duration: the simulation finished after 3–5 min, with 50–100 robots, whereas the real experiments required 10–15 min for 20 robots. We see a need to calibrate the simulator with real-time processes for any energetic experiments.

The final experiments with collective energy foraging were repeated many times with different setups both during the original submission and the revision phase of this work. For these experiments, we used the setup shown in Fig. 9(a), with an arena measuring 140 × 115 cm containing 50 robots (46 moving robots and 4 initial communication points). Communication between the robots was carried out as described for the previous experiments. Since the swarm density was relatively high, we decreased both the collision avoidance radius of robots to 3–5 cm and the motion velocity of the robots, to avoid physical collisions at a smaller collision radius. The docking station ran at 5.5 V and used the first approach (from Section 2.1), in which each robot can become a communication point. We expect that 8–12 robots can simultaneously recharge in this setup. To obtain values for total power consumption, a digital multimeter measured the total current between the docking station and the power supply and sent the data to a computer (one value every 5 s). Each robot has an implemented individual energy homeostasis, shown in Figs. 3, and 4(b).

In the first experiment, see Fig. 10(a), all the robots were first fully charged manually and allowed to move for 50 min in the arena. Thus, all the robots were approximately 60% discharged. The robots subsequently used the egoistic greed approach to recharging: those robots which were close to the docking station docked into the free slots (the docking approach for 11 robots took

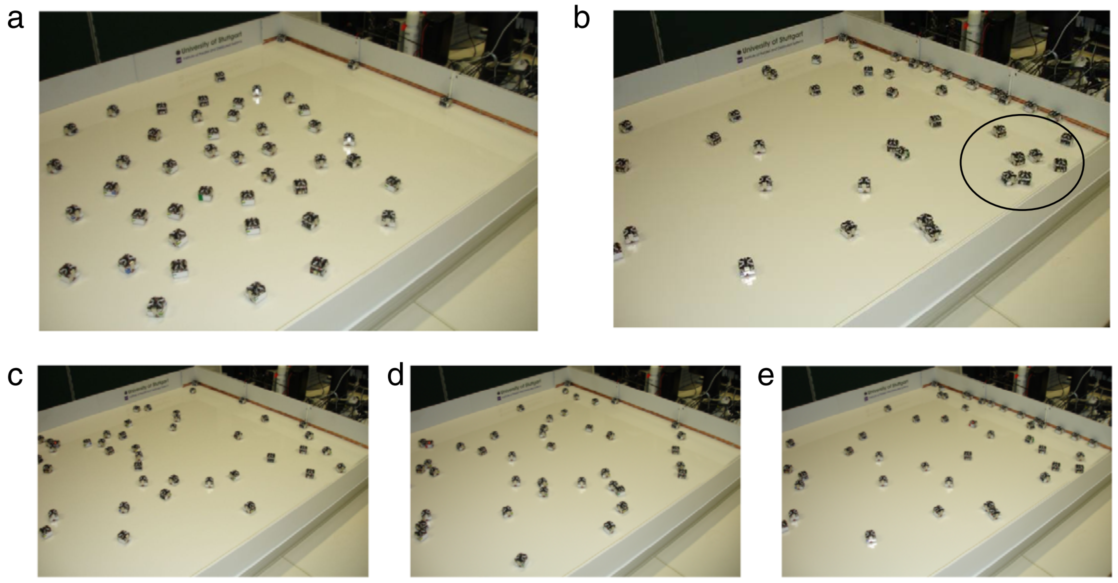


Fig. 9. Collective energy foraging in a swarm of "Jasmine" microrobots. (a) Initial set-up: the marked robots are initial communication points; (b) Final state of the experiment—the circle at the right shows a cluster of robots blocking part of the docking station; (c)–(e) Several images from a particular run of the experiment with collective decision-making.

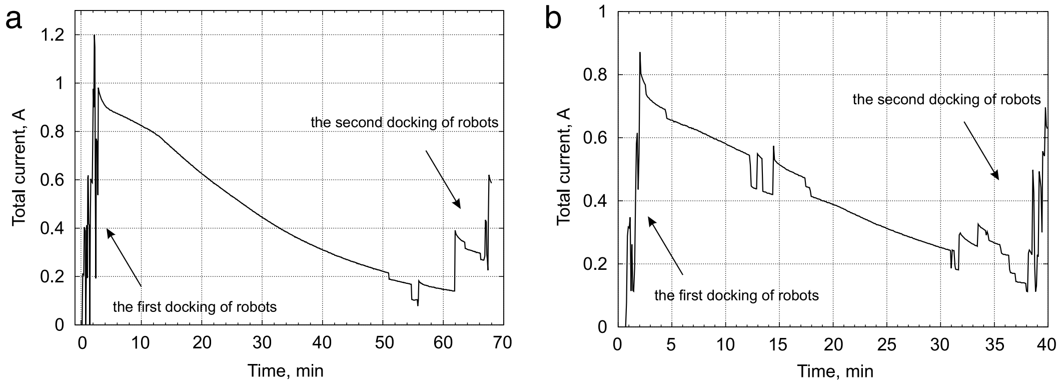


Fig. 10. Total current consumption for the greed recharging strategy without collective decision-making for (a) 60% initial discharging; (b) 30% initial discharging.

about 4–5 min) and then recharged until reaching the "full" state. All the robots from the first docking left the station between 55 and 62 min into the experiment and new robots then docked. We repeated this experiment using fully-charged robots allowed to move for 25 min (30% discharged) which then recharged using the same egoistic greed strategy (see Fig. 10(b)). In this run, we observed lower current consumption (initial 1 A vs initial 0.7 A) and shorter recharging time (62 min vs 35 min). When we run these experiments further, we observe a repetition of the qualitative pattern seen in Fig. 10(a), with large initial currents and long recharging times (maximum values are around 2 A for the initial current and about 90 min duration). In the six experiments performed (three times with 50 min of motion and 3 times with 25 min of motion), the average current for the greed strategy varies between 0.38 A and 0.47 A, see Table 2.

In conclusion, for the greed strategy, we observe a relative low efficiency, explained by low recharging current in the "satisfied" state (when the accumulator is recharged more than 80%–85%). To improve efficiency, first, the most discharged robots should arrive the docking station first (they consume a maximum current) and the duration of recharging should be limited depending on the level of collective energy E_c.

Collective decision-making addresses these two issues and is implemented thus: all robots, except for currently recharging ones, communicate and calculate the value of E_c using the expression (1). Each robot receiving the q value fixes its own $q_0 = E_c$ and calculates q using Eq. (2), using the slow update algorithm. For synchronization, we used an approach based on counting information updates after achieving a stable fix point. After 30 such updates, the robots decide whether to recharge or to continue with their task. This decision is based on the scheme shown in Fig. 4(b). Between two decision-making procedures, the robots exchange the value "255", which resets all the internal states of the robots. The first robot to send the value of q initiates a new decision-making procedure. These values are selected so that robots asynchronously start collective decision-making every 10–15 min (alternatively, a timer can reset internal states).

Individual energy consumption depends on the relationship between the time spent on the robot's different activities. To emulate different activities, and so different energy consumptions, each robot moves randomly but, during collision avoidance, waits for a time t_{wait}. Thus, the general level of energy consumption depends on t_{wait}, introducing duty cycles in useful activities. In this way, we can experimentally test swarm foraging behavior for different energy states in robots.

In the first experiment implementing collective decision-making, all the robots are first manually recharged and half of them allowed to move for 30 min, creating a swarm with different

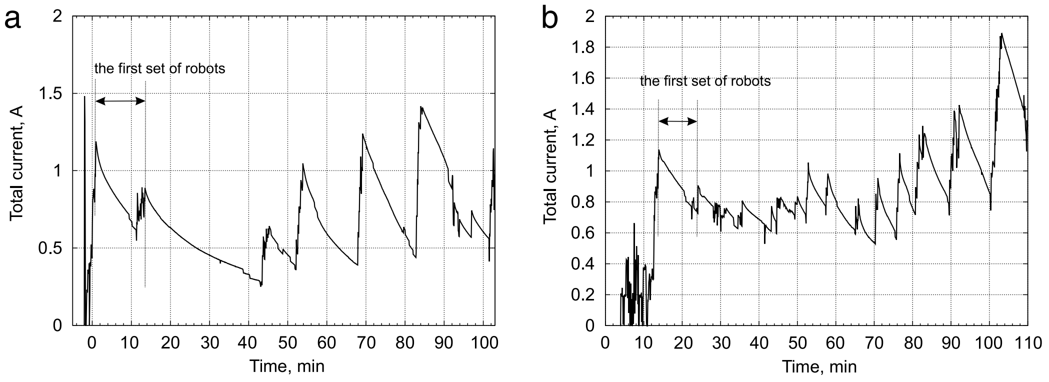


Fig. 11. Total current consumption for the recharging strategy with collective decision making for (a) the case of a high initial level of collective energy E_c^c; (b) the case of a low initial level of collective energy E_c^c.

Table 2
Parameters of the experiments: N_{robots}—the number of robots recharged at the docking station; $t_{experiment}$—duration of the experiment after the docking station is first fully occupied; $t_{firstSet}$—duration of recharging for the first set of robots (first docked robots); $\frac{N_{robots}}{t_{experiment}}$—average number of recharged robots during the whole experiment; $\frac{N_{changes}}{t_{experiment}}$—number of current changes at the docking station; $\frac{\sum(I_S)}{N_{I\text{-}samples}}$—average current, calculated as the sum of current samples (obtained from a digital multimeter) over the number of these samples. Each experiment was repeated three times, the respective values are separated by commas.

Experiment	N_{robots}	$t_{experiment}$ (min)	$t_{firstSet}$ (min)	$\frac{N_{robots}}{t_{experiment}}$	$\frac{N_{changes}}{t_{experiment}}$	$\frac{\sum(I_S)}{N_{I\text{-}samples}}$ (A)
1. Fig. 10(a)	11, 10, 12	62, 60, 65	62, 60, 65	0.17, 0.16, 0.18	0.01, 0.01, 0.01	0.45, 0.40, 0.48
2. Fig. 10(b)	11, 8, 11	35, 31, 37	35, 31, 37	0.31, 0.25, 0.29	0.03, 0.03, 0.03	0.41, 0.37, 0.42
3. Fig. 11(a)	33, 35, 29	100, 100, 100	14, 12, 17	0.33, 0.35, 0.29	0.07, 0.07, 0.06	0.71, 0.75, 0.68
4. Fig. 11(b)	45, 47, 42	100, 100, 100	10, 12, 9	0.45, 0.47, 0.42	0.15, 0.17, 0.14	0.85, 0.90, 0.81

energy levels and a high E_c^c. When the common energy level is high, the low-energy robots can be the first to recharge and can recharge for longer. In Fig. 11(a) we see this behavior. After the first robots to recharge leave the docking station, new robots can begin recharging. This experiment lasted 100 min and we observed a slight increase in total energy consumption.

In the second experiment, after manual recharging, all the robots were allowed to move for 30 min and the common energy level E_c^c was low. The robots recharged for a short time, to take a maximum amount of energy, see Fig. 11(b). Changes of robots in the docking station were more frequent and we observed more distortions in the recharging process. Since the initial energy levels were low, the robots that recharged later became more and more discharged. When they docked, we observed an increasing total current over time. Moreover, all the later-recharging robots needed more time to recharge, to least to the satisfactory level, in contrast to the first robots, which recharged more quickly. We observed that several robots become energetically dead.

These experiments were repeated three times; Table 2 shows several parameters for these experiments. The two most significant parameters represent the duration of recharging for the first set of robots (first docked robots) $t_{firstSet}$ and the average current $\frac{\sum(S)}{N_{samples}}$ flowing through the docking station throughout the experiment. The value of $t_{firstSet}$ indicates the selected recharging strategy: long times mean more egoistic behavior, short times mean more social behavior. We see that collective decision-making primarily balances collective and individual needs by shortening the recharging time. The value of the average current $\frac{\sum(I_S)}{N_{I\text{-}samples}}$ shows the energy income of the robot swarm. As we see, collective decision-making allows an almost doubling of the energy income, see Fig. 12(a).

Analyzing the results of these experiments, we see that the limited energy income in the swarm is not enough for the long-term powering of all robots. For instance, the last experiment demonstrated the best value of $I_{income} = 0.9$ A, whereas we estimate for $t_{wait} = 0$ about $I_{consumption} = 4.6$ A for 46 robots. Thus, we expect energetically dead robots to appear until both values become balanced, that is, $I_{income} = I_{consumption}$. To avoid energetic death, the robots should monitor the value E_c^c and, on the continuous decreasing of collective energy, should increase the duty cycles by changing t_{wait}, that is, the robots should move less to save energy. This scheme can represent a global self-regulation mechanism. We implemented the simplest version of this approach with $t_{wait1} = 0$ and $t_{wait2} = 10$ s and switching between both values at the fixed threshold $E_c^c = 150$ (that is, the critical state). This experiment lasted three hours (with 30 min of movement of the robots before the start of the experiment). The total current consumption is shown in Fig. 12(b), where we can see at least two switching points of t_{wait}. During these 270 min we saw no energetically dead robots.

One run of this experiment, performed during the revision of this paper, lasted for around 26 h (from 18:00 on day one until around 20:00 the next day). We expected to test the possibility of long-term autonomy, to discover hardware implications for long-term behavior and to explore the number of dead robots as the experiment progressed. First, we encountered a relatively high number of energetically dead robots. With an average capacity of the docking station of 10 robots, we expected that all 46 robots could have enough energetic resources to survive for a long time with 75% of duty cycles. However, between 4 and 12 h after the experiment began, more than half of the robots had become completely discharged. At the end of the experiment, only 9 robots survived. Of the dead robots, we noted that the open-geared motor-wheel coupling of 5 robots was blocked by dust collected during movement; 3 robots had broken docking contacts; 1 robot had a defective fuse on its Li–Po accumulator and 1 robot had a deeply discharged accumulator. Unfortunately, three of the broken robots were close to the docking station, creating a cluster of energetically dead robots (as in Fig. 9(b)), which eventually blocked the left half of the docking station and decreased the recharging capability of the station.

6. Conclusions

In this paper, we have suggested approaches for maintaining energetic homeostasis in a robot swarm. This approach includes

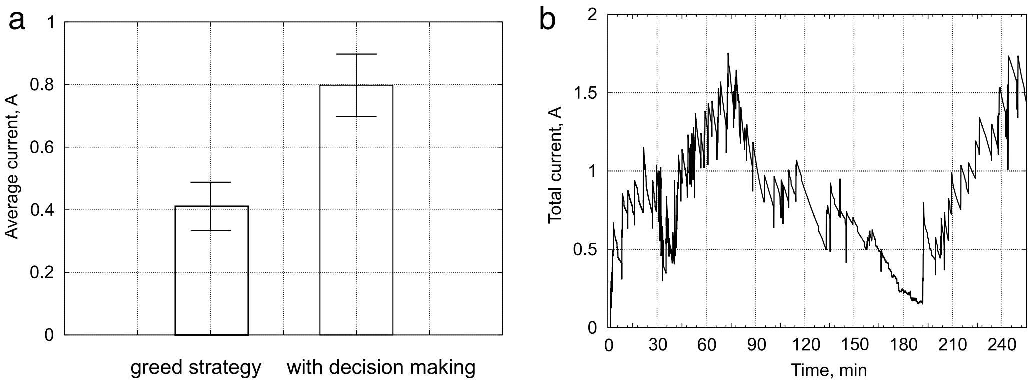


Fig. 12. (a) Average current for the greed strategy and for the decision-making approach;(b) Total current consumption for the recharging strategy with collective decision-making and with global self-regulation by changing t_{wait}.

hardware and software mechanisms at the individual robotic level, as well as at the collective level of the whole swarm. We have demonstrated that collective values, such as the collective energy level, can be useful in adapting individual behavior to collective needs and for increasing energetic efficiency. In particular, the value of collective energy can be used either for deciding on priority and duration of individual recharging, or for global self-regulation by adjusting individual duty cycles. The goal of collective regulation is to support the efficient use of limited recharging stations, to adapt robot swarms to the energetic conditions in their environment [38], and to avoid energetic death of robots with critically low energy.

Another issue is the optimal foraging strategy at the individual as well as the collective levels. In this paper, we did not investigate the possibilities of optimizing collective values and foraging behavior thus created, especially for specialized tasks (where the priorities of "looking for food" and "doing a job" can be changed). In [11], we offered several ideas for such optimization by mixing behavioral strategies, but this area represents a place for further research.

Regarding long-term self-regulation, which was very briefly shown in the final experiment, we faced several problems unrelated to the robots' behavior and which need to be taken into account in the design of long-term experiments for more complex reconfigurable systems [39]. First, for long-term autonomy, a robust mechatronic design must be created [40]. Issues such as wheels blocked by dust are negligible in a single experiment with human assistance, but become critical in longer-term experiments without human maintenance. Second, the area around the docking station is a bottleneck in the system. When this area is blocked (for example by waiting or dead robots), the docking station cannot be used to full capacity. The best strategy is to install docking stations on all the walls of the arena. Thus, we indirectly confirm the statement in [20]: for optimal foraging, energetic resources should be uniformly distributed around the arena.

During the development of these experiments, we encountered a few issues that can extend the boundary of robotic research. First is the "free will" required to survive due to individual energetic homeostasis. Robots can stop the task they are undertaking, because they are "hungry", and start looking for a "food place". This radically differs from the "industrial approach" [41,42], in which the robot's freedoms are limited. "Free will" can be influenced by the respective priorities of "hungry" states and collective tasks. However, "free will" can also cause undesired effects at the collective level and should be investigated in more detail.

Generally, we have also shown that the implementation of autonomous homeostasis in microrobots leads to very interesting behaviors with many analogies to biological swarm systems. The robots collectively adapt to the energetic situation. For instance, they build a "buffer" before the docking station, in which the number of robots in the "buffer" depends on the relationship between the collective energy consumption and capabilities of the docking station. This is an emergent phenomenon [43], not programmed into the robots' behavior and to some extent, demonstrates an emergent self-regulating mechanism [44]. In critical situations, robots in the "buffer" compete for a free slot in the docking station. These analogies offer not only the opportunity for better design of robotic systems, but also a deeper understanding of biological organisms and the phenomenon of collective intelligence.

Acknowledgments

We thank our students Kristof Jebens [22], Dagmar Hábe [45], Afshin Attarzadeh [23], Tanya Kancheva [24], Victor Prieto Martinez [35], Zheng Fu [37] and others for active participation in the experiments performed. The authors are supported by the following grants:EU-IST FET project "I-SWARM", grant agreement no. 507006; EU-ICT FET project "SYMBRION", grant agreement no. 216342; EU-ICT project "REPLICATOR", grant agreement no. 216240. Additionally, we want to thank all members of the projects for fruitful discussions.

References

[1] M. Neal, J. Timmis, Once more unto the breach: towards artificial homeostasis, in: Recent Developments in Biologically Inspired Computing, Idea Group Publishing, 2005, pp. 340–365.
[2] D. Stephens, J. Krebs, Foraging Theory, Princeton University Press, 1987.
[3] A. Seth, Modelling group foraging: individual suboptimality, interference, and a kind of matching, Adaptive Behavior 9 (2002) 67–91.
[4] N. Owens, J. Timmis, A. Greensted, A. Tyrrell, On Immune Inspired Homeostasis for Electronic Systems, in: LNCS, vol. 4628, Springer, 2007, pp. 216–227.
[5] I-Swarm, I-Swarm: Intelligent small world autonomous robots for micro-manipulation, in: European Union 6th Framework Programme Project No. FP6-2002-IST-1, 2003–2007.
[6] M. Silverman, D. Nies, B. Jung, G. Sukhatme, Staying alive: a docking station for autonomous robot recharging, in: Proc. of the IEEE International Conference on Robotics and Automation, Washington, DC, 2002, pp. 1050–1055.
[7] F. Sempé, A. Muńoz, A. Drogoul, Autonomous robots sharing a charging station with no communication: a case study, in: H. Asama (Ed.), Distributed Autonomous Robotic Systems, MIT Press, Cambridge, MA, USA, 2002, pp. 91–100.
[8] B.E. Logan, B. Hamelers, R. Rozendal, U. Schröder, J. Keller, S. Freguia, P. Aelterman, W. Verstraete, K. Rabaey, Microbial fuel cells: methodology and technology, Environmental Science & Technology 40 (17) (2006) 5181–5192.
[9] S.P. Beeby, M.J. Tudor, N.M. White, Energy harvesting vibration sources for microsystems applications, Measurement Science and Technology 17 (12) (2006) R175.
[10] W. Liu, A. Winfield, J. Sa, A macroscopic probabilistic model of adaptive foraging in swarm robotics systems, in: Proc. of 6th Vienna International Conference on Mathematical Modelling, Mathmod 2009, 2009.
[11] S. Kernbach, V. Nepomnyashchikh, T. Kancheva, O. Kernbach, Specialization and generalization of robotic behavior in swarm energy foraging, Mathematical and Computer Modelling of Dynamical Systems (2011) doi:10.1080/13873954.2011.601421.

[12] S. Kernbach (Ed.), Handbook of Collective Robotics: Fundamentals and Challenges, Pan Stanford Publishing, Singapore, 2011.
[13] S. Kornienko, O. Kornienko, P. Levi, Generation of desired emergent behavior in swarm of micro-robots, in: R. Lopez de Mantaras, L. Saitta (Eds.), Proc. of the 16th European Conference on Artificial Intelligence, ECAI 2004, IOS Press, Valencia, Spain, Amsterdam, 2004, pp. 239–243.
[14] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, New York, 1999.
[15] S. Kernbach, R. Thenius, O. Kernbach, T. Schmickl, Re-embodiment of honeybee aggregation behavior in artificial micro-robotic system, Adaptive Behavior 17 (3) (2009) 237–259.
[16] S. Camazine, J.-L. Deneubourg, N. Franks, J. Sneyd, G. Theraulaz, E. Bonabeau, Self-Organization in Biological Systems, Princeton University Press, Princeton, NJ, USA, 2003.
[17] S. Kornienko, O. Kornienko, P. Levi, Minimalistic approach towards communication and perception in microrobotic swarms, in: Proc. of the International Conference on Intelligent Robots and Systems, IROS-2005, Edmonton, Canada, 2005, 2228–2234.
[18] S. Kornienko, O. Kornienko, P. Levi, Collective AI: context awareness via communication, in: Proc. of the IJCAI 2005, Edinburgh, UK, 2005, pp. 1464–1470.
[19] O. Kernbach, The synergetic approach towards analysing and controlling the collective phenomena in multi-agents systems, Ph.D. Thesis, University of Stuttgart, 2011.
[20] A.F.T. Winfield, S. Kernbach, T. Schmickl, Collective foraging: cleaning, energy harvesting and trophallaxis, in: S. Kernbach (Ed.), Handbook of Collective Robotics: Fundamentals and Challenges, Pan Stanford Publishing, Singapore, 2011, pp. 257–300.
[21] S. Kernbach, Encoder-free odometrical system for autonomous microrobots, Mechatronics (2011) (in press).
[22] K. Jebens, Development of a docking approach for autonomous recharging system for micro-robot Jasmine, Studient Thesis, University of Stuttgart, Germany, 2006.
[23] A. Attarzadeh, Development of advanced power management for autonomous micro-robots, Master Thesis, University of Stuttgart, Germany, 2006.
[24] T. Kancheva, Adaptive role dynamics in energy foraging behavior of a real micro-robotic swarm, Master Thesis, University of Stuttgart, Germany, 2007.
[25] G. Weiss (Ed.), Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence, MIT Press, 1999.
[26] O. Kornienko, S. Kornienko, P. Levi, Collective decision making using natural self-organization in distributed systems, in: Proc. of Int. Conf. on Computational Intelligence for Modelling, Control and Automation, CIMCA'2001, Las Vegas, USA, 2001, pp. 460–471.
[27] G. Coulouris, J. Dollimore, T. Kindberg, Distributed Systems, Addison-Wesley, Longman, Amsterdam, 2001.
[28] K. Kaneko, Theory and Application of Coupled Map Lattices, John Willey & Sons, Chichester, New York, Brisbane, Toronto, Singapore, 1993.
[29] U. Krause, T. Nesemann, Differenzegleichungen und Diskrete Dynamische Systeme, B.G. Teubner Stuttgart, Leipzig, 1999.
[30] R. Grigoriev, M. Cross, H. Schuster, Pinning control of spatiotemporal chaos, Physical Review Letters 79 (15) (1997) 2795–2798.
[31] S. Sinha, Random coupling of chaotic maps leads to spatiotemporal synchronization, Physical Review E 66 (2002) 016209-1–016209-9.
[32] H. Atmanspachera, H. Scheingraber, Stabilization of causally and non-causally coupled map lattices, Physica A 345 (2005) 435–447.
[33] P. Levi, M. Schanz, S. Kornienko, O. Kornienko, Application of order parameter equation for the analysis and the control of nonlinear time discrete dynamical systems, International Journal of Bifurcation and Chaos 9 (8) (1999) 1619–1634.
[34] Breve: a 3D simulation environment for multi-agent simulations and artificial life. http://www.spiderland.org/breve/.
[35] V. Prieto, Development of cooperative behavioural patterns for swarm robotic scenarios, Master Thesis, University of Stuttgart, Germany, 2006.
[36] N.L. Johnson, A.W. Kemp, S. Kotz, Univariate Discrete Distributions, John Wiley & Sons, 2005.
[37] Z. Fu, Swarm-based computation and spatial decision making, Master Thesis, University of Stuttgart, Germany, 2005.
[38] S. Kernbach, E. Meister, F. Schlachter, O. Kernbach, Adaptation and self-adaptation of developmental multi-robot systems, International Journal on Advances in Intelligent Systems 3 (1–2) (2010) 121–140.
[39] P. Levi, S. Kernbach (Eds.), Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution, Springer Verlag, 2010.
[40] S. Kernbach, E. Meister, F. Schlachter, K. Jebens, M. Szymanski, J. Liedke, D. Laneri, L. Winkler, T. Schmickl, R. Thenius, P. Corradi, L. Ricotti, Symbiotic robot organisms: REPLICATOR and SYMBRION projects, in: Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems, PerMIS'08, ACM, New York, NY, USA, 2008, pp. 62–69.
[41] S. Kornienko, O. Kornienko, J. Priese, Application of multi-agent planning to the assignment problem, Computers in Industry 54 (3) (2004) 273–290.
[42] S. Kernbach, Towards application of collective robotics in industrial environment, in: G. Rigatos (Ed.), Industrial Systems: Modelling, Automation and Adaptive Behaviour, IGI Global, 2010, pp. 18–49.
[43] S. Kornienko, O. Kornienko, P. Levi, About nature of emergent behavior in micro-systems, in: Proc. of the Int. Conf. on Informatics in Control, Automation and Robotics, ICINCO 2004, Setubal, Portugal, 2004, pp. 33–40.
[44] S. Kernbach, Structural Self-Organization in Multi-Agents and Multi-Robotic Systems, Logos Verlag, Berlin, 2008.
[45] D. Häbe, Bio-inspired approach towards collective decision making in robotic swarms, Master Thesis, University of Stuttgart, Germany, 2007.

Serge Kernbach is a head of collective robotics group at the University of Stuttgart, Germany. He graduated in electronic engineering and computer science in 1994. During 1996–1998 he received several fellowships; he was a guest scientist in the center of Synergetics led by Prof. H. Haken; in 2007 his doctoral thesis won the faculty-award as the best dissertation of the year. Since 2004 he has been a coordinator of several European research projects on the field of collective robotics. Serge's main research interest is focused on artificial collective systems, he is an author and co-author of over 100 articles in international journals and conferences and he edited a few books related to robotics.

Olga Kernbach received her B.S. degree at the Moscow State Technical University in 1991 and, in 1994, she graduated in computer science at the Taganrog State University, Russia with an honorary diploma. In 1996, she received Russia's presidential research award and a research grant from the German academic exchange service. In 1997, she received a second research grant for scientific work in the Center of Synergetics, headed by Prof. H. Haken. Currently, she is a research assistant at the University of Stuttgart. Olga's main research interests include distributed AI, co-operative problem solving and decision making in swarm systems, and their applications in robotics and manufacturing.