Article

The International Journal of Robotics Research
1–17

DOI: 10.1177/0278364914525244
ijr.sagepub.com

Self-organized aggregation without computation

Melvin Gauci, Jianing Chen, Wei Li, Tony J. Dodd and Roderich Groß

Abstract

This paper presents a solution to the problem of self-organized aggregation of embodied robots that requires no arithmetic computation. The robots have no memory and are equipped with one binary sensor, which informs them whether or not there is another robot in their line of sight. It is proven that the sensor needs to have a sufficiently long range; otherwise aggregation cannot be guaranteed, irrespective of the controller used. The optimal controller is found by performing a grid search over the space of all possible controllers. With this controller, robots rotate on the spot when they perceive another robot, and move backwards along a circular trajectory otherwise. This controller is proven to always aggregate two simultaneously moving robots in finite time, an upper bound for which is provided. Simulations show that the controller also aggregates at least 1000 robots into a single cluster consistently. Moreover, in 30 experiments with 40 physical e-puck robots, 98.6% of the robots aggregated into one cluster. The results obtained have profound implications for the implementation of multi-robot systems at scales where conventional approaches to sensing and information processing are no longer applicable.

Keywords

Aggregation, line-of-sight sensor, minimal information processing, mobile and distributed robotics, swarm intelligence

1. Introduction

Many studies in distributed robotics have investigated systems in which the robots only make use of information from their immediate surroundings. This "principle of locality" has come to define a paradigm in the design of systems with decentralized control (Parker, 2008), and is currently the primary approach in the field of swarm robotics (Brambilla et al., 2013). When using sensors of a limited range, robots can only learn about their environment gradually, and as Brooks (1991) argues, "when intelligence is approached in an incremental manner …, reliance on representation disappears." Without such a reliance, a system is more likely to be adaptable to changing environments.

Using a limited sensing range is also widely believed to help relax the hardware requirements of the individual robots. This paper investigates an alternative approach, and reveals a surprising result: by designing robots that only use a minimal amount of unrestricted range information, the need for information processing—within each robot and among the robots—can be immensely reduced. Moreover, the non-reliance on a global representation remains intact. The approach of using a minimal amount of unrestricted range information also brings about other advantages, namely: (a) the fact that only a minimal amount of features need to be extracted from the environment allows for system designs that can be transferred from simulation to reality with minimal effort (Jakobi, 1997); (b) using information that is not restricted in range allows the system to solve problems that are otherwise unsolvable (see Litovsky et al. (1993), referred to in Klavins (2007)).

In the system presented in this paper,[1] the complexity and amount of information processing is reduced to the extreme that

- each robot only needs one sensor that provides it with one bit of information about its environment;
- the robots do not need to store any information during run time (and have no IDs);
- the robots are not required to perform any arithmetic computations.

Sheffield Centre for Robotics and Department of Automatic Control and Systems Engineering, The University of Sheffield, UK

Corresponding author:
Roderich Groß, Sheffield Centre for Robotics and Department of Automatic Control and Systems Engineering, The University of Sheffield, Pam Liversidge Building, Mappin Street, Sheffield, S1 3JD, UK.
Email: r.gross@sheffield.ac.uk

Despite all these limitations, as shown in this paper, it is possible to solve canonical problems in distributed robotics. This paper uses the task of self-organized aggregation (also known as "gathering" (Cieliebak et al., 2003; Gordon et al., 2004; Flocchini et al., 2005; Fatès, 2010) and "rendezvous" (Cortés et al., 2006; Zebrowski et al., 2007; Yu et al., 2012)) as a case study in order to illustrate the proposed methodology. The problem of aggregating robots with limited information is a challenging one, because the robots, if not controlled properly, may end up forming separate clusters. The solution presented here is the simplest one that has been presented so far, in terms of both sensor and controller complexity. Each robot can only tell whether or not there is another robot in its line of sight (in principle, this sensor can be implemented as a single pixel CCD or CMOS sensor). Furthermore, this simple sensing scheme does not come at the cost of a complex controller. Rather, it turns out that any possible controller is equivalent to an *"if-then-else"* statement, and this is sufficient to solve the task. The controller simply chooses one of two predefined sets of actuation parameters for the robot, according to the sensor reading.

We believe that it is the right time to study the collective capabilities of systems of robots that have severely limited computational and memory resources, and that are only able to retrieve a very limited amount of information about their environment (in our case, a single bit). This truly minimalistic approach helps to pave the way for implementing massively distributed robotic systems at scales where conventional approaches to sensing and information processing are no longer applicable (Requicha, 2003).

This paper is organized as follows. Section 2 presents related work. Section 3 presents the methods used, including the problem definition, the performance metrics, and the robotic and simulation platforms. Section 4 details the structure of the controller and shows that, if the range of the sensor is limited, it is not possible to guarantee aggregation, irrespective of the controller used. It also presents the method used for obtaining the optimal controller, along with its results. Section 5 provides a mathematical analysis of the optimal controller, including a proof and a time upper bound for the case of two simultaneously moving robots. Section 6 presents a number of studies in simulation, including an analysis of the effects of the range of the sensor and noise on the sensor, as well as a scalability study with up to 1000 robots. Section 7 discusses the implementation of the solution on 40 physical e-puck robots, and presents the results obtained from a systematic experiment on this system. Section 8 concludes the paper.

2. Related work

This paper uses the task of self-organized aggregation as a case study in order to illustrate the proposed methodology. Trianni (2008) argues that, in distributed robotic systems, "aggregation is of particular interest since it stands as a prerequisite for other forms of cooperation." This section discusses how aggregation occurs in nature, and how it has been implemented on distributed robotic systems so far.

2.1. Aggregation in nature

Self-organized aggregation is a widely observed phenomenon in nature. It occurs in a range of organisms, including bacteria, arthropods, fish and mammals (Parrish and Edelstein-Keshet, 1999; Camazine et al., 2001). In some cases, self-organized aggregation is aided by environmental heterogeneities, such as areas providing shelter or thermal energy (Halloy et al., 2007; Kernbach et al., 2009; Fatès, 2010). However, aggregation can also occur in homogeneous environments, purely as a result of interactions between the individuals. For example, Deneubourg et al. (1990) observed that bark beetle larvae aggregate in homogeneous Petri dishes by using a mechanism based on pheromone secretion and detection.

2.2. Probabilistic algorithms

Jeanson et al. (2005) investigated aggregation in cockroach larvae and developed a model of their behavior. The cockroaches were reported to join and leave clusters with probabilities that depend on the sizes of the clusters. The larger a cluster is, the more likely a cockroach is to join it, and the less likely it is to leave it, and vice versa. Several works have implemented algorithms based on this model for self-organized robot aggregation (Soysal and Şahin, 2005; Garnier et al., 2008; Bayindir and Şahin, 2009). Perhaps most notably, the work of Garnier et al. (2008) demonstrated aggregation with 20 physical Alice robots in a homogeneous environment.

Probabilistic aggregation algorithms have the advantage that, as long as the environment is bounded, it is not required to assume that the robots initially form a connected graph in terms of their sensing and/or communication ranges. Nevertheless, by analyzing a model similar to the one in Jeanson et al. (2005) and Garnier et al. (2008), Correll and Martinoli (2011) showed that the "robots [still] need a minimum combination of communication range and locomotion speed in order to aggregate into a single cluster when using probabilistic aggregation rules."

The algorithms in this category require the agents to obtain estimates of cluster sizes or robot densities. In order to meet this requirement, Garnier et al. (2008) used local infra-red communication to estimate the number of nearby robots. Bayindir and Şahin (2009) showed how the cluster size can be estimated from its area using circle packing theory. As an alternative to explicitly estimating cluster sizes, Soysal and Şahin (2005) proposed a variation on the probabilistic finite state machine method of aggregation, where each robot emits a sound, and a robot, when in the "approach" state, moves in the direction of the maximum sound it perceives. Kernbach et al. (2013) proposed

another method of bypassing the direct measurement of cluster sizes, based on a weakly correlated random walk in a heterogeneous environment. In this method, a robot, upon meeting another robot, decides on a time to wait before moving on, based on an activation function.

The algorithms discussed in this section all assume arithmetic computations on the part of the robots, in order to calculate the probabilities governing their joining and leaving of clusters. A particularly challenging aspect of implementing them on physical robotic systems lies in the estimation of cluster sizes and/or robot densities.

2.3. Deterministic algorithms

Ando et al. (1999) introduced an algorithm for achieving aggregation in a group of mobile robots with a limited sensing range. Cortés et al. (2006) generalized this algorithm and showed that it can be used to achieve aggregation in arbitrarily high dimensions. These algorithms require that the robots initially form a connected visibility graph, and are based on geometrically ensuring that this graph is maintained in every time step. The robots are required to measure the relative positions (distances and angles) of all their neighbors. Gordon et al. (2004) relaxed this requirement, such that the robots only need to measure the angles to their neighbors, and not the distances. Although the algorithm was theoretically proven to work, simulation results revealed that "the merging process [was] generally agonizingly slow" (Gordon et al., 2004). In order to address this problem, in follow up work, Gordon et al. (2008) introduced an additional crude distance sensing capability to the algorithm in Gordon et al. (2004), where the robots could discriminate between closer and further neighbors.

Gennaro and Jadbabie (2006) developed an aggregation algorithm based on every robot computing the Laplacian matrix of its neighbors. Gasparri et al. (2012b) developed an algorithm for aggregation based on an attractive/repulsive force model that was introduced by Gazy and Passino (2003), and generalized further by Li (2008). This algorithm was further developed to handle actuator saturation by Gasparri et al. (2012a), and to cope with obstacles by Leccese et al. (2012).

The algorithms presented in this section so far all require the robots to perform arithmetic computations, based on the positions and/or orientations of perceived robots. To our knowledge, none of these algorithms have been evaluated on physical robots, with the exception of (Gasparri et al., 2012b,a; Leccese et al., 2012). These algorithms were implemented on five SAETTA robots, but the sensing was performed in a centralized fashion, with an external camera being used to measure the distances between the robots.

Yu et al. (2012) proposed a connectivity-based algorithm for Dubins-type vehicles that, as this paper, is in the spirit of minimalism. Their algorithm assumes relatively little sensing and control capabilities on the part of the agents; however, for the algorithm to work, each agent has to either be able to track a specific agent that it is assigned to, or at least choose the closest one from a number of agents that are within its "wind shield". Furthermore, the algorithm requires the robots to synchronize their control when they are in proximity, and it has only been validated in simulation with point-robots.

2.4. Evolutionary approaches

Dorigo et al. (2004) addressed the problem of aggregation by using an evolutionary robotics approach (Nolfi and Floreano, 2000) in order to design a deterministic control algorithm. In their system, the robots can emit a sound and can sense each other using proximity sensors and directional microphones. A controller, in the form of a feedforward neural network, was evolved and validated in simulation with up to 40 robots. Bahceci and Şahin (2005) used a similar setup and investigated the effects of several parameters, such as the number of robots, arena size, and run time.

These controllers all rely on memory and/or arithmetic computation. They have all been evolved and evaluated with simulated embodied robots. The feasibility of implementing them on physical robotic systems remains an open question.

3. Methods

3.1. Problem formulation

The problem that we address here is as follows. An unbounded, homogeneous 2-D environment, with no obstacles, contains a number, n, of robots, that are initially placed in arbitrary positions, facing in arbitrary directions. Here, we consider robots that are circular and differential wheeled.[2] The robots are memoryless (in the sense that they cannot store any information during run time) and all execute the same controller. They are also embodied, meaning that they cannot overlap with each other.

Each robot is equipped with one sensor at its front, which allows it to know whether or not there is another robot in its direct line of sight. This sensor is therefore a binary line-of-sight sensor. We say that the sensor gives a reading of $I = 1$, if there is a robot in the line of sight, and a reading of $I = 0$ otherwise. It does not provide the distance to the robot being perceived.

The objective is to aggregate the robots as quickly as possible, via decentralized control, into one cluster that is as compact as possible.

3.2. Performance metrics

3.2.1. Dispersion metric. Following Graham and Sloane (1990), we use the second moment of the robots as a measure of their dispersion, which we want to minimize. Let r represent the radius of one robot. Let $\mathbf{p}_i^{(t)}$ represent the position of robot i at time t, and let $\bar{\mathbf{p}}^{(t)} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{p}_i^{(t)}$ represent

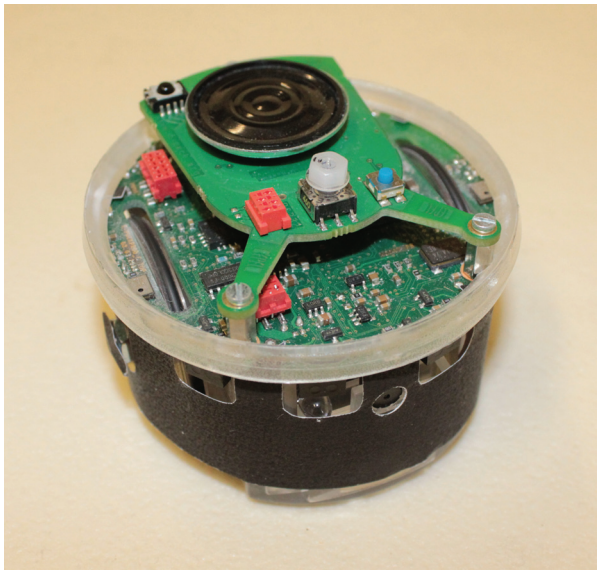

Fig. 1. The e-puck robot fitted with a black 'skirt'. The e-puck's diameter and height are approximately 7.4 cm and 5.5 cm, respectively.

the centroid of the positions of the robots. Then, the second moment of the robots is given by:

$$u^{(t)} = \frac{1}{4r^2} \sum_{i=1}^{n} ||\mathbf{p}_i^{(t)} - \bar{\mathbf{p}}^{(t)}||^2 \quad (1)$$

The $4r^2$ in the denominator serves to normalize $u^{(t)}$ such that it becomes independent of r for geometrically similar configurations. $u^{(t)}$ does not have an upper bound, because the robots can be arbitrarily dispersed. It has a positive lower bound, because of the physical constraint that the robots cannot overlap with each other, i.e. $||\mathbf{p}_j^{(t)} - \mathbf{p}_i^{(t)}|| \geq 2r$, $i \neq j$. Graham and Sloane (1990) report lower bounds of $u^{(t)}$ for several values of n up to $n = 499$. Except for $n = 212$, the packings corresponding to the lower bounds of $u^{(t)}$ are optimal among hexagonal packings (Chow, 1995).

3.2.2. Cluster metric. Let a cluster of robots be defined as a maximal connected subgraph of the graph defined by the robots' positions, where two robots are considered to be adjacent if another robot cannot fit in between them (in other words, robots i and j are adjacent if $||\mathbf{p_j}^{(t)} - \mathbf{p_i}^{(t)}|| < 4r$). The sizes of the clusters can be found by using a depth-first search algorithm (Knuth, 1997). As the objective of this work is to bring all the robots into a single cluster, we define the cluster metric as the proportion of robots in the largest cluster:

$$c^{(t)} = \frac{\text{number of robots in the largest cluster at time } t}{n} \quad (2)$$

3.3. Robotic platform

The robotic platform that has been used in this study is the e-puck robot (Mondada et al., 2009), shown in Figure 1, which is a miniature, differential wheeled mobile robot. The e-puck's diameter and height are approximately 7.4 cm and 5.5 cm, respectively, and its weight is approximately 150 g.

The e-puck is equipped with a directional camera located at its front, which has been used in this study to realize the binary sensor in the physical implementation[3] (see Section 7). The e-puck's processor is a Microchip dsPIC micro-controller with 8 KB of RAM and 144 KB of flash memory.

3.4. Simulation platform

The simulations were performed using the open-source Enki library (Magnenat et al., 2011), which is used by Webots™ (Michel, 2008) in 2-D mode. Enki is capable of modeling the kinematics and the dynamics of rigid bodies in two dimensions, and has a built-in model of the e-puck. In Enki, the body of an e-puck is modeled as a disk of diameter 7.4 cm and mass 152 g. The inter-wheel distance is 5.1 cm. The velocities of the left and right wheels along the ground[4] can be set independently in $[-12.8, 12.8]$ cm/s. The binary sensor was realized by projecting a line from the robot's front and checking whether it intersects with another robot's body. The length of the control cycle was set to 0.1 s, and the physics was updated at a rate of 10 times per control cycle.

4. Controller synthesis

4.1. Controller structure

As the robots are memoryless, and are only equipped with one binary sensor, any controller for their behavior can be reduced to a mapping from each of the two possible sensor readings, $I = 0$ and $I = 1$, onto a pair of pre-defined velocities for the left and right wheels. This controller is an *if-then-else* statement, and performs no arithmetic computations. Let $\bar{v}_l, \bar{v}_r \in [-1, 1]$ represent the normalized left and right wheel velocities, respectively, where -1 (1) corresponds to the wheel rotating backwards (forwards) with maximum speed. The controller can be written as a quadruple,

$$\mathbf{x} = (\bar{v}_{\ell 0}, \bar{v}_{r0}, \bar{v}_{\ell 1}, \bar{v}_{r1}), \ \mathbf{x} \in [-1, 1]^4 \quad (3)$$

where $\bar{v}_{\ell 0}$ refers to the normalized velocity of the left wheel when $I = 0$, and so on.

4.2. Impossibility of guaranteeing aggregation with a limited sensing range

Theorem 4.1. *Let the range of the sensor be limited to some value δ, and let the robots initially form a connected visibility graph.[5] Under the problem formulation of Section 3.1, it is impossible to guarantee that the robots will always aggregate into one cluster (as defined in Section 3.2.2), irrespective of the controller used.*

Proof. Henceforth, we neglect the inertia of the robots, and assume that a robot's velocity changes instantly when its actuation parameters change. For a fixed pair of wheel velocities, in general, a differential wheeled robot moves along a circular trajectory. Two special cases arise: the radius of curvature can be zero, in which case the robot rotates on the spot, and it can be infinite, in which case the robot moves in a straight line. Let N denote that a robot does not move at all; S denote that it rotates on the spot (in either direction), and F and B that it moves forwards or backwards along a circular trajectory (in either direction; possibly with an infinite radius of curvature). The controller (see Section 4.1) maps each of the two possible sensor readings onto one of these four behaviors, meaning that the same sensor reading always results in the same relative displacement (unless the robot collides with other robots). Let us use a tuple $(*_0, *_1) \in \{N, S, F, B\}^2$, to denote what the robot does when $I = 0$ and $I = 1$. The proof now reduces to finding, for each of the $4 \times 4 = 16$ possible tuples, at least one initially connected configuration of robots with a limited sensing range δ, that does not lead to aggregation.

The seven cases where the tuple contains at least one N are trivial, as is the case (S, S). Pathological initial configurations for the cases $(B, *)$ and $(F, *)$ are easily achieved with $n = 2$, where the robots start off at a distance δ apart,[6] not seeing each other, and it is ensured that they will never see each other as they move. The case (S, B) is also easily eliminated with $n = 2$ robots that are initialized a distance δ apart: once one of these robots sees the other, it moves backwards and the connectivity is broken and never restored. The remaining case is (S, F). This case cannot be eliminated with $n = 2$; in particular, in the special case where the robots move forward in a straight line when $I = 1$, this controller can easily be shown to always aggregate $n = 2$ robots. We therefore consider $n = 3$ robots: i starts off at a distance δ from j, which in turn starts off at a distance δ from k. With the right set of initial orientations, j sees k first, and moves towards it, while i rotates on the spot. i is now disconnected from j and k, and rotates on the spot indefinitely, while j and k move towards each other. Diagrams of pathological initial configurations for the non-trivial cases are provided in Appendix B. □

In view of this result, we use an unlimited sensing range in the simulations performed in the rest of this paper, except in Section 6.2, where we investigate the effect of the sensing range. We believe that the good experimental results obtained (Section 7) show that in practice, the solution that we propose is not undermined by this result.

4.3. Controller optimization

We performed a grid search over the entire space of possible controllers, in order to (a) find the optimal controller (within a finite resolution), and (b) obtain a visualization of the performance landscape. We used a resolution of 21 settings per parameter, with each parameter taking values in $\{-1, -0.9, \ldots, -0.1, 0, 0.1, \ldots, 0.9, 1\}$. Therefore, $21^4 = 194,481$ controllers were evaluated.

In order to evaluate each controller, 100 simulations employing the controller were run for $T = 1800$ time steps (corresponding to 180 s) with different initial configurations of $n = 10$ robots.[7] In each simulation, the robots were initialized with a uniform random distribution in a (virtual) square of sides 316.23 cm, such that the area per robot was, on average, $10,000\ \text{cm}^2$. The performance of the controller, $\mathbf{x}$, in each simulation $k \in \{1, 2, \ldots, 100\}$, was evaluated by using a metric that is based on the robots' dispersion, as defined in Equation (1):

$$U(\mathbf{x}) = \sum_{t=0}^{T-1} t\, u^{(t)} \tag{4}$$

Equation (4) is designed to reward both a low dispersion at the end of the time interval, as well as the speed with which the robots aggregate. It penalizes large values of $u^{(t)}$ at every time instant (except for the very first one), but gives increasing importance to small values of $u^{(t)}$ for increasing t. The overall performance of the controller was calculated as the average of these values, i.e. $\bar{U}(\mathbf{x}) = \sum_{k=1}^{100} U_k(\mathbf{x})/100$.

The performance landscape of the controller is five-dimensional (four controller parameters plus the performance measure, $\bar{U}$), and cannot be visualized directly. Therefore, we considered each combination of two controller parameters ($\binom{4}{2} = 6$ combinations) as a sub-space, and for each point in this sub-space, we calculated the performance measure as the best (i.e. the minimum) value that is achievable with the remaining two parameters as degrees of freedom. For instance, on the sub-space $(\bar{v}_{\ell 0}, \bar{v}_{r0})$, the performance measure $\bar{U}^*(\bar{v}_{\ell 0}, \bar{v}_{r0})$ was calculated as:

$$\bar{U}^*(\bar{v}_{\ell 0}, \bar{v}_{r0}) = \min_{\bar{v}_{\ell 1}, \bar{v}_{r1}} \bar{U}(\bar{v}_{\ell 0}, \bar{v}_{r0}, \bar{v}_{\ell 1}, \bar{v}_{r1})$$

Figure 2 shows the landscape of the reciprocal of $\bar{U}^*$ over the six sub-spaces. The reciprocal was taken in order to improve the plot's visuals, because otherwise, extreme values of U, caused by the robots diverging due to a badly performing controller, dominate the landscape and render the other points indistinguishable from each other.

As the robots used here are symmetrical, if their left and right wheel velocities are inverted, their behavior is unaffected, other than that it is 'inverted' in space. For this reason, we expect (a) the landscapes on $(\bar{v}_{\ell 0}, \bar{v}_{r0})$ and $(\bar{v}_{\ell 1}, \bar{v}_{r1})$ to be symmetrical along the diagonals $\bar{v}_{\ell 0} = \bar{v}_{r0}$ and $\bar{v}_{\ell 1} = \bar{v}_{r1}$, respectively, (b) the landscape on $(\bar{v}_{\ell 0}, \bar{v}_{r1})$ to be identical to the one on $(\bar{v}_{r0}, \bar{v}_{\ell 1})$, and (c) the landscape on $(\bar{v}_{r0}, \bar{v}_{r1})$ to be identical to the one on $(\bar{v}_{\ell 0}, \bar{v}_{\ell 1})$. The landscapes shown in Figure 2 agree with these expectations. The small discrepancies that are present arise from the fact that the controllers are only evaluated a finite number of times.

We now analyze the features of the landscape. Note that the landscape is not uni-modal, but rather reveals rich multi-modal features, with well-performing controllers

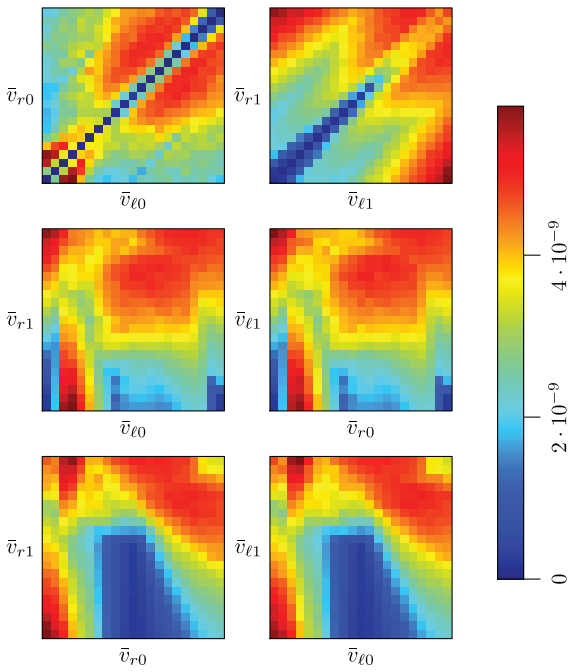

Fig. 2. Visualization of the performance landscape over the entire space of possible controllers. Each axis in each plot ranges between -1 and 1, corresponding to the maximum backwards and forwards velocity, respectively, of one of the robot's wheels. A higher value signifies a better performance. See the text for more details.

being present in different areas of the controller space. The landscape on the sub-space $(\bar{v}_{\ell0}, \bar{v}_{r0})$ (top-left plot in Figure 2) shows a marked "hill" along the diagonal defined by $\bar{v}_{\ell0} = \bar{v}_{r0}$, implying that having the robots move in a straight line when they do not perceive another robot leads to poor performance. The landscape on the sub-space $(\bar{v}_{\ell1}, \bar{v}_{r1})$ (top-right plot in Figure 2) also shows a "hill" along the diagonal defined by $\bar{v}_{\ell1} = \bar{v}_{r1}$, implying that it is sub-optimal for the robots to move straight towards perceived robots. This is an interesting, and somewhat counter-intuitive, result that illustrates the challenges involved in hand-designing controllers for collective behaviors.

In order to investigate how sensitive the performance landscape is to the particular initial configuration of robots used, we performed a separate grid search with a different initialization pattern. The robots were initialized in a straight line, spaced 50 cm apart, with random orientations. The resulting landscape (corresponding to the one presented in Figure 2), is available in the online supplementary material (see Gauci et al. (2013)). This landscape is not substantially different from the one in Figure 2; however, the resultant optimal controller lies in a different region. Nevertheless, simulations confirmed that the controller presented here (i.e. the optimal one in Figure 2) also consistently aggregates robots that are initialized in a straight line.

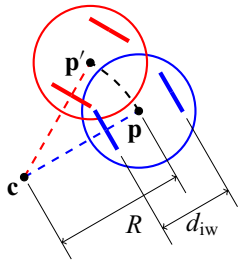

Fig. 3. Differential wheeled robot kinematics. Over a short time interval, the robot moves along a circular trajectory from position $\mathbf{p}$ (blue) to position $\mathbf{p}'$ (red). R and $\mathbf{c}$ denote the radius and the center of curvature, respectively, and d_{iw} denotes the robot's inter-wheel distance.

The optimal controller over the entire space of possible controllers in Figure 2 is given by $\mathbf{x} = (\bar{v}_{\ell0}, \bar{v}_{r0}, \bar{v}_{\ell1}, \bar{v}_{r1}) = (-0.7, -1, 1, -1)$. The following analyses and experiments in this paper have been performed with this controller.

5. Controller analysis

In this section, we show how the optimal controller that was found by the grid search in Section 4 exploits the geometry and the physics that are brought about by the robots' embodiment, in order to bypass the need for computation. In Section 5.1, we begin by analyzing how a robot behaves when it is executing the controller. In Section 5.2, we prove that a robot that is executing the controller always aggregates with another, static robot, or a circular, static cluster of already-aggregated robots. We also calculate an upper bound on the aggregation time. In Section 5.3, we generalize the analysis of Section 5.2 to the case of two robots that are simultaneously executing the controller. We prove that these always aggregate, and derive equations that can be used to numerically calculate an upper bound on the aggregation time.

5.1. Individual robot behavior

Recall that for a fixed pair of wheel velocities, a differential wheeled robot moves, in general, along a circular trajectory of radius R, with an angular speed ω. As shown in Figure 3, let $\mathbf{p}$ represent the position of a differential wheeled robot, $\mathbf{c}$ represent its center of curvature, d_{iw} represent its inter-wheel distance, and v_ℓ and v_r represent its left and right wheel velocities along the ground, respectively. Then, it can be shown (Dudek and Jenkin, 2010) that R and ω are related to v_ℓ and v_r by:

$$R = \frac{d_{\text{iw}}}{2}\left(\frac{v_r + v_\ell}{v_r - v_\ell}\right), \qquad \omega = \frac{1}{d_{\text{iw}}}(v_r - v_\ell) \tag{5}$$

In the optimal controller found by the grid search, for $I = 0$, $\bar{v}_{\ell0} = -0.7$, $\bar{v}_{r0} = -1$. By multiplying these normalized values by a factor of 12.8 cm/s (the maximum speed of the e-puck), and using Equation (5), we obtain

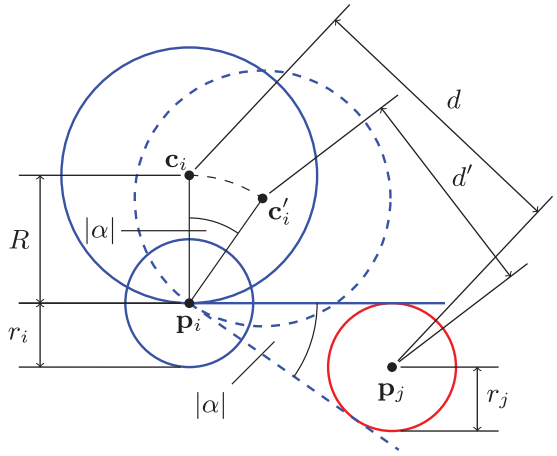

Fig. 4. Scenario with a moving robot i (small blue circle) and a static robot (or circular cluster) j (red circle). Robot i moves backwards along its circular trajectory centered at c_i (large solid blue circle), until it sees robot j (solid blue line). It then rotates clockwise on the spot through an angle of size $|\alpha|$, until it no longer sees robot j (dashed blue line). As it rotates, robot i's center of curvature moves from c_i to c'_i. Robot i then moves backwards along its new circular trajectory (dashed blue circle).

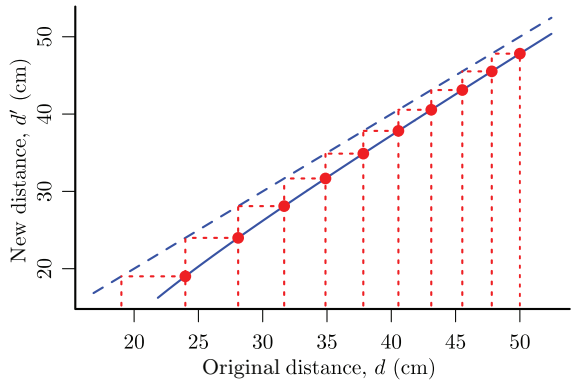

Fig. 5. Mapping from the original distance d to the new distance d' between the center of curvature of a moving robot, i, to a static robot, j. The red trajectory illustrates how, starting some value $d_0 > R + r_i + r_j$, d decreases stepwise until it becomes smaller than or equal to $R + r_i + r_j$. For the values used here, $R + r_i + r_j$ corresponds to 21.85 cm. The value of d_0 used in this figure is 50 cm.

$R_0 \approx 14.45$ cm, $\omega_0 \approx -0.75$rad/s. Similarly, for $I = 1$, we obtain $R_1 = 0$ cm, $\omega_1 \approx -5.02$ rad/s. We can now describe the behavior of the robot as follows. When $I = 0$, it moves backwards along a circular trajectory in a clockwise fashion. When $I = 1$, it rotates clockwise on the spot, with the maximum possible angular velocity.

In the remainder of this section, we assume that the robots react instantaneously when I changes from 0 to 1 or vice versa (i.e. the sensing is continuous). As $R_1 = 0$ and will not feature in the analyses, we henceforth drop the "0" subscript from R_0, with R representing this quantity unless otherwise stated.

5.2. *One moving robot and one static robot or circular cluster*

Let us analyze the situation where the environment contains one robot that is executing the controller, and one circular, static object. The latter can represent either a static robot, or a cluster of already-aggregated robots that is quasi-circular and quasi-static. Figure 4 shows the moving robot, i, of radius r_i, (small blue circle) and the static robot (or circular cluster), j, of radius r_j (red circle). Let us define $d = ||\mathbf{p}_j - \mathbf{c}_i||$. Note that the two robots cannot collide with each other as long as $d > R + r_i + r_j$, and we therefore consider them as being aggregated when this condition no longer holds.

Theorem 5.1. *One moving robot will always aggregate with another static robot or circular cluster.*

Proof. Consider the scenario shown in Figure 4. We want to obtain an expression for $d' = ||\mathbf{p}_j - \mathbf{c}'_i||$ in terms of $d = ||\mathbf{p}_j - \mathbf{c}_i||$. Without loss of generality, let $\mathbf{c}_i = [0, 0]^{\mathrm{T}}$, and let the sensor of robot i point to the right along the horizontal at the moment when it starts seeing robot j. It follows that:

$$\mathbf{p}_j = \begin{bmatrix} \delta \\ -(R + r_j) \end{bmatrix} \tag{6}$$

where δ is the length of the "sensory beam" (from the center of robot i to the point at which it touches the circumference of robot j), at the instant when robot i starts seeing robot j. δ is given by:

$$\delta = \sqrt{d^2 - (R + r_j)^2} \tag{7}$$

From the geometry of Figure 4, the coordinates of $\mathbf{c}'_i$ are given by:

$$\mathbf{c}'_i = \begin{bmatrix} R \sin|\alpha| \\ R(\cos|\alpha| - 1) \end{bmatrix} \tag{8}$$

where $|\alpha| \in (0, \pi)$. Moreover, we can write the equation $\tan(|\alpha|/2) = r_j/\delta$, from which we obtain:[8]

$$\sin|\alpha| = \frac{2\delta r_j}{\delta^2 + r_j^2}, \quad \cos|\alpha| = \frac{\delta^2 - r_j^2}{\delta^2 + r_j^2} \tag{9}$$

Finally, by substituting Equations (6)–(9) into $d' = ||\mathbf{p}_j - \mathbf{c}'_i||$, and simplifying the resulting expression,[9] we obtain:

$$d' = \sqrt{d^2 - 4Rr_j}, \qquad d > R + r_i + r_j \tag{10}$$

Figure 5 shows the mapping from d onto d' according to Equation (10), for the values of R and r_j that are used here. The red trajectory illustrates how, starting from some value $d_0 > R + r_i + r_j$, d decreases stepwise until it becomes smaller than or equal to $R + r_i + r_j$, at which point the robots are considered to be aggregated. □

Theorem 5.2. *The time, in s, that it takes for a moving robot to aggregate with a static robot or circular cluster of radius r_j is bounded by $2\pi m/|\omega_0|$ s, where*

$$m = \left\lceil \frac{d_0^2 - \left(R + r_i + r_j\right)^2}{4Rr_j} \right\rceil \tag{11}$$

and d_0 is the initial value of $||\mathbf{p}_j - \mathbf{c}_i||$ (see Figure 4).

Proof. We begin by calculating the number of $d \to d'$ updates that must occur until $d \leq R + r_i + r_j$. Let us write Equation (10) as a recurrence relation:

$$d_{m+1} = \sqrt{d_m^2 - 4Rr_j}, \qquad d_m > R + r_i + r_j \tag{12}$$

Now, let $\hat{d}_m = d_m^2$, such that we can re-write Equation (12) as $\hat{d}_{m+1} = \hat{d}_m - 4Rr_j$, the solution to which is given by $\hat{d}_m = \hat{d}_0 - 4Rr_j m$. Therefore, the solution to Equation (12) is given by:

$$d_m = \sqrt{d_0^2 - 4Rr_j m}, \qquad d_m > R + r_i + r_j \tag{13}$$

The number of $d \to d'$ that are required until $d \leq R + r_i + r_j$ is now given by setting $d_m = R + r_i + r_j$ in Equation (13), solving for m, and taking the ceiling of this value, which leads to:

$$m = \left\lceil \frac{d_0^2 - \left(R + r_i + r_j\right)^2}{4Rr_j} \right\rceil \tag{14}$$

Given m, we can now calculate an upper bound on the time that it takes until $d \leq R + r_i + r_j$. From the time that robot i stops seeing robot j, it takes robot i slightly less than one cycle along its circular trajectory until it once again starts seeing robot j. The time that it takes for robot i to rotate on the spot while it sees robot j is negligible, because $|\omega_1|$ is much larger than $|\omega_0|$. At most one $d \to d'$ update occurs per cycle of robot i along its circular trajectory. The aggregation time is thus bounded by $2\pi m/|\omega_0|$ s. □

5.3. Two simultaneously moving robots

We now analyze the situation with two robots that are simultaneously executing the controller. In this scenario, it is sensible to define $d = ||\mathbf{c}_j - \mathbf{c}_i||$. Note that the two robots cannot collide with each other as long as $d > 2(R + r)$, and we therefore consider them as being aggregated when this condition no longer holds.

Theorem 5.3. *Two simultaneously moving robots will always aggregate, and an upper bound on the time that this takes is given by solving Equations* (15)–(18).

Proof. Consider the arrangement shown in Figure 6. Without loss of generality, let $\mathbf{c}_i = [0, 0]^{\mathrm{T}}$, and $\mathbf{c}_j = [d, 0]^{\mathrm{T}}$. As in the case of the previous section, we want to calculate $d' = ||\mathbf{c}_j - \mathbf{c}_i'||$; d' now depends not only on d, but also on the position of robot j along its circular trajectory at the instant when robot i starts seeing it. The latter can be parametrized by the angle that robot j makes with the horizontal,[10] $\theta_i \in \left(-\frac{\pi}{2}, 0\right)$. From the geometry of Figure 6:

$$\mathbf{c}_i' = \mathbf{p}_i + R \begin{bmatrix} \cos\left(\pi + \theta_i - |\alpha|\right) \\ \sin\left(\pi + \theta_i - |\alpha|\right) \end{bmatrix} \tag{15}$$

where $\mathbf{p}_i = \mathbf{c}_i + R\left[\cos\theta_i, \sin\theta_i\right]^{\mathrm{T}}$. We now need expressions for θ_i and $|\alpha|$ in terms of d and θ_j.

The expression for θ_i can be obtained from the geometry of Figure 6 as:

$$\theta_i = \arcsin\left(\frac{R\sin\theta_j}{||\mathbf{p}_j - \mathbf{c}_i||}\right) + \arcsin\left(\frac{R + r}{||\mathbf{p}_j - \mathbf{c}_i||}\right) - \frac{\pi}{2} \tag{16}$$

where $\mathbf{p}_j = \mathbf{c}_j + R\left[\cos\theta_j, \sin\theta_j\right]^{\mathrm{T}}$.

We now turn to calculating $|\alpha|$. In Figure 6, robot j moves along its circular trajectory while robot i rotates on the spot.[11] Let t_{rot} denote the time that robot i spends rotating on the spot while seeing robot j, from which it follows that $|\alpha| = |\omega_1| t_{\mathrm{rot}}$. Now, we can express $\mathbf{p}_j'$ in terms of t_{rot}, as $\mathbf{p}_j' = \mathbf{c}_j + R\left[\cos\left(\theta_j - |\omega_0| t_{\mathrm{rot}}\right), \sin\left(\theta_j - |\omega_0| t_{\mathrm{rot}}\right)\right]^{\mathrm{T}}$. Using this, in turn, we can express $|\alpha|$ in terms of d, θ_j, and t_{rot} by noting from the geometry of Figure 6 that:

$$\begin{aligned} |\alpha|\left(d, \theta_j, t_{\mathrm{rot}}\right) = {} & \angle\left(\mathbf{p}_j - \mathbf{p}_i\right) - \angle\left(\mathbf{p}_j' - \mathbf{p}_i\right) \\ & + \arcsin\left(\frac{r}{||\mathbf{p}_j - \mathbf{p}_i||}\right) \\ & + \arcsin\left(\frac{r}{||\mathbf{p}_j' - \mathbf{p}_i||}\right) \end{aligned} \tag{17}$$

where $\angle(\cdot)$ denotes the angle that the vector makes with the horizontal axis. The value of t_{rot}, and hence that of $|\alpha|$, is given when:

$$|\alpha|\left(d, \theta_j, t_{\mathrm{rot}}\right) = |\omega_1| t_{\mathrm{rot}} \tag{18}$$

The left hand side of Equation (18) is a complex expression in terms of d, θ_j, and t_{rot}, and as a consequence, it does not seem possible to obtain an analytical solution for $|\alpha|$ in terms of d and θ_j. Nevertheless, it is possible to obtain an accurate solution by solving Equation (18) numerically.[12]

We are now in possession of expressions for θ_i in terms of d and θ_j (Equation (16)) and an equation from which we can calculate $|\alpha|$ numerically given these quantities (Equation (18)). By substituting the values of θ_i and $|\alpha|$ into Equation (15), we obtain $\mathbf{c}_i'$, and from this, in turn, we obtain $d' = ||\mathbf{c}_i' - \mathbf{c}_j||$. Figure 7 shows a plot of $d' - d$ against θ_j for one $d \to d'$ update. The lowermost curve corresponds to $d = 2(R + r)$, and subsequent curves correspond to d increasing in steps of 10 cm. As one expects, as d increases, the curves become flatter (meaning that the effect of θ_j on d' diminishes), and $d' - d$ tends towards zero.

We can now calculate, starting from some initial $d = d_0 > 2(R + r)$, an upper bound on the number m of $d \to d'$ updates that need to occur until $d \leq 2(R + r)$. We do this by assuming the worst case scenario—that in each update,

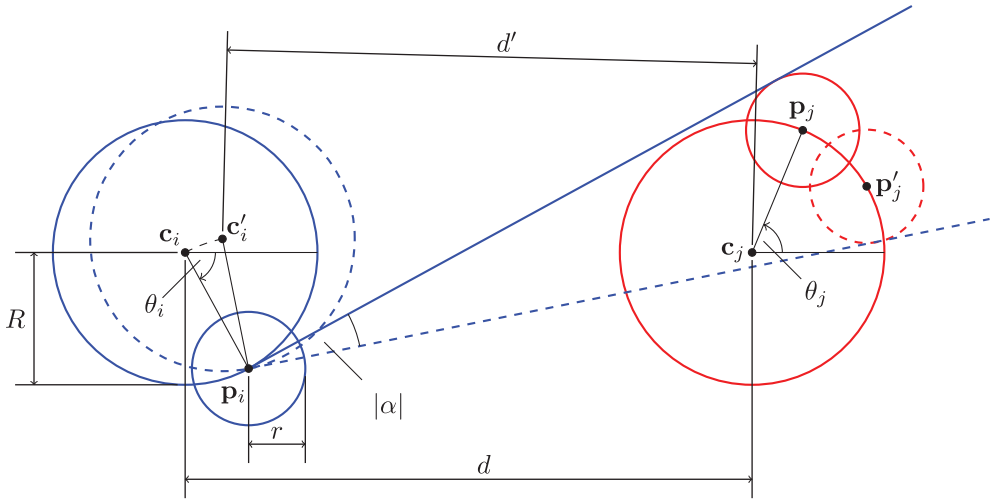

Fig. 6. Scenario with two simultaneously moving robots, i and j (small blue and red circles, respectively). Robot i moves backwards along its circular trajectory centered at c_i (large solid blue circle), until it sees robot j (solid blue line). Robot i then rotates clockwise on the spot through an angle of size $|\alpha|$, while robot j moves backwards along its circular trajectory. As it rotates, robot i's center of curvature moves from c_i to c'_i. Robot i stops rotating when it stops seeing robot j (dashed blue line), and it then moves backwards along its new circular trajectory (dashed blue circle).

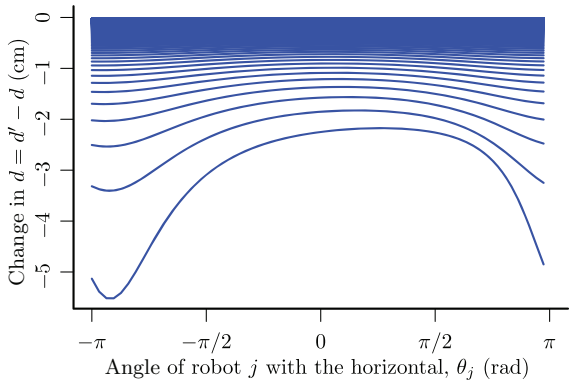

Fig. 7. $d' - d$ against θ_j for one $d \to d'$ update in the case of two simultaneously moving robots. The lowermost curve corresponds to $d = 2\,(R + r)$, and subsequent curves correspond to d increasing in steps of 10 cm.

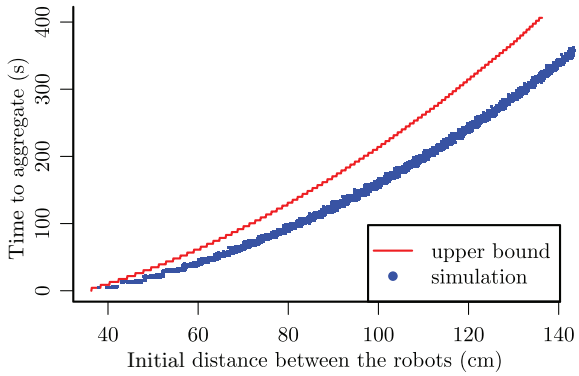

Fig. 8. The time required for two simultaneously moving robots to aggregate.

$d' - d$ assumes the closest value to zero that is possible (these values correspond to the maxima of the curves in Figure 7). As we do not have a closed-form expression for $|\alpha|$, we find m by numerically iterating from d onto d' using Equations (15)–(18), and counting the number of iterations that are required until $d' \leq 2\,(R + r)$. Once we have m, we obtain the time that elapses until $d \leq 2\,(R + r)$, as we did for the case of one moving robot in Section 5.2. In this case, at most two $d \to d'$ updates can occur per $2\pi/|\omega_0|$ s, because the two robots are moving simultaneously. Therefore, the aggregation time is bounded by $2\pi m/2|\omega_0| = \pi m/|\omega_0|$ s.

Figure 8 shows a plot of the upper bound aggregation time, starting from an initial value $d = d_0$ (red curve). Overlain on the plot are 1000 data points obtained from simulations with two simultaneously moving robots (blue crosses). By re-plotting Figure 8 using log-scales for both axes, it turns out that the relationship between log (time) and log (d_0) is sublinear for small values of d_0, and becomes asymptotically linear as d_0 increases. A least squares regression fit reveals that the slope of the linear part is almost exactly 2, which means that the relationship between d_0 and the time required until $d \leq 2\,(R + r)$ is asymptotically quadratic (compare this to the quadratic relationship in the case of one moving robot in Section 5.2, given by Equation (14)). □

5.4. *n robots*

We are not able to provide a rigorous mathematical analysis for the general case of n simultaneously moving robots.

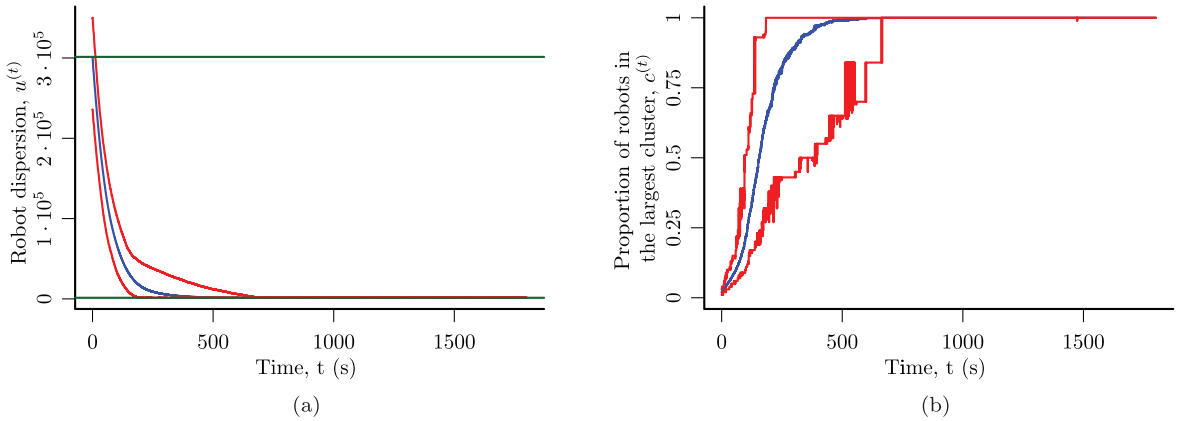

Fig. 9. These plots show the aggregation dynamics with $n = 100$ robots. In both plots, the horizontal axis represents the time, t. In (a), the vertical axis represents the dispersion of the robots (Equation (1), lower is better), and, in (b), it represents the proportion of the robots in the largest cluster (Equation (2), higher is better). The blue curves show the mean measure across 100 simulations with different initial configurations, whereas the red curves show the range (minimum, maximum) of the measure across the 100 simulations. In (a), the lower and the upper green lines show, respectively, the minimum possible dispersion for 100 robots, and their expected dispersion at the point of initialization.

Their aggregation is the result of complex collective movements of clusters of physically interacting robots, as can be seen in the accompanying video (Extension 1). We provide a qualitative analysis of this effect in Section 7.3.

6. Simulation studies

In the controller synthesis stage (Section 4), only $n = 10$ robots were used, in order to keep the computation time within reasonable limits. In the following, we analyze, with $n = 100$ robots, the aggregation dynamics (Section 6.1), and the effects of the sensing range (Section 6.2) and sensory noise (Section 6.3) on the aggregation performance. In Section 6.4 we analyze the scalability of the controller with up to $n = 1000$ robots. In all the simulations (regardless of n), the robots were initialized with a uniform random distribution in a (virtual) square, such that on average, the area per robot was 10000 cm^2—identical to that used in the controller synthesis stage.

6.1. Aggregation dynamics

We performed 100 simulations with $n = 100$ robots. In every run, the robots formed a single cluster. Figures 9(a) and (b) show, respectively, how the dispersion and the cluster metrics of the robots developed over time. The longest time taken to form a single cluster was 663 s, and on average, the dynamics reached steady state after around 750 s.

For $n = 100$ robots, the minimum possible dispersion $u^{(t)}$ (among hexagonal packings) is 25.9913 (Graham and Sloane, 1990). In the simulations, the final value of $u^{(t)}$ (i.e. after 1800 s) was only 8.92% larger than this theoretical lower bound (averaged across the 100 simulations).

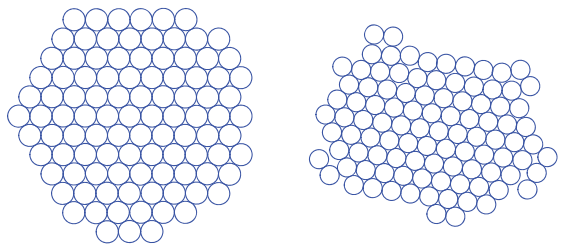

Fig. 10. (a) The configuration of 100 robots with the minimum possible dispersion (Equation (1)). (b) The final configuration of 100 robots after 1800 s with the controller found by the grid search in Section 4. This configuration comes from the run that had the median dispersion after 1800 s across 100 simulations with different initial configurations.

Figure 10 shows (a) the configuration with the minimum possible dispersion for 100 robots, and (b) the final configuration of 100 robots after 1800 s—the trial chosen for this figure was the one that led to the median robot dispersion in the final configuration, as found from the 100 simulations.

6.2. The effect of the sensing range

In this section, we study how the aggregation performance is affected when the sensing range of the robots (assumed to be unlimited in the other sections) is finite. As the robots are initialized within a virtual square of sides 1000 cm, a sensing range of $\delta_\infty = \left(1000^2 + 1000^2\right)^{\frac{1}{2}} = 1414$ cm can be considered as being virtually unlimited. We performed 100 simulations for each $\delta/\delta_\infty = \{0.0, 0.1, \ldots, 1.0\}$. In each

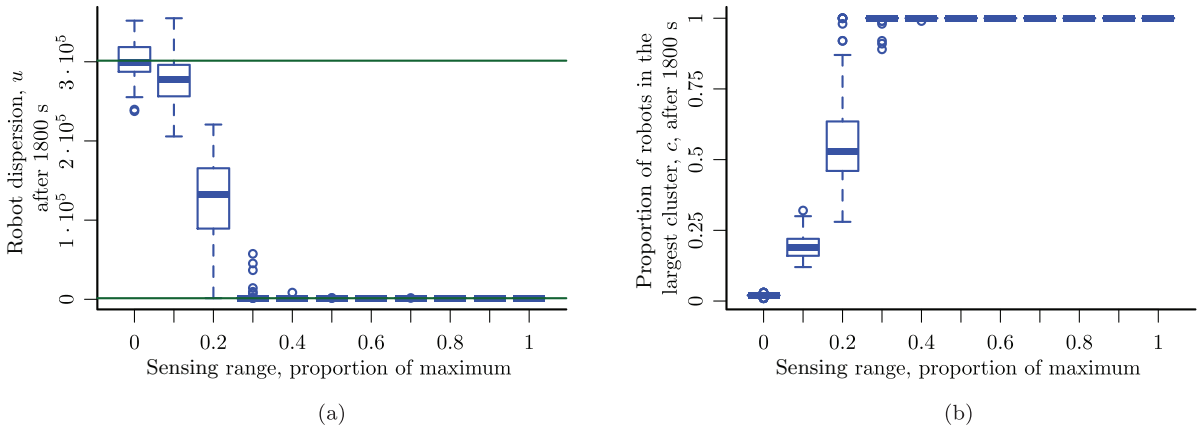

Fig. 11. These box plots show the effect of the sensing range on the aggregation performance with $n = 100$ robots. In both plots, the horizontal axis represents the sensing range, as a proportion of the maximum possible distance between two robots at the point of initialization. In (a), the vertical axis represents the dispersion of the robots after 1800 s (Equation (1), lower is better), and, in (b), it represents the proportion of the robots in the largest cluster after 1800 s (Equation (2), higher is better). In both plots, each box represents values obtained from 100 simulations with different initial configurations of robots. In (a), the lower and the upper green lines show, respectively, the minimum possible dispersion for 100 robots, and their expected dispersion at the point of initialization.

simulation, the dispersion of the robots and the proportion of robots in the largest cluster after 1800 s were recorded.

Figure 11 shows a box plot[13] with the results. The performance is virtually unaffected as δ/δ_∞ is reduced from 1.0 to 0.3. As δ/δ_∞ is reduced to 0.1, the median performance drops instantly, almost to the expected value at the point of initialization—equivalent to robots that do not move throughout the simulation (upper green line in Figure 11(a)).

6.3. The effect of sensory noise

In this section, we investigate how the aggregation performance is affected when the readings of the robots' binary line-of-sight sensors are corrupted by noise. We considered two types of noise on the sensor. False negative noise flips an input of $I = 1$ to $I = 0$ (i.e. a perceived robot is missed) with probability p_n. False positive noise flips an input of $I = 0$ to $I = 1$ (i.e. an imaginary robot is perceived) with probability p_p. We considered a grid of $11 \times 11 = 121$ noise level combinations, with p_n and p_p taking the values $\{0.0, 0.1, \ldots, 1.0\}$. For each combination, we performed 100 simulations, and the mean dispersion of the robots and the proportion of the robots in the largest cluster after 1800 s were recorded.

Figure 12 shows the noise performance landscapes as color maps. Note that for noise level combinations that lie on the diagonal defined by $p_n + p_p = 1.0$, the reading of the sensor is 0 with probability p_n, and 1 with probability p_p, irrespective of whether or not there is a robot in the direct line of sight. On this diagonal, the sensor provides no meaningful information. In the region $p_n + p_p > 1.0$, the sensor reading could be inverted so as to obtain a performance corresponding to the noise combination that mirrors the actual one in the diagonal $p_n + p_p = 1.0$.

The controller leads to a low dispersion and a high proportion of robots in the largest cluster for a sizeable subregion of noise level combinations—approximately, for up to $p_n + p_p < 0.6$. The controller is thus highly robust to both types of noise on the sensor.

6.4. Scalability

In order to investigate the scalability of the aggregation performance with binary line-of-sight sensors, the controller was evaluated with increasing numbers of robots. 100 simulations were performed for each value of $n \in \{100, 200, \ldots, 1000\}$.

Each simulation was stopped when the robots formed a single cluster, and the time taken for them to do so was recorded. Every trial finished, meaning that the controller is capable of achieving consistent, error free aggregation with at least 1000 robots. Figure 13 shows a box plot of the times taken to achieve a single cluster. A least squares regression fit of the form An^k (dashed red line in Figure 13) to the means of the times (red squares in Figure 13) yields $A = 5.34$, $k = 0.88$, which suggests that the time taken to achieve a single cluster increases slightly sub-linearly with the number of robots.

7. Experiments

The sensor/controller solution described in the previous sections was implemented on a system of 40 physical e-puck robots, in order to validate its feasibility.

7.1. Sensor implementation

The sensor was implemented using the e-puck's directional camera. The robots were fitted with black "skirts" in order

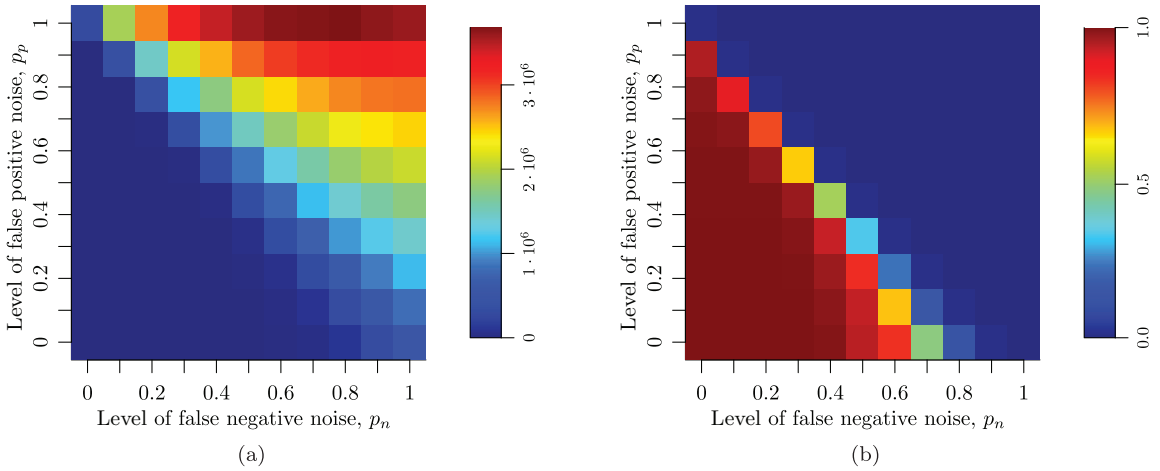

Fig. 12. These plots show the effect of sensory noise on the aggregation performance with $n = 100$ robots. In both plots, the horizontal and the vertical axes show, respectively, the level of false negative and false positive noise (see the text for details). In (a), the colors represent the dispersion of the robots after 1800 s (Equation (1), lower is better). In (b), they represent the proportion of the robots in the largest cluster after 1800 s (Equation (2), higher is better). In both plots, the color in each box represents the mean value across 100 simulations with different initial configurations.

to make them distinguishable against the white walls of the arena (see Figure 1). The e-puck's camera is a CMOS RGB color camera with a resolution of 640 (horizontal) by 480 (vertical) pixels, with corresponding viewing angles of 56° and 42°, respectively. Note that the amount of RAM available on the e-puck's micro-controller is not large enough to even store a single raw color image from the camera, which comes to illustrate the importance of keeping the amount of information processing to a minimum on small-scale robotic systems.

In principle, using one pixel of the camera is sufficient for implementing the binary line-of-sight sensor. However, in order to account for vertical misalignments between the orientations of the robots' cameras, we used the middle column of pixels from the camera to implement the sensor. This column of pixels is sub-sampled in order to obtain 15 equally spaced pixels from its bottom to its top, resulting in an area of width 1 pixel and height 15 pixels. The gray values of these 15 pixels are compared against a threshold, which was empirically set to 2/3 of the maximum possible value (pure white). If one or more of the pixels have a gray value below this threshold, the sensor gives a reading of $I = 1$. Otherwise it gives a reading of $I = 0$. The implemented sensor has been found to provide reliable readings for a range of up to around 150 cm.

7.2. Experimental setup and procedure

The arena used for the experiments is a rectangle of size 400 cm × 225 cm. It has a light gray floor, and is surrounded by white walls that are 50 cm in height. Its floor is marked with a grid of $15 \times 8 = 120$ points, spaced 25 cm from each other and from the walls. For each trial, 40 of these points were chosen randomly to serve as the initial positions of the robots. With the robots positioned on these

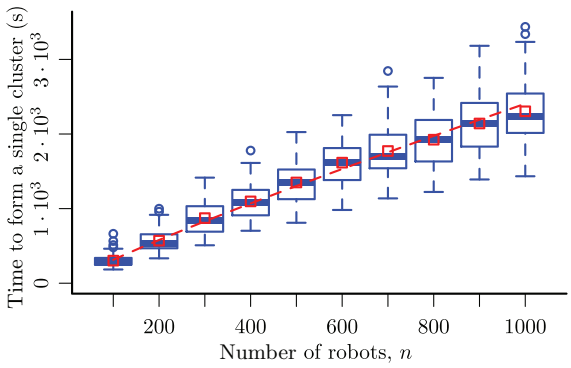

Fig. 13. This box plot shows the time taken by the controller to aggregate increasing numbers of robots into a single cluster. Each box represents values obtained from 100 simulations with different initial configurations of robots. The red points show the mean values, and the dashed red line shows a least squares regression fit to these points of the form An^k, where $A = 5.34$, $k = 0.88$.

points, an infrared signal was issued to instruct each robot to turn on the spot through a randomly generated portion of a revolution, such that robots now faced in random directions. Another infrared signal was then issued to instruct the robots to start executing the controller. The robots were programmed to stop automatically after 900 s. If a robot stopped moving during a trial, the start signal was reissued in an attempt to restart it (moving robots were unaffected by this signal). If the robot failed to restart, it was left in the arena until the end of the trial.

We performed 30 trials with $n = 40$ robots. Each trial was recorded by an overhead camera. All the 30 videos are available in the online supplementary material (Gauci et al. (2013)).

7.3. Results

Figure 14 shows a series of snapshots taken at different times from a single trial. After around 180 s, the robots had formed two clusters, and these clusters eventually approached each other, made contact after around 360 s, and amalgamated into one "body" after around 540 s. The dynamics of this trial are representative of those of most of the other trials. Figure 15 shows a plot of the performance metrics over time (compare this to Figure 9). For $n = 40$ robots, the lower bound of $u^{(t)}$, as reported in (Graham and Sloane, 1990), is 220.33. The final value of $u^{(t)}$ (i.e. after 900 s) was 49.08% larger than this theoretical lower bound (averaged across the 30 trials).

Figure 16 shows the final configurations of the robots in all the 30 trials. In 27 of the trials, all the 40 robots were aggregated into one cluster within 900 s. In two of the trials, the largest cluster contained 39 robots. In one trial, the robots were divided into two clusters of 25 and 15 robots after 900 s.[14] Therefore, on average, across the 30 trials, the proportion of robots in the largest cluster after 900 s was equal to 98.6%.

We observed that, after starting to execute the controller, most of the robots quickly form small clusters. These clusters then move collectively, and merge into one another. The collective movement of a cluster occurs by an effect that is similar to convection. Robots that are on the periphery of the cluster, and which do not see other clusters as they rotate on the spot, often displace themselves, and end up on another part of the cluster's periphery, from which other clusters can be seen. As this happens repeatedly, the cluster moves as a whole towards other clusters. This can be seen in the accompanying video (Extension 1) and in the videos in the online supplementary material (Gauci et al. (2013)). Note that this behavior seems to be rather inefficient in terms of the speed of the aggregation process. We therefore conjecture that the severe limitations imposed on the amount of sensing and information processing lead to a performance trade-off as compared to deterministic algorithms that make use of more sensory information and/or computation (Section 2.3).

A current limitation of this work is that the robots are unable to stop searching for other clusters, even when none are present. This can lead to a continual energy consumption. However, the robots could be instructed to sleep when they have been rotating on the spot for some time, and wake up periodically in order to check whether this is still the case. In this way, robots that are inside a cluster would consume very little energy.

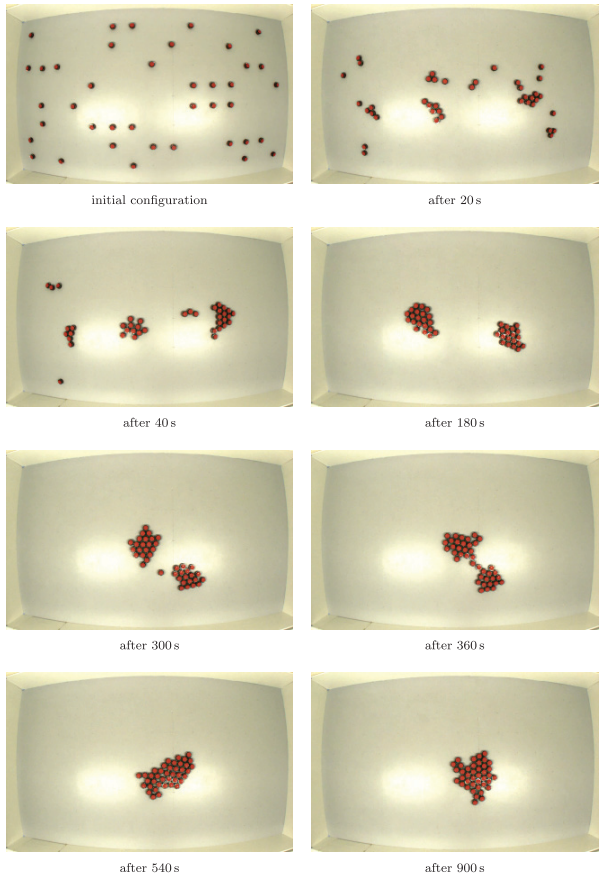

Fig. 14. A sequence of snapshots taken during one of the experiments with 40 physical e-puck robots. The dimensions of the arena are 400 cm × 225 cm. The bright patches that can be seen in the arena are reflections from the overhead lighting, and did not affect the behavior of the robots.

8. Conclusion

This paper has presented the simplest solution so far to the problem of self-organized robot aggregation in a homogeneous environment. In this solution, each robot has only one sensor that provides it with one bit of information about its environment: whether or not there is another robot in its line of sight. As the sensor is binary and the robots are memoryless, computation by the controller is futile, as any controller can be reduced to a mapping of each of the two possible sensor readings onto a set of pre-defined actuation parameters.

The optimal controller was found by a grid search over the entire space of possible controllers, using 10 robots that are initialized uniformly randomly within a square region. Proofs of aggregation, as well as time upper bounds, have been provided for the cases of one moving robot and one static robot or circular cluster, and two simultaneously moving robots. Simulations have shown that the robots can form very dense packings (on average, only 8.92% worse than the optimum) when employing the sensor/controller solution. Such dense packings are desirable, for example, if after aggregating, the robots are required to perform tasks as a physically connected robotic ensemble (Goldstein et al., 2005; Groß and Dorigo, 2008). The simulations have also shown that the sensing range of the robots can be reduced to around 30% of the span of their initial configuration, without a significant detriment to the aggregation performance; that the solution is highly robust with respect to noise on

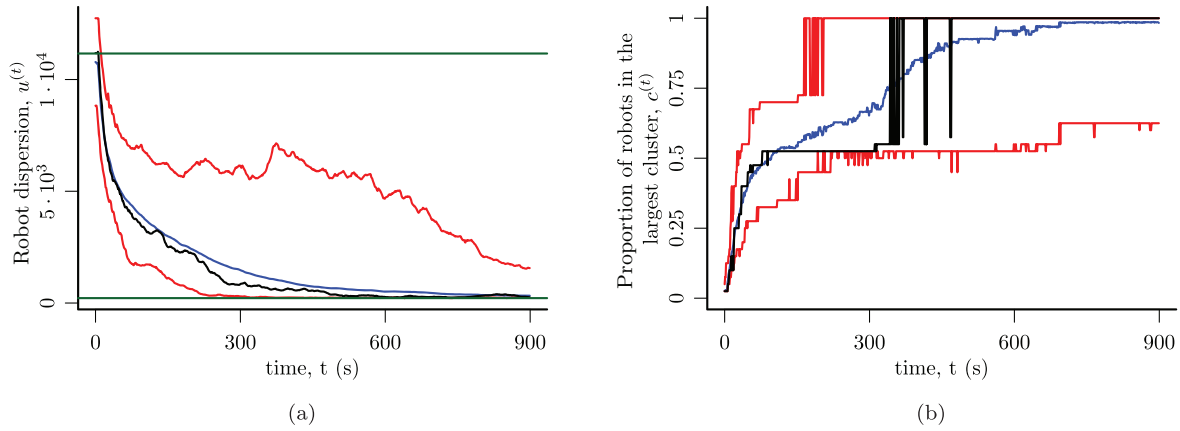

Fig. 15. These plots show the aggregation dynamics with $n = 40$ physical e-puck robots. In both plots, the horizontal axis represents the time, t. In (a), the vertical axis represents the dispersion of the robots (Equation (1), lower is better), and, in (b), it represents the proportion of the robots in the largest cluster (Equation (2), higher is better). The blue curves show the mean measure across the 30 trials, whereas the red curves show the range (minimum, maximum) of the measure across the 30 trials. The black curves show the measures corresponding to the trial shown in Figure 14. In (a), the lower and the upper green lines show, respectively, the minimum possible dispersion for 40 robots, and their expected dispersion at the point of initialization.

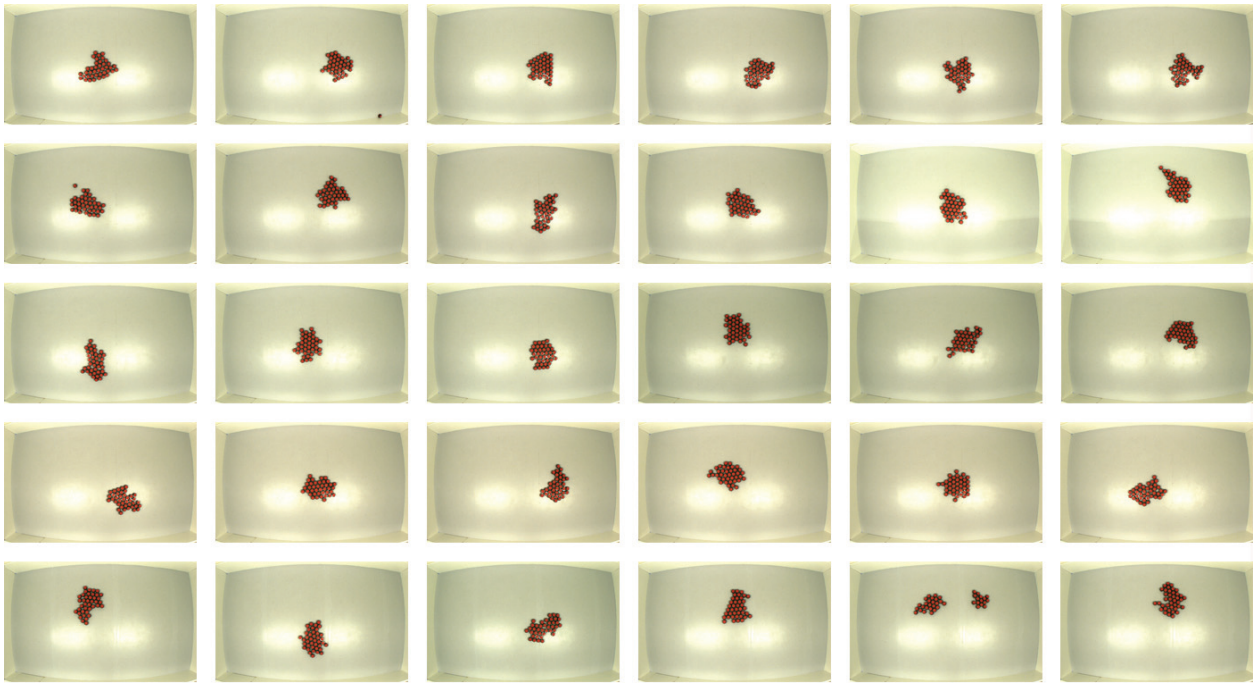

Fig. 16. Overhead snapshots of the final configurations of the 40 robots in the 30 experiments. The dimensions of the arena are 400 cm×225 cm. The bright patches that can be seen in the arena are reflections from the overhead lighting, and did not affect the behavior of the robots.

the sensor; and that the solution can aggregate at least 1000 robots consistently. The solution was ported onto a system of 40 physical e-puck robots. On average, across 30 experiments, 98.6% of the robots aggregated into one cluster within 900 s. From these experiments, we observed that the aggregation depends on the complex collective movement of clusters of physically interactive robots, which happens by an effect that is similar to convection.

We believe that the solution provided here is not restricted to the e-puck robotic platform on which it has been validated so far. Rather, it is employable on any system of robots that have the abilities to (a) tell whether there is another robot in their line of sight, and (b) move along circular trajectories of different curvatures. We anticipate that the ease of porting the synthesized controller onto a system of physical robots that was demonstrated here will

also carry forward to other robotic platforms, because of the extreme simplicity of the sensor and the controller. However, we suspect that the shape and the material properties of the robots will have an impact on the aggregation performance and the structures formed (see, e.g., Damasceno et al. (2012)), which is yet to be investigated.

In this study, a binary line-of-sight sensor was implemented on a relatively small scale robotic platform using a CMOS camera. It remains to be investigated how such a sensor could be implemented on smaller scales. Potentially, if this helps the implementation, the system could be modified such that the sensor, while still being binary in nature, covers a relatively wider angle than the very narrow field of view associated with our single-pixel-wide realization. This possibility will be explored in future work.

It is high time to gain more understanding of how meaningful collective behaviors in distributed robotic systems can be obtained with exceptionally low hardware requirements on the part of the individual robots. This will pave the way for the implementation of such systems with millions of robotic units (Fitch and Butler, 2008; Ashley-Rollman et al., 2011), and at scales ranging from the micro down to the nano (Sitti, 2007).

The framework introduced in this work has been shown to be applicable to the relatively more complex task of object clustering, both in simulation and on a physical robotic platform (Gauci et al., 2014). In the future, we intend to investigate the applicability of the framework to further tasks and in more complex environments, such as ones with obstacles. We also intend to implement the solutions on a physical robotic system at the micro scale.

Funding

This work was supported by a Strategic Educational Pathways Scholarship (Malta), the European Union—European Social Fund (ESF) under Operational Programme II - Cohesion Policy 2007–2013, "Empowering People for More Jobs and a Better Quality of Life", and a Marie Curie European Reintegration Grant within the 7th European Community Framework Programme (grant number PERG07-GA-2010-267354).

Notes

1. A preliminary version of this work was presented at the 2012 International Symposium on Distributed Autonomous Robotic Systems, Baltimore, MD, USA (Gauci et al., 2012). This paper includes a substantial amount of new analyses (e.g. all the plots, and Section 5), as well as results from a new set of 30 experiments with 40 physical e-puck robots (Section 7).
2. It will later be shown that the optimal controller can also be implemented on other robotic platforms.
3. As we will explain in Section 7.1, we only use an area of the image that is a single pixel wide. Although this inevitably makes the sensor have a narrow field of view (rather than being strictly a line of sight), the results obtained show that this does not noticeably undermine the system's performance.
4. This refers to the instantaneous linear velocity of the robot at the point at which the wheel makes contact with the ground. It is equal to the wheel's angular velocity multiplied by the wheel's radius.
5. In this graph, two robots are defined as being connected if the distance between them does not exceed their sensing ranges.
6. More precisely, the distance δ refers to the length of the sensors. If this is measured between the robots' peripheries, then the distance between their centers is actually $\delta + 2r$, where r is their radius.
7. This combination of number of robots and run time was found to achieve a good balance between keeping the computation time of the grid search within reasonable limits, and sufficiently capturing the dynamics of aggregation such that the resulting controller is scalable to larger numbers of robots.
8. These relationships can be derived by using standard trigonometric identities.
9. This simplification is not straightforward to work out by hand. We obtained it using the mathematical software Sage (http://www.sagemath.org/).
10. We use the convention of counter clockwise angles being positive, and clockwise angles being negative.
11. It should be noted that for a small range of θ_j, robot j will simultaneously be seeing robot i, and will therefore be rotating clockwise on the spot. This range, however, is very small, and its effect can therefore be neglected by treating robot j as if it were still moving along its circular trajectory. This assumption does not violate the worst case scenario.
12. We use the function `uniroot` that is available in the R software environment (http://www.r-project.org/).
13. The box plots presented here are all as follows. The line inside the box represents the median of the data. The edges of the box represent the lower and the upper quartiles (25th and 75th percentiles) of the data, and the whiskers represent the lowest and the highest data points that are within 1.5 times the inter-quartile range from the lower and the upper quartiles, respectively. Circles represent outliers.
14. After the trial was completed, the robots were restarted, and the two clusters eventually merged into one after around an additional 180 s.

References

Ando H, Oasa Y, Suzuki I, et al. (1999) Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Transactions on Robotics and Automation* 15: 818–828.

Ashley-Rollman MP, Pillai P and Goodstein ML (2011) Simulating multi-million-robot ensembles. In: *Proceedings of 2011 IEEE International conference on robotics and automation*, Shanghai, China, 9–13 May 2011, pp. 1006–1013.

Bahceci E and Şahin E (2005) Evolving aggregation behaviors for swarm robotic systems: A systematic case study. In: *Proceedings of 2005 IEEE swarm intelligence symposium*, Pasadena, CA, USA, 8–10 June 2005, pp. 333–340.

Bayindir L and Şahin E (2009) Modeling self-organized aggregation in swarm robotic systems. In: *Proceedings of 2009 IEEE swarm intelligence symposium*, Nashville, TN, USA, 30 March–2 April 2009, pp. 88–95.

Brambilla M, Ferrante E, Birattari M, et al. (2013) Swarm robotics: A review from the swarm engineering perspective. *Swarm Intelligence* 7: 1–41.

Brooks RA (1991) Intelligence without representation. *Artificial Intelligence* 47: 139–159.

Camazine S, Franks NR, Sneyd J, et al. (2001) *Self-Organization in Biological Systems*. Princeton, NJ: Princeton University Press.

Chow TY (1995) Penny-packings with minimal second moments. *Combinatorica* 15: 151–158.

Cieliebak M, Flocchini P, Prencipe G, et al. (2003) Solving the robots gathering problem. In: Baeten J, Lenstra J, Parrow J, et al. (eds) *Automata, Languages and Programming* (Lecture Notes in Computer Science, Vol. 2719). Berlin; Heidelberg: Springer-Verlag, pp. 192–192.

Correll N and Martinoli A (2011) Modeling and designing self-organized aggregation in a swarm of miniature robots. *International Journal of Robotics Research* 30: 615–626.

Cortés J, Martinez S and Bullo F (2006) Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automation and Control* 51: 1289–1298.

Damasceno PF, Engel M and Glotzer SC (2012) Predictive self-assembly of polyhedra into complex structures. *Science* 337: 453–457.

Deneubourg JL, Gregoire JC and Fort EL (1990) Kinetics of larval gregarious behavior in the bark beetle *Dendroctonus micans* (Coleoptera: Scolytidae). *Journal of Insect Behavior* 3: 169–182.

Dorigo M, Trianni V, Şahin E, et al. (2004) Evolving self-organizing behaviors for a swarm-bot. *Autonomous Robots* 17: 223–245.

Dudek G and Jenkin M (2010) *Computational Principles of Mobile Robotics*. 2nd edition. New York, NY: Cambridge University Press.

Fatès N (2010) Solving the decentralised gathering problem with a reaction–diffusion–chemotaxis scheme. *Swarm Intelligence* 4: 91–115.

Fitch R and Butler Z (2008) Million module march: Scalable locomotion for large self-reconfiguring robots. *International Journal of Robotics Research* 27: 331–343.

Flocchini P, Prencipe G, Santoro N, et al. (2005) Gathering of asynchronous robots with limited visibility. *Theoretical Computer Science* 337: 147–168.

Garnier S, Jost C, Gautrais J, et al. (2008) The embodiment of cockroach aggregation behavior in a group of micro-robots. *Artificial Life* 14(4): 387–408.

Gasparri A, Oriolo G, Priolo A, et al. (2012a) A swarm aggregation algorithm based on local interaction for multi-robot systems with actuator saturations. In: *Proceedings of 2012 IEEE/RSJ International conference on intelligent robots and systems*, Algarve, Portugal, 7–12 October 2012, pp. 539–544.

Gasparri A, Priolo A and Ulivi G (2012b) A swarm aggregation algorithm for multi-robot systems based on local interaction. In: *Proceedings of 2012 IEEE International conference on control applications*, Dubrovnik, Croatia, 3–5 October 2012, pp. 1497–1502.

Gauci M, Chen J, Dodd TJ, et al. (2012) Evolving aggregation behaviors in multi-robot systems with binary sensors. In: *Proceedings of 2012 International symposium on distributed autonomous robotic systems*, Baltimore, MD, USA, 8–11 November 2012.

Gauci M, Chen J, Li W, et al. (2014) Clustering objects with robots that do not compute. In: *Proceedings of 13th International conference on autonomous agents and multiagent systems*, Paris, France, 5–9 May 2014.

Gauci M, Li W, Chen J, et al. (2013) Online supplementary material. Available at: http://naturalrobotics.group.shef.ac.uk/supp/2013-001/.

Gazy V and Passino KM (2003) Stability analysis of swarms. *IEEE Transactions on Automation and Control* 48: 692–697.

Gennaro MD and Jadbabie A (2006) Decentralized control of connectivity for multi-agent systems. In: *Proceedings of 45th IEEE conference on decision and control*, San Diego, CA, USA, 13–15 December 2006, pp. 3628–3633.

Goldstein SC, Campbell JD and Mowry TC (2005) Programmable matter. *IEEE Computer* 38: 99–101.

Gordon N, Elor Y and Bruckstein AM (2008) Gathering multiple robotic agents with crude distance sensing capabilities. In: *Proceedings of 6th International conference on ant colony optimization and swarm intelligence* (Lecture Notes in Computer Science, volume 5217). Berlin; Heidelberg: Springer-Verlag, pp. 72–83.

Gordon N, Wagner IA and Bruckstein AM (2004) Gathering multiple robotic a(ge)nts with limited sensing capabilities. In: *Proceedings of 4th International Conference on ant colony optimization and Swarm Intelligence* (Lecture Notes in Computer Science, Volume 3172). Berlin; Heidelberg: Springer-Verlag, pp. 142–153.

Graham RL and Sloane NJA (1990) Penny-packing and two-dimensional codes. *Discrete Computational Geometry* 5: 1–11.

Groß R and Dorigo M (2008) Self-assembly at the macroscopic scale. *Proceedings of the IEEE* 96: 1490–1508.

Halloy J, Sempo G, Caprari G, et al. (2007) Social integration of robots into groups of cockroaches to control self-organized choices. *Science* 318: 1155–1158.

Jakobi N (1997) Half-baked, ad-hoc and noisy: Minimal simulations for evolutionary robotics. In: *Proceedings of the 4th European conference on artificial life*. Cambridge, MA: MIT Press, pp. 348–357.

Jeanson R, Rivault C, Deneubourg JL, et al. (2005) Self-organized aggregation in cockroaches. *Animal Behavior* 69: 169–180.

Kernbach S, Häbe D, Kernbach O, et al. (2013) Adaptive collective decision-making in limited robot swarms without communication. *International Journal of Robotics Research* 32: 35–55.

Kernbach S, Thenius R, Kernbach O, et al. (2009) Re-embodiment of honeybee aggregation behavior in an artificial micro-robotic system. *Adaptive Behavior* 17: 237–259.

Klavins E (2007) Programmable self-assembly. *IEEE Control Systems Magazine* 27(4): 43–56.

Knuth DE (1997) *The Art of Computer Programming. Volume 1: Fundamental Algorithms*. 3rd edition. Redwood City, CA: Addison Wesley Longman Publishing Co.

Leccese A, Gasparri A, Priolo A, et al. (2012) A swarm aggregation algorithm based on local interaction with actuator saturations and integrated obstacle avoidance. In: *Proceedings of 2013 IEEE/RSJ International conference on intelligent robots and systems*, Karlsruhe, Germany, 6–10 May 2013, pp. 1857–1862.

Li W (2008) Stability analysis of swarms with general topology. *IEEE Transactions on Systems, Man, and Cybernetics—A: Systems and Humans* 38: 1084–1097.

Litovsky I, Méivier Y and Zielonka W (1993) The power and limitations of local computations on graphs. In: *Graph-Theoretic Concepts in Computer Science* (Lecture Notes in Computer

Science, Volume 657). Berlin; Heidelberg: Springer-Verlag, pp. 333–345.
Magnenat S, Waibel M and Beyeler A (2011) Enki: The fast 2D robot simulator. Available at: http://home.gna.org/enki/.
Michel O (2008) Webots: Professional mobile robot simulation. *International Journal of Advanced Robotic Systems* 1(1): 39–42.
Mondada F, Bonani M, Raemy X, et al. (2009) The e-puck, a robot designed for education in engineering. In: *Proceedings of 9th conference on autonomous robot systems and competitions*, Volume 1, Castelo Branco, Portugal, 7 May 2009, pp. 59–65.
Nolfi S and Floreano D (2000) *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. Cambridge, MA: MIT Press.
Parker LE (2008) Multiple mobile robot systems. In: Siciliano B and Khatib O (eds) *Springer Handbook of Robotics* (Springer Handbooks). Berlin; Heidelberg: Springer-Verlag, pp. 921–941.
Parrish JK and Edelstein-Keshet L (1999) Complexity, pattern, and evolutionary trade-offs in animal aggregation. *Science* 284: 99–101.
Requicha A (2003) Nanorobots, NEMS, and nanoassembly. *Proceedings of the IEEE* 91: 1922–1933.
Sitti M (2007) Microscale and nanoscale robotics systems [grand challenges of robotics]. *IEEE Robotics and Automation Magazine* 14: 53–60.
Soysal O and Şahin E (2005) Probabilistic aggregation strategies in swarm robotic systems. In: *Proceedings of 2005 IEEE swarm intelligence symposium*, Pasadena, CA, USA, 8–10 June 2005, pp. 325–332.
Trianni V (2008) *Evolutionary Swarm Robotics*, (Studies in Computational Intelligence, Vol. 108). Berlin; Heidelberg: Springer-Verlag.
Yu J, LaValle S and Liberzon D (2012) Rendezvous without coordinates. *IEEE Transactions on Automation and Control* 57: 421–434.
Zebrowski P, Litus Y and Vaughan RT (2007) Energy efficient robot rendezvous. In: *Proceedings of 4th Canadian conference on computer and robot vision*, Montreal, QC, Canada, pp. 139–148.

Appendix A: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org

Table of Multimedia Extensions

Extension	Media Type	Description
1	Video	Aggregation of 40 physical e-puck robots and of 100 simulated e-puck robots.

Appendix B: Pathological initial configurations for Theorem 4.1

Pathological initial configurations for Theorem 4.1 are shown in Figures 17, 18 and 19. For more details, and variable definitions, see Sections 4.2 and 5.1.

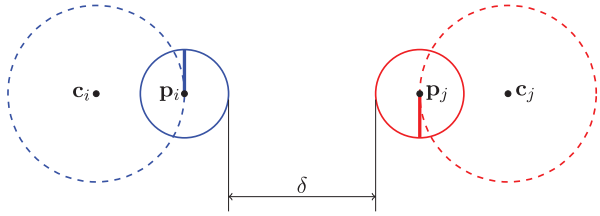

Fig. 17. A pathological initial configuration for the six cases $(B, *)$ and $(F, *)$.

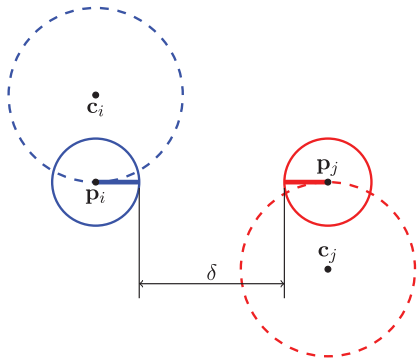

Fig. 18. A pathological initial configuration for the case (S, B).

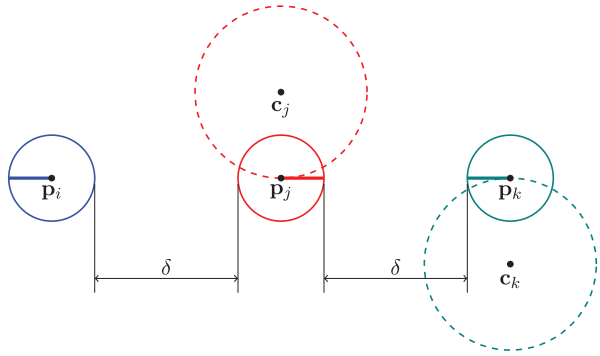

Fig. 19. A pathological initial configuration for the case (S, F).