# Shepherding with Robots That Do Not Compute

Anıl Özdemir[1], Melvin Gauci[2], and Roderich Groß[1]

[1]Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield, UK
[2]Wyss Institute for Biologically Inspired Engineering, Harvard University, Boston, MA
r.gross@sheffield.ac.uk

## Abstract

We examine the problem solving capabilities of swarms of computation- and memory-free agents. Each agent has a single line-of-sight sensor providing two bits of information. The agent maps this information directly onto constant motor commands. In previous work, we showed that such simplistic agents can solve tasks requiring them to organize spatially (multi-robot aggregation and circle formation) and manipulate passive objects (clustering). In the present work, we address the *shepherding problem*, where the computation- and memory-free agents—the shepherds—are tasked to gather and move a group of dynamic agents—the sheep—towards a pre-defined goal. The shepherds and sheep are modelled as e-puck robots using computer simulations. Our findings show that the shepherding problem does not fundamentally require arithmetic computation or memory to be solved. The obtained controller solution is robust with respect to sensory noise, and copes well with changes in the number of sheep.

## Introduction

The simplicity of individual robots has always been at the core of multi-robot systems research. It is intimately related with the fundamental advantages that multi-robot systems aim to offer in comparison to monolithic robot systems, namely robustness, scalability, and flexibility (Brambilla et al., 2013). Over the last two decades, several bodies of work have shown and reaffirmed that simple robots can solve complex tasks with a performance that matches—and in some cases even exceeds—that of complex, individual robots (Parker, 2008). However, the simplicity of the individual robots constituting a multi-robot system is usually measured in *relative* terms; that is, in comparison with the complexity of the task. As such, in many multi-robot systems to date the individual robots are still fairly complex machines, often combining various sensing modalities as well as powerful computation and communication capabilities. More recently, there has been a surging interest in exploring what is possible with robots that operate at a level of extreme simplicity; a question that had hitherto received little attention (Jones and Mataric, 2003; Groß and Dorigo, 2008; Yu et al., 2012; Becker et al., 2013; Rubenstein et al., 2013). A major motivation for such robotic systems is their potential feasibility for implementation at small scales, where more conventional robotic systems are not feasible to implement due to the acute constraints on the space available for hardware (Requicha, 2003).

In previous work, we introduced a framework to study the capabilities of robots of extreme *objective* simplicity. The robots lacked the ability to compute arithmetically and store information during run-time. They used a single line-of-sight sensor that returns discrete readings about the environment; namely, what the robot is instantaneously pointing towards. The sensor does *not* provide any other information about numbers of or distances to objects in the environment. This framework was first applied to the problems of multi-robot aggregation and circle formation [see Gauci et al. (2014b), and references therein], where a number of robots that are initially dispersed in the environment are required to organize spatially without using external cues. The framework was then applied to a more complex scenario in the form of object clustering (Gauci et al., 2014a). This time, the robots had to interact with static objects in the environment in order to bring *them* together into a single cluster. Johnson and Brown (2015) have applied this framework to some other problems, such as perimeter formation and foraging. Using novelty search, Brown et al. (2017) discovered that the framework can also be used to produce wall following, dispersal, and milling behaviors. In this paper, we show for the first time that the framework can be applied to scenarios where the minimalist robots are required to interact with other *active* agents in the environment.

We study the *shepherding problem*, which involves guiding the motion of multiple dynamic *sheep* agents by one or more *shepherd* agents towards a pre-specified goal location. Aside from the shepherding of actual sheep by dogs, this problem, in a more general setting, has other manifestations in nature—one example is human crowd control in large-scale events by trained officials. Potential robotics applications involving this task include: containing oil spillages (oil on the surface of water behaves as a dynamic entity), manipulation of micro-organisms such as bacteria, and other uses in nanomedicine (Requicha, 2003). We now provide a brief

Carole Knibbe et al, eds., Proceedings of the ECAL 2017, Lyon, France, 4-8 September 2017,
(Cambridge, MA: The MIT Press, ©2017 Massachusetts Institute of Technology).

overview of related work in robotics that has addressed the shepherding problem.

One of the earliest studies on the shepherding problem was conducted by Vaughan et al. (2000). They developed a controller strategy to herd (real) ducks using a single robot. The controller required an external camera system for tracking the robot's position and orientation. This, along with information on the center of mass and size of the flock of ducks, was computed to provide quasi-instantaneous path planning for the robot to herd the ducks towards a goal location. Lien et al. (2005) proposed and analyzed several formations that the shepherds can assume. Their work suggested that multiple robotic shepherds are superior to a single one in solving the problem. Strömbom et al. (2014) developed an algorithm based on empirical data from sheep/dog interactions. This algorithm contains two steps: firstly, the shepherds gather a dispersed flock of sheep; secondly, they herd them to a goal location. Pierson and Schwager (2015) proposed another controller for a multi-robot system where the robots have non-holonomic constraints (a unicycle model). Using Lyapunov theory, they proved that this controller is always guaranteed to herd the flock to the goal location.

In all of these works, the controller strategies required the shepherd agents to have memory, and be able to perform arithmetic computations. In addition, they required the sensors to provide distance information. In contrast, we here present a minimalist controller strategy for the shepherd agents that eliminates all these requirements.

This paper is organized as follows. We first present the methods, including the problem formulation, details about the simulations of shepherds and sheep, and the optimization method used for synthesizing the shepherd controllers. We then evaluate the controller obtained and discuss the results obtained from simulation experiments.

# Methods

## Problem Formulation

Consider an unbounded, continuous-space environment containing $m \geq 1$ *shepherd* agents, $n \geq 1$ *sheep* agents, and an object representing a *goal location*. The sheep and shepherds are initially dispersed and away from the goal location. The shepherding problem is to control the shepherds such that they gather the sheep and herd them towards the goal location.

## Simulation Setup

We use the open-source physics library Enki (Magnenat et al., 2009), which simulates the dynamics and kinematics of rigid bodies in two dimensions. Space is represented continuously (with floating point precision). The physics are updated at $0.01\,\mathrm{s}$ intervals.

The shepherds and sheep are modelled as e-puck robots (Mondada et al., 2009). The robots are represented as cylinders of radius $3.7\,\mathrm{cm}$. In order to allow the line-of-sight sensor to make distinctions, the shepherds are colored green and the sheep are colored red. The goal is represented as a blue cylinder and has a radius of $22.2\,\mathrm{cm}$, which is six times larger than the radius of the e-puck.

The robots have two wheels arranged in a differential drive configuration (axle length $= 5.2\,\mathrm{cm}$). The control cycle of the robot is activated every $0.1\,\mathrm{s}$. The robot needs to set the desired velocities of its left and right wheels, $v_\ell \in [-1, 1]$ and $v_r \in [-1, 1]$, where $-1$ and $1$ represent the normalized maximum angular velocity at which the wheel can turn backward and forward, respectively[1]. The corresponding velocity of the robot ranges in $\pm 12.8\,\mathrm{cm/s}$.

## Shepherd

The shepherd has a line-of-sight sensor pointing forwards. The range of the sensor is unlimited[2]. The sensor reading, $I$, is defined by the first object that is detected in the line of sight, if any:

$$I = \begin{cases} 0 & \text{if no object is detected,} \\ 1 & \text{if a red object (sheep) is detected,} \\ 2 & \text{if a green object (shepherd) is detected,} \\ 3 & \text{if the blue object (goal) is detected.} \end{cases} \quad (1)$$

Note that sensor reading $I$ does not contain any information about the distance to a perceived object, or about the number of objects in a given area.

All shepherds execute an identical controller. The controller is reactive: it maps the shepherd's input $I$ onto its output—the pair of wheel velocities, $(v_\ell, v_r)$. Formally, the controller is defined by:

$$\mathbf{v} = (v_{\ell,0}, v_{r,0}, v_{\ell,1}, v_{r,1}, v_{\ell,2}, v_{r,2}, v_{\ell,3}, v_{r,3}) \in [-1, 1]^8, \quad (2)$$

where the left and right wheel velocities for sensor reading $I = 0$ are denoted by $v_{\ell,0}$ and $v_{r,0}$, and so on. Note that the controller does not need to store information during runtime, and does not need to perform arithmetic computations.

The velocities in Eqn. (2) are free parameters that need to be tuned in order for the swarm to produce the desired global behavior (in this case, shepherding). The method for optimizing these parameters will be discussed in the sections *Optimization Method* and *Objective Function*.

## Sheep

In contrast to the shepherd agents, the behavior of the sheep agents is not subjected to an optimization process. Rather, the sheep agents execute a fixed, manually-designed behavior in which they react both to each other and to shepherd agents. This behavior is based on the magnitude-dependent

[1] Uniform noise of $5\,\%$ is applied to each value.

[2] In a practical scenario, the environment could be bounded. The sensor would then need to span the entire environment.

Carole Knibbe et al, eds., Proceedings of the ECAL 2017, Lyon, France, 4-8 September 2017,
(Cambridge, MA: The MIT Press, ©2017 Massachusetts Institute of Technology).

motion control model proposed by Ferrante et al. (2012). The sheep have omnidirectional vision[3], and can distinguish between shepherds and other sheep; however, they cannot see the goal. Each sheep is repelled by all shepherds and—to a lesser extent—by other sheep[4]. Let $S$ be the set of (suitably relabelled) indices of all agents. Formally, the repulsion force is given by

$$\mathbf{F}_i = \sum_{k \in S \setminus \{i\}} \frac{c_k}{\|\mathbf{x}_i - \mathbf{x}_k\|^2} \hat{\mathbf{r}}_{ki}, \tag{3}$$

where $\mathbf{x}_i$ is the position of sheep $i$, $\mathbf{x}_k$ is the position of agent $k$ (either a shepherd or a sheep, but excluding the focal sheep), $\hat{\mathbf{r}}_{ki}$ is the unit vector pointing from agent $k$ towards sheep $i$, and $c_k$ is $450$ if agent $k$ is a shepherd, and $100$ otherwise. In other words, the sheep repel more strongly from the shepherds than from other sheep.

The motion of each sheep is the result of (i) the repulsion force and (ii) a natural tendency to move forward. Formally,

$$\begin{pmatrix} v_\ell \\ v_r \end{pmatrix} = \begin{pmatrix} K_1 & K_2 \\ K_1 & -K_2 \end{pmatrix} \begin{pmatrix} f_x \\ f_y \end{pmatrix} + \begin{pmatrix} u \\ u \end{pmatrix}, \tag{4}$$

where $f_x$ and $f_y$ are the horizontal and vertical components of the repulsion force in the sheep's local coordinate frame, $K_1 = 2.0$ and $K_2 = 1.3$ are the linear and angular gain, and $u = 2.0\,\mathrm{cm/s}$ is the constant forward speed. The maximum speed for a sheep is $6.4\,\mathrm{cm/s}$—which is half of a shepherd's maximum speed. If the velocities exceed their range, they get truncated.

## Optimization Method

We have so far devised the structure of a reactive controller for the shepherds (Eqn. (2)). The remaining problem is to optimize the eight free parameters in this controller such that it leads to the desired global behavior (i.e., shepherding). We employ an evolutionary robotics approach to this problem (Nolfi and Floreano, 2000; Trianni et al., 2008), whereby an optimizer searches for the best controller parameters that maximize an objective function.

As an optimizer, we use the covariance matrix adaptation-evolution strategy (CMA-ES) (Hansen and Ostermeier, 2001). CMA-ES is a stochastic optimization algorithm and operates on real-valued decision variables. It self-adapts the variance of each decision variable, as well as all the covariances between the decision variables.

In our problem, the decision variables are the set of all possible left and right wheel speeds corresponding to the sensor readings of the shepherd robots (Eqn. (2)). Considering normalized wheel speeds, this corresponds to the space $[-1, 1]^{2d}$, where $d$ is the number of possible sensor states. In its original version, CMA-ES operates in unconstrained, real space: $\mathbb{R}^{2d}$. Therefore we need to perform a mapping from the candidate solutions provided by CMA-ES onto valid controllers. We achieve this by applying the following sigmoid-based function to each value in the candidate solution:

$$\mathrm{sig}(x) = \frac{1 - e^{-x}}{1 + e^{-x}}, \quad \forall x \in \mathbb{R}. \tag{5}$$

The only external parameters that are required by the CMA-ES algorithm are the following: a population size $\lambda$, an initial guess of a solution, $\mathbf{m}^{(0)}$, and an initial step size $\sigma^{(0)}$. We set the population size to $\lambda = 20$. We set $\mathbf{m}^{(0)} = \mathbf{0}$ and $\sigma^{(0)} = 0.72$. Using Monte Carlo simulations, Gauci et al. (2014a) reported that these settings provide an approximately uniform distribution over $[-1, 1]^{2d}$ in the initial generation.

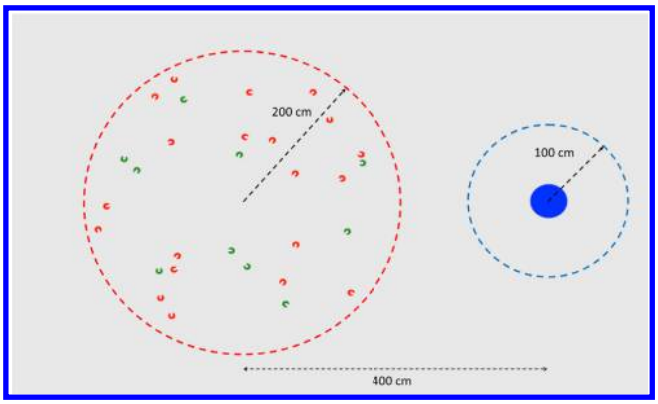



Figure 1: Illustration of the experimental setup. The goal object is represented by the blue disk, the shepherds and the sheep are indicated by the green and the red disks, respectively. Initially, the robots are randomly distributed into a circular region of radius $200\,\mathrm{cm}$ and $400\,\mathrm{cm}$ away from the goal object. The goal region is the circular area indicated by the blue dashed line with radius $100\,\mathrm{cm}$ from the center of the goal object.

## Objective Function

The optimization method requires that each candidate solution (i.e., controller) is assigned a value reflecting its quality (hereafter referred to as *fitness*) that reflects how well it addresses the problem. This is achieved by running a number of simulations using the controller and computing an objective function (hereafter referred to as *fitness function*) based on the performance of the agents during these simulations. Figure 1 shows the simulation setup. Initially, the shepherds and the sheep are distributed uniformly randomly within a circular region of radius $200\,\mathrm{cm}$, whose center is $400\,\mathrm{cm}$ away from the goal.

The fitness function for a single simulation—to be

[3] It is typical for natural "prey" to have very wide fields of view (Piggins and Phillips, 1996).

[4] Although it may be unrealistic to assume that a sheep could perceive all other agents in the environment, the magnitude of the repulsive force decreases as the square of distance, and hence its effect can be neglected when the other agent is far away.

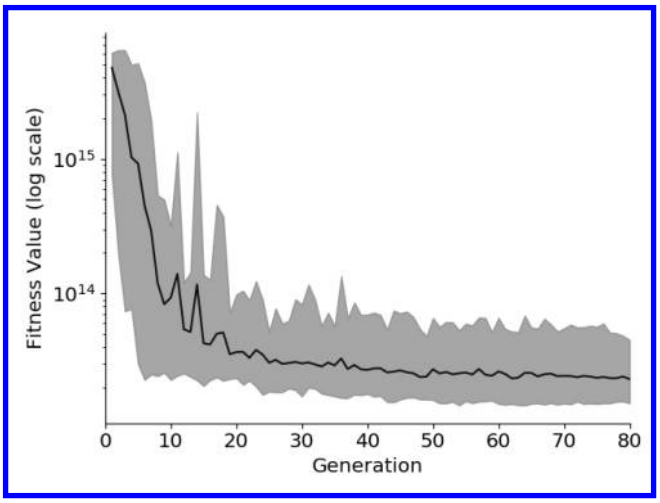



Figure 2: Evolutionary dynamics. Best fitness (solid line), averaged over 30 evolutionary runs. The envelope indicates the minimum and the maximum fitness values in each generation.

minimized—is given by

$$F(T) = \sum_{t=1}^{T} t f(t), \tag{6}$$

where $T$ is the duration of the simulation and

$$f(t) = \frac{1}{4nr^2} \sum_{i=1}^{n} \|\bar{\mathbf{x}}(t) - \mathbf{x}_i(t)\|^2 \|\bar{\mathbf{x}}(t) - \mathbf{g}\|^2, \tag{7}$$

where $n$ is the number of sheep, $r$ is the radius of the agents, $\mathbf{x}_i(t)$ is the position of sheep $i$ at time $t$, $\bar{\mathbf{x}}(t)$ is the centroid of the sheep flock at time $t$, and $\mathbf{g}$ is the position of the goal. The multiplier outside the summation normalizes for $n$ and $r$. Eqn. (7) takes into account how widely the sheep are scattered and how far away they are from the goal. The weighted summation over time in Eqn. (6) rewards solutions for accomplishing the task faster, while still giving prominence to a stable configuration later on in the simulation.

The overall fitness of a controller is given by averaging $F(T)$ over a number $N$ of simulations with different initial conditions.

## Results

A set of 30 evolutions were performed with $m = 10$ shepherds and $n = 20$ sheep. Each evolution was run for 80 generations, and in each generation, each of the $\lambda = 20$ candidate solutions was evaluated using $N = 50$ trials with different initial configurations of agents. Each trial returned a fitness value according to Eqn. (6), and the mean of these 50 values was used as the overall fitness of that candidate solution. Each trial lasted 1500 s (i.e., $T = 15000$ in Eqn. (6)).

Figure 2 shows the evolutionary dynamics. During the first approximately 20 generations, the fitness values of the best individuals improve rapidly. The fitness values continue to improve thereafter, but seem to approach a stable plateau.

| $v_{\ell,0}$ | $v_{\ell,1}$ | $v_{\ell,2}$ | $v_{\ell,3}$ |
|---|---|---|---|
| 0.9998 | 0.0082 | 0.5471 | 0.9993 |
| $v_{r,0}$ | $v_{r,1}$ | $v_{r,2}$ | $v_{r,3}$ |
| 0.8520 | 0.9996 | 0.6098 | 0.9447 |

Table 1: The best controller found by the optimization method.

### Selecting the Best Controller

Throughout an evolution, a population consists of $\lambda = 20$ candidate solutions (i.e., shepherd controllers). The candidate solution with the best fitness in the last generation was considered as the best controller for that evolution. As 30 evolutions were performed, there were 30 best controller candidates in total. In a post-evaluation session, each of these 30 controllers was reevaluated 100 times in simulations with different initial configurations of agents. The controller with the best average fitness was selected, and is considered as the *best controller* across the set of evolutions. Its parameter values are provided in Table 1.

### Behavioral Analysis

Figure 3 shows a sequence of snapshots taken from a simulation trial with the best controller. At the beginning, the shepherds spread out from the initial formation towards the periphery. They then cage the sheep by orbiting around them in a clockwise manner. As the sheep are repelled more by the shepherds than by each other, they assume a compact, round formation. While orbiting around the sheep, the shepherds are also attracted towards the goal. This results in the gradual movement of the agents (i.e., shepherds and sheep) towards the goal.

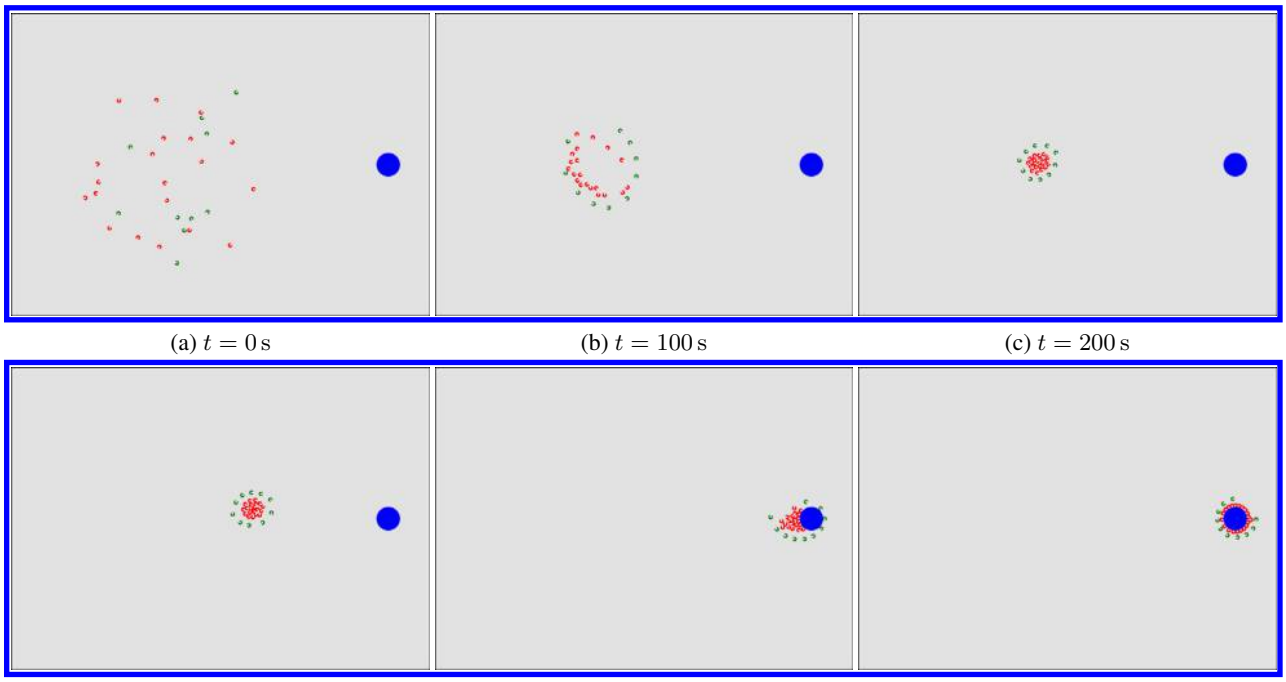
To gain a deeper understanding of the shepherding behavior, we monitored the sensor reading values of shepherds over 100 additional simulation trials. Note that the reading values directly determine a shepherd's action (i.e., wheel velocities) according to Eqn. (2). Figure 4 shows the average number of shepherds detecting objects of different types. Shepherds most often detect nothing, followed by other shepherds. The detection of sheep seems to be relevant only in the initial stages—prior to the caging being completed. The goal becomes more frequently observed as the agents approach it, until the agents are caging it as well.

### Noise Analysis

In the following sections, we explore the capabilities of the controller through noise, sensitivity, and scalability analyses. In these analyses, a *success rate* is used to measure the performance of the shepherds. The success rate is defined as the percentage of sheep that reside within the goal region (see Figure 1) after 1500 s.



Carole Knibbe et al, eds., Proceedings of the ECAL 2017, Lyon, France, 4-8 September 2017,
(Cambridge, MA: The MIT Press, ©2017 Massachusetts Institute of Technology).



(a) $t = 0\,\mathrm{s}$ (b) $t = 100\,\mathrm{s}$ (c) $t = 200\,\mathrm{s}$

(d) $t = 400\,\mathrm{s}$ (e) $t = 600\,\mathrm{s}$ (f) $t = 1500\,\mathrm{s}$

Figure 3: Sequence of snapshots showing how a group of 10 shepherds gather and move a group of 20 sheep towards the goal.

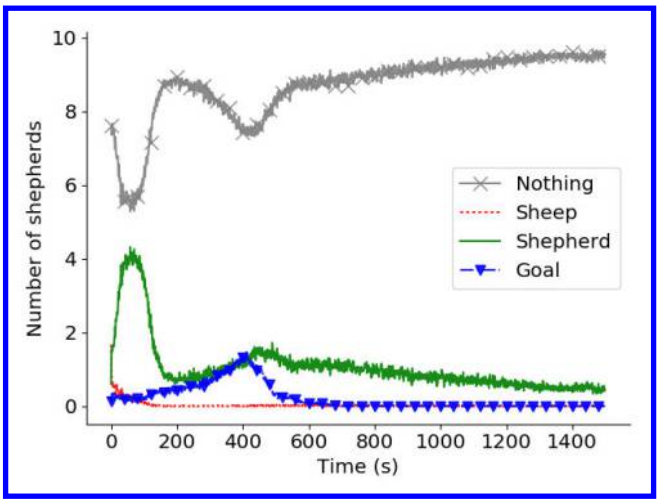



Figure 4: Number of shepherds that detect objects of particular types (averaged over 100 trials with 10 shepherds and 20 sheep). The shepherds can either detect nothing ($I = 0$), sheep ($I = 1$), other shepherds ($I = 2$), or the goal ($I = 3$).

We examine the effect of noise on the performance of shepherds. In particular, we consider false negative noise; in other words, noise preventing the detection of objects (i.e., sheep, shepherd, or goal). Formally, given an unperturbed sensor reading of $I \in \{1, 2, 3\}$, the actual reading value returned by the sensor is 0 with probability $p$, and $I$ otherwise. If no object is in the line of sight of the shepherd, the sensor value remains $I = 0$. We performed 100 simulation trials for each of $p \in \{0, 0.1, 0.2, ..., 1\}$. In addition, we performed an equivalent number of trials for the situation where only a single type of sensor reading (e.g., $I = 1$) was affected by the noise.

Figure 5 show the success rates for the different probability levels of noise. The success rate decreases rapidly if the sensor is subjected to noise levels of more than $p = 0.3$ on *all* readings (black curve). However, the impact of noise varies if only certain readings are subjected to it. For example, if the shepherds cannot reliably detect the sheep (red curve) the performance is better than if they cannot reliably detect the goal (blue curve). The shepherds are also tolerant to the situation where they cannot reliably detect each other (green curve)—even at a noise level of $p = 0.5$, they succeed in solving the task cooperatively without any significant degradation in performance.

## Sensitivity Analysis

We examine how sensitive the controller's performance is with respect to changes in its parameters. Each of the eight controller parameters was varied from $-1$ to 1 in steps of

Carole Knibbe et al, eds., Proceedings of the ECAL 2017, Lyon, France, 4-8 September 2017,
(Cambridge, MA: The MIT Press, ©2017 Massachusetts Institute of Technology).

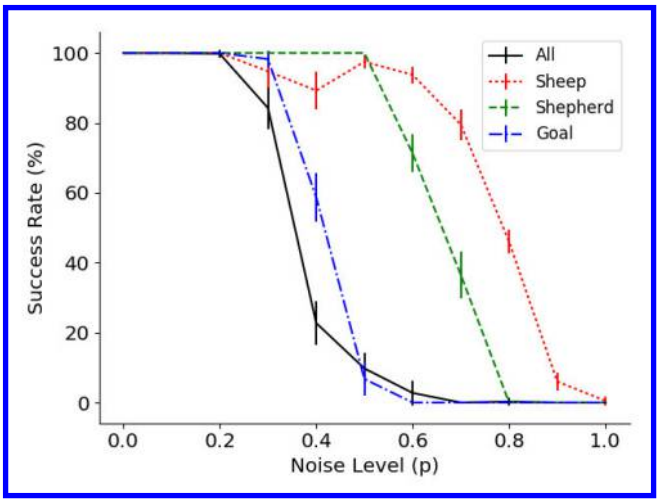



Figure 5: Noise analysis. The colored curves represent different types of sensor readings that experience noise (see the text for details). Error bars represent standard deviations.

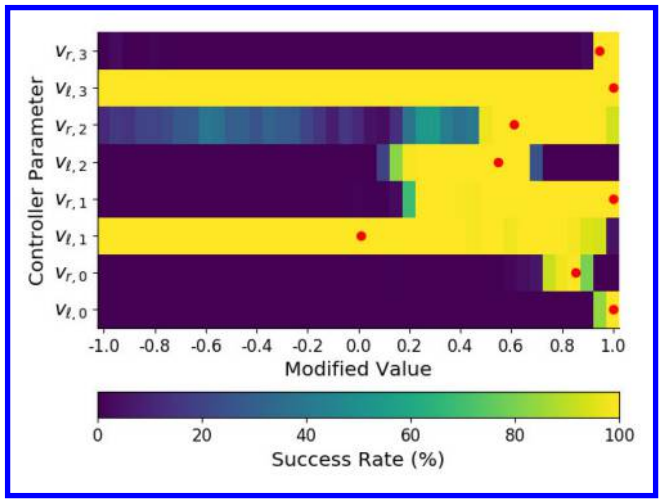



Figure 6: Sensitivity analysis. Each of the eight velocity parameters is varied from $-1$ to 1 while the remaining seven parameters are kept constant. The color map shows the average success rate across 100 trials. The red circles represent the unchanged parameters of the best controller, as shown in Table 1.

0.05, with the other seven parameters remained fixed according to Table 1. For each parameter configuration, 100 trials were performed.

Figure 6 shows the average success rate obtained in the sensitivity analysis trials. The controller is sensitive to the parameters associated with the sensor reading for nothing ($v_{\ell,0}$ and $v_{r,0}$). This is the most commonly observed sensor reading according to Fig. 4. On the other hand, the shepherd's motion when it detects a sheep ($I = 1$) is not highly critical, as long as $v_{\ell,1} < v_{r,1}$ (i.e., the robot turns left). When the shepherd detects the goal ($I = 3$), surprisingly, the velocity of the left wheel, $v_{\ell,3}$, is not critical. For $v_{\ell,3} < v_{r,3}$, a distinct strategy emerges, where shepherds orbit around both sheep and goal throughout the trial. When the shepherd detects another shepherd, there is some leeway in sensitivity as long as $v_{\ell,2} \leq v_{r,2}$, but the margin is smaller than for the sheep and goal cases.

## Scalability Analysis

We examine the performance of the shepherd's controller in situations where the number of shepherds ($m$) and/or sheep ($n$) are different with respect to the standard conditions. The numbers of shepherds considered were $\{5, 10, 15, 20, 30, 40\}$. The numbers of sheep considered were $\{10, 20, \ldots, 100\}$. For each combination of $m$ and $n$, 100 simulation trials with the best controller were performed. Each trial lasted 1500 s. The initialization region for all robots remained the same as shown in Figure 1.

Figure 7(a) shows the success rate. Due to the dynamic interactions, the relation between success rate and the number of agents can be nonlinear. The performance of the shepherd's controller scales well up to 70 sheep and 10 shepherds (which is the number of shepherds that was used during optimization). Beyond this point, however, the performance degrades. As the number of other agents in the environment increases, it becomes less likely for the shepherd to detect the goal. We hypothesize that the reason for the drop in performance could be that the sight of the goal is occluded for most of the time.

To alleviate the problem, we equipped the shepherd with a dedicated goal sensor, which can detect the goal even if behind some agents[5]. In this new setup, the shepherd can distinguish between six sensory states. Accordingly, the sensor reading, $I$, can be redefined by:

$$I = \begin{cases} 0 & \text{if no object is detected,} \\ 1 & \text{if a sheep is detected,} \\ 2 & \text{if a shepherd is detected,} \\ 3 & \text{if only the goal is detected,} \\ 4 & \text{if both a sheep and the goal are detected,} \\ 5 & \text{if both a shepherd and the goal are detected.} \end{cases} \tag{8}$$

Note that the sensor is unable to detect a sheep or shepherd if located "behind" the goal.

We conducted 30 evolutions using 10 shepherds (with the 6-state sensor) and 20 sheep. We refer to the best controller as the *extended controller*. Figure 7(b) shows its success rate. The performance of the extended controller scales far better with the number of sheep and shepherds.

Figure 8 shows the behavior of the shepherds through a sequence of snapshots taken from a simulation trial using the extended controller. It can be observed that it is similar to the

[5]In practice, this is achievable if the goal object is significantly taller than the robots.



Carole Knibbe et al, eds., Proceedings of the ECAL 2017, Lyon, France, 4-8 September 2017,
(Cambridge, MA: The MIT Press, ©2017 Massachusetts Institute of Technology).

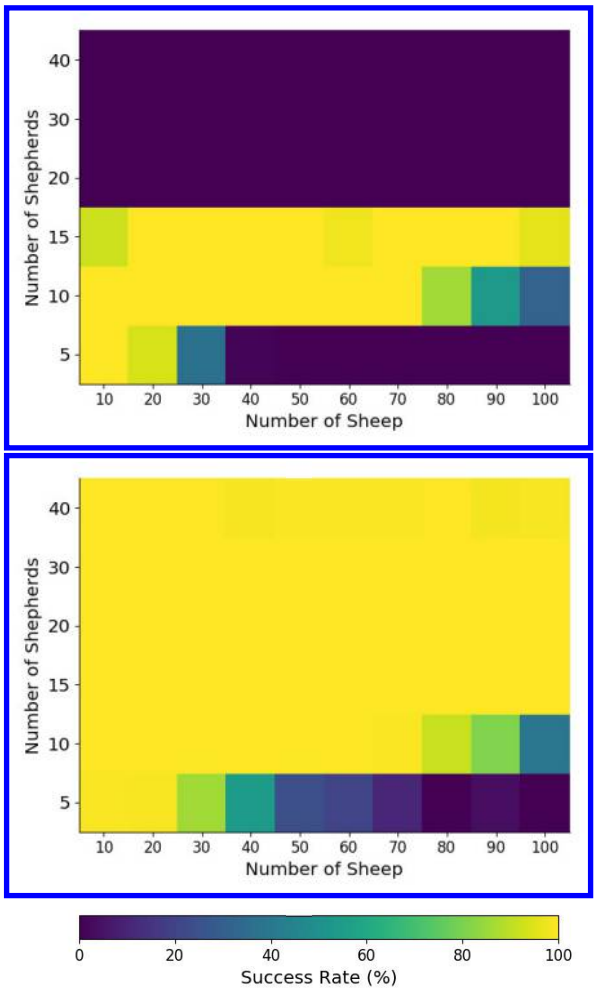



Figure 7: Success rates (percentage of sheep retrieved to the goal region, averaged over 100 trials) for different numbers of shepherds and sheep. (a) Controller optimized for a 4-state sensor; (b) controller optimized for a 6-state sensor.

behavior using the best controller. The process of herding 100 sheep takes about twice the time than herding 20 sheep. Please check the supplementary materials (Özdemir et al., 2017) for demo videos.

The experimental setup can be modified to have a dynamic goal location. The controller still performs successfully (see the supplementary materials for videos).

## Conclusions

This paper showed for the first time that the problem of herding a group of dynamic agents can be addressed by a control group of embodied agents that lack the ability to compute. The controlling agents needed only to extract 2 bits of information from their environment—what object they first detect in their line of sight, if any. The controller directly mapped this information onto constant motor commands. This was sufficient to move the controlled group reliably to a goal region. The controller solution, which was obtained by an evolutionary algorithm, was robust with respect to sensory noise. It was also flexible with respect to moderate changes in the number of shepherds and sheep. We also investigated shepherds using 2 trits (i.e., 2 ternary digits) of information. These shepherds were able to simultaneously detect another agent and the goal, if in the line of sight. This enhanced sensing modality yielded better scalability with respect to the number of sheep and shepherds.

Although various solutions to the collective shepherding problem had been previously proposed, minimizing the information that needs to be gathered and processed by the individual agents could help pave the way for applications at small scale—such as in nanomedicine—where the space and energy available for hardware is at a premium. An example of the shepherding concept in this domain can be observed in the manner in which white blood cells chase and engulf pathogens in the body. Future work will study to generalize our shepherding strategy to operate in more realistic $3D$ environments.

## References

Becker, A., Habibi, G., Werfel, J., Rubenstein, M., and McLurkin, J. (2013). Massive uniform manipulation: Controlling large populations of simple robots with a common input signal. In *Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 520–527. IEEE.

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

Brown, D. S., Turner, R., Hennigh, O., and Loscalzo, S. (2017). Discovery and exploration of novel swarm behaviors given limited robot capabilities. In *Proceedings of the 13th International Symposium on Distributed Autonomous Robotic Systems*. Springer.

Ferrante, E., Turgut, A. E., Huepe, C., Stranieri, A., Pinciroli, C., and Dorigo, M. (2012). Self-organized flocking with a mobile robot swarm: A novel motion control method. *Adaptive Behavior*, 20(6):460–477.

Gauci, M., Chen, J., Li, W., Dodd, T. J., and Groß, R. (2014a). Clustering objects with robots that do not compute. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multiagent Systems*, pages 421–428. IFAAMAS.

Gauci, M., Chen, J., Li, W., Dodd, T. J., and Groß, R. (2014b). Self-organized aggregation without computation. *The International Journal of Robotics Research*, 33(8):1145–1161.

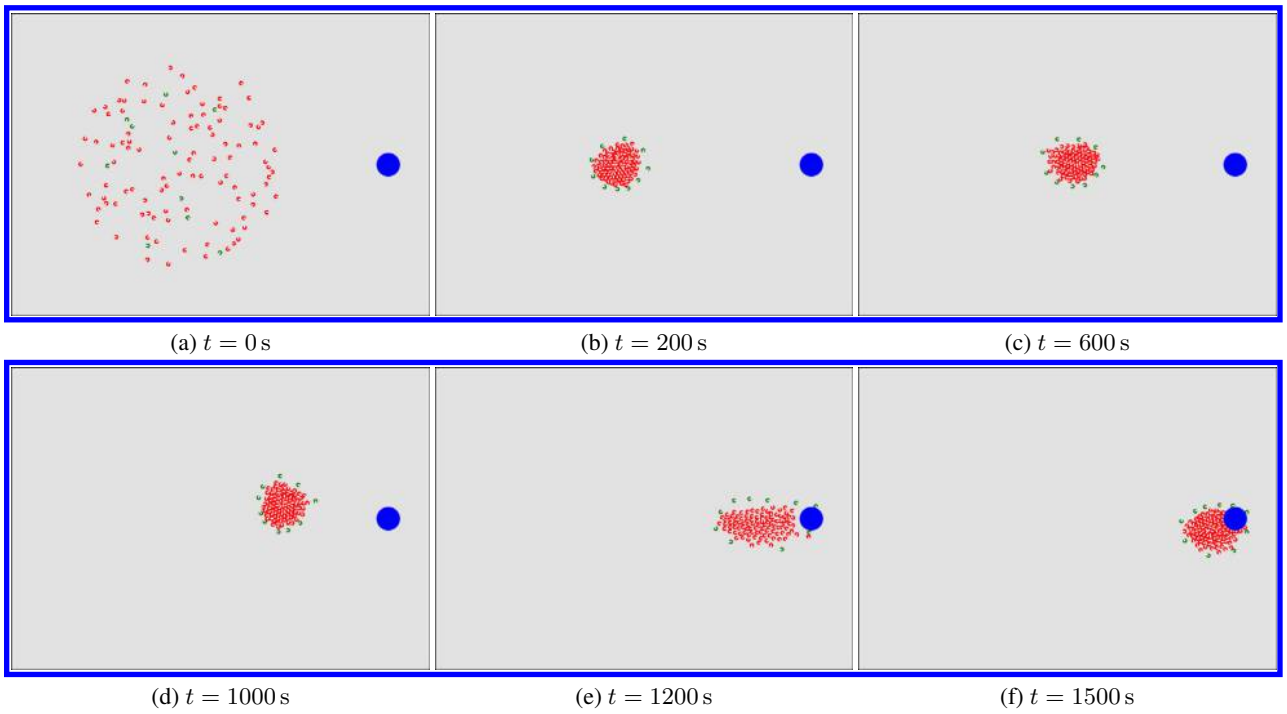
Groß, R. and Dorigo, M. (2008). Evolution of solitary and group transport behaviors for autonomous robots capable of self-assembling. *Adaptive Behavior*, 16(5):285–305.

Hansen, N. and Ostermeier, A. (2001). Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195.

Johnson, M. and Brown, D. S. (2015). Evolving and controlling perimeter, rendezvous, and foraging behaviors in a



Carole Knibbe et al, eds., Proceedings of the ECAL 2017, Lyon, France, 4-8 September 2017,
(Cambridge, MA: The MIT Press, ©2017 Massachusetts Institute of Technology).



(a) $t = 0\,\mathrm{s}$ (b) $t = 200\,\mathrm{s}$ (c) $t = 600\,\mathrm{s}$

(d) $t = 1000\,\mathrm{s}$ (e) $t = 1200\,\mathrm{s}$ (f) $t = 1500\,\mathrm{s}$

Figure 8: Sequence of snapshots showing how a group of 10 shepherds gather and move a group of 100 sheep towards the goal using the extended controller.


computation-free robot swarm. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies*, pages 311–314. ICST, Brussels, Belgium.

Jones, C. and Mataric, M. J. (2003). Adaptive division of labor in large-scale minimalist multi-robot systems. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1969–1974. IEEE.

Lien, J.-M., Rodriguez, S., Malric, J.-P., and Amato, N. M. (2005). Shepherding behaviors with multiple shepherds. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3402–3407. IEEE.

Magnenat, S., Waibel, M., and Beyeler, A. (2009). Enki: An open source fast 2D robot simulator. `http://home.gna.org/enki/`.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.-C., Floreano, D., and Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, volume 1, pages 59–65.

Nolfi, S. and Floreano, D. (2000). *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. MIT press.

Özdemir, A., Gauci, M., and Groß, R. (2017). Online supplementary material. `http://naturalrobotics.group.shef.ac.uk/supp/2017-002/`.

Parker, L. E. (2008). Multiple mobile robot systems. In *Springer Handbook of Robotics*, pages 921–941. Springer.

Pierson, A. and Schwager, M. (2015). Bio-inspired non-cooperative multi-robot herding. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, pages 1843–1849.

Piggins, D. and Phillips, C. (1996). The eye of the domesticated sheep with implications for vision. *Animal Science*, 62(2):301–308.

Requicha, A. A. (2003). Nanorobots, NEMS, and nanoassembly. *Proceedings of the IEEE*, 91(11):1922–1933.

Rubenstein, M., Cabrera, A., Werfel, J., Habibi, G., McLurkin, J., and Nagpal, R. (2013). Collective transport of complex objects by simple robots: Theory and experiments. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multiagent Systems*, pages 47–54. IFAAMAS.

Strömbom, D., Mann, R. P., Wilson, A. M., Hailes, S., Morton, A. J., Sumpter, D. J., and King, A. J. (2014). Solving the shepherding problem: Heuristics for herding autonomous, interacting agents. *Journal of the Royal Society Interface*, 11(100):20140719.

Trianni, V., Nolfi, S., and Dorigo, M. (2008). *Evolution, self-organization and swarm robotics*, pages 163–191. Springer.

Vaughan, R., Sumpter, N., Henderson, J., Frost, A., and Cameron, S. (2000). Experiments in automatic flock control. *Robotics and Autonomous Systems*, 31(1):109–117.

Yu, J., LaValle, S. M., and Liberzon, D. (2012). Rendezvous without coordinates. *IEEE Transactions on Automatic Control*, 57(2):421–434.




Carole Knibbe et al, eds., Proceedings of the ECAL 2017, Lyon, France, 4-8 September 2017,
(Cambridge, MA: The MIT Press, ©2017 Massachusetts Institute of Technology).