Finding Consensus Without Computation

Anıl Özdemir, *Student Member, IEEE*, Melvin Gauci, Salomé Bonnet, and Roderich Groß, *Senior Member, IEEE*

***Abstract*—A canonical problem for swarms of agents is to collectively choose one of multiple options in their environment. We present a novel control strategy for solving this problem—the first to be free of arithmetic computation. The agents do not communicate with each other nor do they store run-time information. They have a line-of-sight sensor that extracts one ternary digit of information from the environment. At every time step, they directly map this information onto constant-value motor commands. We evaluate the control strategy with both simulated and physical e-puck robots. By default, the robots are expected to choose, and move to, one of two options of equal value. The simulation studies show that the strategy is robust against sensory noise, scalable to large swarm sizes, and generalizes to the problems of choosing between more than two options or between unequal options. The experiments—50 trials conducted with a group of 20 e-puck robots—show that the group achieves consensus in 96% of the trials. Given the extremely low hardware requirements of the strategy, it opens up new possibilities for the design of swarms of robots that are small in size ($\ll 10^{-3}$ m) and large in numbers ($\gg 10^3$).**

***Index Terms*—Swarms, behavior-based systems, distributed robot systems, multi-robot systems, collective choice.**

I. Introduction

THE ability of groups of decision-making agents to reach consensus has been studied in a range of disciplines [1]–[3]. In general, a group of agents (e.g., humans, animals, robots) are operating in an environment that presents multiple options to choose from. The agents have some means of accessing information about these options, and of influencing each other. For example, some ant species use pheromone trails to select the most efficient path to a food source from multiple options [4]. In addition, some ant and bee species perform "house hunting", in which they collectively select and move to a new nest site [5]. The objective for the agents is to reach an agreement on which option to choose. In the following, we refer to this problem as the *collective choice* problem.

Manuscript received September 10, 2017; accepted December 31, 2017. Date of publication January 23, 2018; date of current version February 22, 2018. This paper was recommended for publication by Associate Editor S. Berman and Editor N. Y. Chong upon evaluation of the reviewers' comments. *(Corresponding author: Roderich Groß.)*

A. Özdemir, S. Bonnet, and R. Groß are with the Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield S10 2TN, U.K. (e-mail: a.ozdemir@sheffield.ac.uk; s.bonnet@sheffield.ac.uk; r.gross@sheffield.ac.uk).

M. Gauci is with the Wyss Institute for Biologically Inspired Engineering, Harvard University, Boston, MA 02138 USA (e-mail: mgauci@seas.harvard.edu).

This letter has supplemental downloadable multimedia material available at http://ieeexplore.ieee.org, provided by the authors. The Supplementary Materials contain a video of two parts. The first part shows a typical simulation trial of a swarm of 20 robots. The second part shows a typical experimental trial with the same number of robots. This material is 2.29 MB in size.

Digital Object Identifier 10.1109/LRA.2018.2795640

In this letter, we study the collective choice problem with a group of simplistic robotic agents, and are interested in the situation where:

1) The group of agents is homogeneous, in other words, all agents are identical;
2) There are two or more options to choose from;
3) The options are of equal value,[1] so the group of agents is no better off choosing one over the other;
4) The agents' environment does not contain any cues that could help or hinder the selection process.

The agents may therefore only rely on sensing and/or communicating with each other for making a decision. A number of solutions have been proposed for similar collective choice problems [6]–[10], including generalizations, such as the best-of-*n* problem [11]. Valentini *et al.* [12] present a detailed review of the best-of-*n* problem in a swarm robotics context.

Halloy *et al.* [6] used robots to explore the collective choice problem in cockroaches. Naturally, cockroaches prefer darker shelters over lighter ones. The researchers introduced a group of robots coated with pheromone such that they were accepted by a group of cockroaches as conspecifics. The robots were programmed to "[...] explore their environment autonomously [and] tune their resting time [in the shelters] in relation to the presence of cockroaches, as cockroaches do" [6]. The robots, being programmed to prefer the lighter shelter, were able to socially influence the cockroaches so that they, on average, also made this 'unnatural' choice.

Parker and Zhang [11] studied a scenario in which a group of robots is expected to choose the best out of a number of unequal options. The robots employ an active recruitment strategy that relies on inter-robot communication. The robots start by looking for options and advocating them to each other, always switching selection to the best-known option. Once a robot's selection becomes sufficiently popular (reaching a *quorum*), the robot becomes committed to it. This enables the group to reach consensus.

Hamann *et al.* [8] studied how a homogeneous group of robots can collectively choose between two global maxima in a light-intensity field. Each robot moves in a straight line until it encounters another robot. Then, it stops and counts the total number of robots in its neighborhood. If this is above some threshold, the robot measures the light intensity and waits for a time proportional to this intensity. This creates a positive feedback effect which enables symmetry breaking between the two options.

Valentini *et al.* [10] studied a swarm of robots that collectively choose among two unequal options. At any moment in

[1]This assumption is relaxed in Section IV-G, where options of different sizes are considered.

time, each robot has an opinion about which option is best. The robot either explores the option, or exchanges information with its neighbors. In the latter case, the robot locally broadcasts its opinion for a duration that is proportional to the perceived quality of the preferred option. Moreover, for a fixed time period it monitors incoming messages and then updates its opinion using the majority rule. The robot then switches to exploring the potentially new option, and the process repeats indefinitely.

In all of the above examples, the agents perform arithmetic computations to determine where or when to move (e.g., artificial potential fields [6, see S13], artificial neural networks [9], timeouts [6], [8], [10] or pseudo-random numbers [7]) or to update internal representations (e.g., preferences [7], [10], [11]). Moreover, the agents need to store information during run time (e.g., quality estimates, behavioral states, or counters). Their sensors typically provide rich information, such as a count of the number of nearby agents. These hardware requirements render it difficult for large quantities of these robots (e.g., $\gg 10^3$) to be produced. Moreover, they render it difficult for the platforms to be scaled down in size to the sub-millimeter level, where the available space for hardware and energy storage is at a premium [13].

Inspired by recent studies on computation-free swarming [14], we hypothesize that—possibly at the cost of a speed of solution trade-off—the aforementioned hardware capabilities are not fundamentally needed for the collective choice problem. Gauci *et al.* [14] showed that a swarm of robots of extreme simplicity were able to aggregate (rendezvous) in a homogeneous environment. Each robot had a binary sensor that detected whether another robot was in the direct line of sight. The robot did not compute, nor did it store information during run time. In [15], the same authors showed that the swarm, using ternary line-of-sight sensors, was able to cluster groups of objects. Brown and Johnson applied the same computation-free swarming framework to solve several tasks including multi-robot rendezvous [16]. Brown *et al.* [17] used novelty search to explore what other tasks could be solved using the framework. In [18], the framework was applied to a swarm of simulated shepherding robots, which controlled herds of simulated sheep. Robots of severely constrained hardware were used as well in [19] and [20], where the authors developed probabilistic strategies for multi-robot rendezvous, using simple sensors and without any means of direct communication.

This paper shows for the first time that a group of robots can collectively choose one of multiple options in their environment without arithmetic computation. The robots use only one ternary digit (trit) of information about their environment, and do not need to store run-time information. The control strategy can therefore be considered to be the simplest solution to date for the collective choice problem.

This paper is organized as follows. Section II defines the collective choice problem, and the sensing, locomotion, and control capabilities of the agents. Section III presents the methodology for obtaining the control strategy. Section IV presents the results obtained when testing the control strategy on swarms of simulated robots. Section V describes how the strategy was ported to a physical robot platform, and presents the experimental results obtained with swarms of 20 e-pucks. Section VI concludes the paper.

II. Problem Definition

A. Scenario and Objective

Consider a 2-D, bounded environment with two identical, circular objects, A and B, referred to as *options*. The options are placed equidistant from the center of the environment. The environment does not contain any other cues. The scenario thus corresponds to the symmetric option qualities and costs variant of the best-of-*n* problem [12]. A group of N mobile agents is initially placed within a region in the center.

The collective choice problem requires the group to commit to either of the two options within a fixed time period. An agent is considered to be committed to option $X \in \{A, B\}$, if it is within a certain range of X. Throughout this paper, an agent can commit to at most one option. The group is considered to be committed to option X, if more than $N/2$ agents—the majority—are committed to X. Note that even if the group committed to an option, a minority of agents could still have committed to the other option, resulting in a split. Splitting is in general undesirable, however, for simplistic agents, not always avoidable [21].

B. Sensing, Locomotion, and Control Capabilities

Each agent is equipped with one line-of-sight sensor at its front, which is able to detect the type of object at which it is pointing. The sensor has a limited range. At each time step, k, an agent's sensor provides one of three possible readings:

$$s[k] = \begin{cases} 2 & \text{if an option is detected;} \\ 1 & \text{if another agent is detected;} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The agent moves using a differential-drive wheel configuration [22]. In other words, it can move forwards or backwards in arcs of arbitrary radius—including straight motion and on-the-spot rotation. The agent therefore has two degrees of freedom, corresponding to the rotational velocities of the left and right wheels, which we denote by v_ℓ and v_r, respectively. Each wheel velocity can be normalized to the interval $[-1, 1]$, where -1 and 1 represent a wheel rotating with maximum speed backwards or forwards, respectively.

We require that the controller[2] shall neither perform any arithmetic computations, nor store any run-time information. From this constraint, it follows that the controller directly maps the sensor reading onto two parameters in $[-1, 1]$—one for each wheel velocity. This takes place at each time step k. Formally,

$$(v_\ell[k], v_r[k]) = \begin{cases} (v_{\ell,0}, v_{r,0}) & \text{if } s[k] = 0; \\ (v_{\ell,1}, v_{r,1}) & \text{if } s[k] = 1; \\ (v_{\ell,2}, v_{r,2}) & \text{if } s[k] = 2, \end{cases} \quad (2)$$

where $v_{\ell,i} \in [-1, 1]$ represents the left wheel velocity corresponding to sensor reading $i \in \{0, 1, 2\}$, and similarly for $v_{r,i}$. Note that the controller is fully specified by the six parameters.

[2]In the following, we refer to the control strategy simply as the *controller*.

III. Controller Synthesis

The problem now reduces to finding the six controller parameters that produce the desired behavior, which is an optimization problem over the real subspace $[-1,1]^6 \subset \mathbb{R}^6$. We approach this problem via the black-box paradigm, where each possible solution can be assigned a score via some evaluation method. An optimization algorithm is employed to find good solutions by iteratively generating candidate solutions and using their quality as feedback. Our evaluation method for a candidate solution (i.e., a controller) consists of running a computer simulation with all agents employing that controller. A suitable metric is used to determine the quality of the candidate solution based on the global behavior that it produces on the agents. Note that while this process is computationally intensive, it is only run once, and it is run off board of the robots. The obtained controller that goes on board the robots is free of arithmetic computation.

In the following, we describe the simulation platform, evaluation of candidate solutions, optimization algorithm, and obtained solution.

A. Simulation Platform

Each agent is a simulated e-puck robot [23], which is a miniature mobile robot with a differential-drive wheel configuration. The e-puck has a circular body with a radius of 3.7 cm and a mass of 150 g. The inter-wheel distance is 5.1 cm, and the maximum wheel velocity is 6.24 rad/s, corresponding to a maximum linear velocity of the robot of 12.8 cm/s. The line-of-sight sensor is implemented by casting a ray and checking whether it intersects with any other object. The ray has a length of 200 cm, limiting the range of the sensor.

The simulator is implemented using the built-in e-puck model of the Enki physics library [24]. Enki simulates the dynamics and interactions of rigid bodies in 2-D. The simulation physics and the control cycle are updated at rates of 100 times per second and 10 times per second, respectively.

B. Evaluation of Candidate Solutions

The evaluation uses a bounded square environment with sides 300 cm, containing a group of $N = 20$ robots.[3] The simulation is run for $T = 5000$ time steps (500 s). We define a quality measure $Q[k]$, which characterizes the distribution of robots at time step k during the trial[4]:

$$Q[k] = \frac{1}{P}\min\left\{\sum_{i=1}^{N}||x_i[k]-x_A||^2, \sum_{i=1}^{N}||x_i[k]-x_B||^2\right\}, \tag{3}$$

where $P = (2R)^2 N$ is a scaling factor, R is the radius of the robot's body, $x_i[k]$ is the position of robot i at time step k, and x_A and x_B are the positions of options A and B, respectively. $Q[k]$ is minimized if the robots collectively opt for either option. The fitness function, to be minimized by the optimizer, is:

$$F = \sum_{k=1}^{T} kQ[k]. \tag{4}$$

[3]Details about their initial placement are described in Section IV-A.

[4]We opted for a continuous function to aid the optimization process.

TABLE I
The Parameters of the Best Controller [see (2)] and the Resulting Motion Primitive

Nothing	Robot (agent)	Option
$v_{\ell,0}$	$v_{\ell,1}$	$v_{\ell,2}$
0.989377	0.999426	−0.106746
$v_{r,0}$	$v_{r,1}$	$v_{r,2}$
−0.348408	0.992379	0.965466
Turn right	Move forward (slight right turn)	Turn left

By taking the time step into account, the fitness function rewards solutions for reaching consensus—the earlier, the better.

C. Optimization Algorithm

As an optimizer, we use the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [25]. CMA-ES is a derivative-free, stochastic, black-box optimization method. We use a population of $\lambda = 20$ candidate solutions, of which $\mu = 10$ are selected for reproduction. The algorithm is executed for 100 generations. Each candidate solution is evaluated 20 times per generation and the average fitness is used.

D. Obtained Controller Solution

We performed 40 evolutionary runs using the aforementioned settings. For each run, we examined the controller of the final generation that exhibited on average the best performance according to (4). We observed that six of these 40 controllers achieved a good performance level. We then conducted preliminary experimental trials using these six controllers, and opted for a controller that retained a good performance level in the physical setup.[5] We refer to this controller as the *best* controller (see Table I).

IV. Simulation Studies

In this section, the best controller is evaluated using simulation experiments.

A. Experimental Setup

The experimental setup is shown in Fig. 1. It defines a region in the center for the robots to start from. At the beginning of a trial, $N = 20$ robots are placed at random positions and with random orientations within this region. The setup also defines two *commitment regions*, one for option A, the other for option B. If a robot resides within a commitment region, it is considered committed to the corresponding option. In the following, we evaluate the robots' commitments after 300 s (i.e., $T = 3000$ time steps).[6]

[5]Note that overdesign—also known as overfitting—is a common issue in evolutionary robotics [26], and may explain why some controllers perform differently in reality than in simulation.

[6]During the optimization process, a larger trial duration of 500 s was used to support the incremental development of promising solutions. Post-analysis of the best controller however revealed that 300 s is sufficient for the swarm to reach consensus, and hence this trial duration is used throughout all simulation and physical experiments.

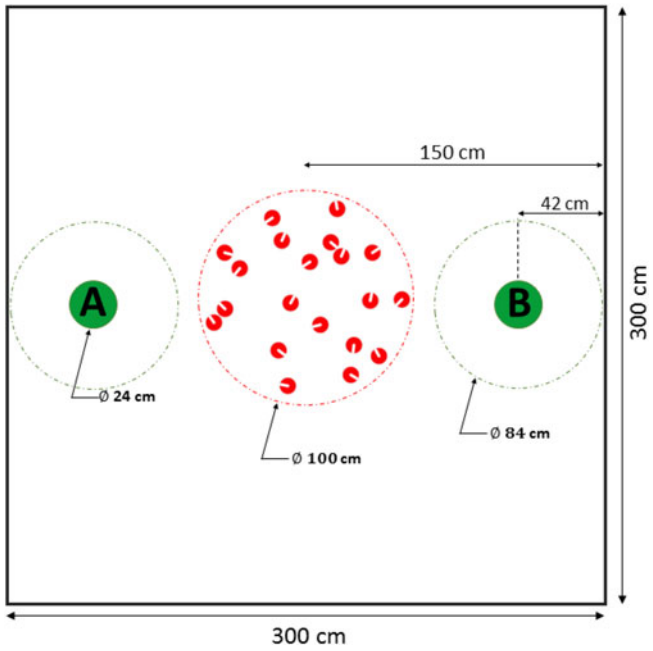

Fig. 1. Illustration of the environment used in the simulation experiments. The black lines indicate the boundary of the environment. The robots start from random positions within the central region. The green solid disks represent options A and B. The circular regions around them indicate the corresponding commitment regions.

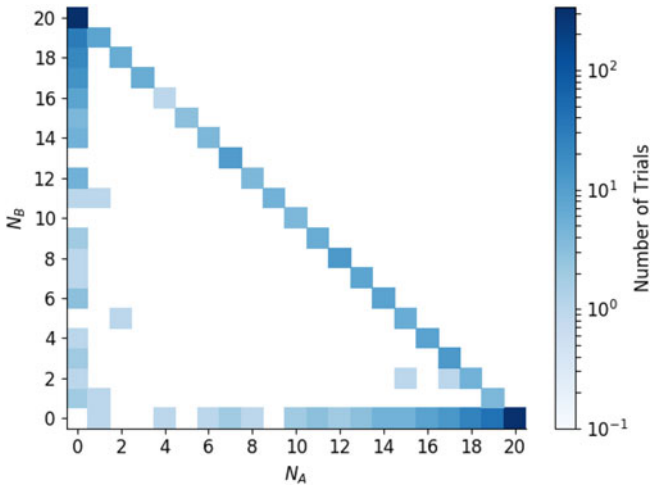

Fig. 2. Number of trials in which N_A and N_B robots committed, respectively, to options A and B. In total, 1000 simulation trials with $N = 20$ robots were conducted, each for a 300 s duration.

B. Analysis of the Behaviors

We conducted 1000 trials using the setup described in Section IV-A.

Fig. 2 shows the number of trials in which $N_A \in \{0, 1, 2, \ldots, N\}$ and $N_B \in \{0, 1, 2, \ldots, N\}$ robots committed, respectively, to options A and B. In 97.3% of the trials, the swarm committed to one of the options, either A (i.e., $N_A > N/2$) or B (i.e., $N_B > N/2$). In 12.8% of the trials, the swarm split across both options (i.e., $N_A > 0$ and $N_B > 0$). In Fig. 2, the inner region of the triangle is virtually empty.[7]

[7]To make outliers visible, a log scale had to be used for the color bar.

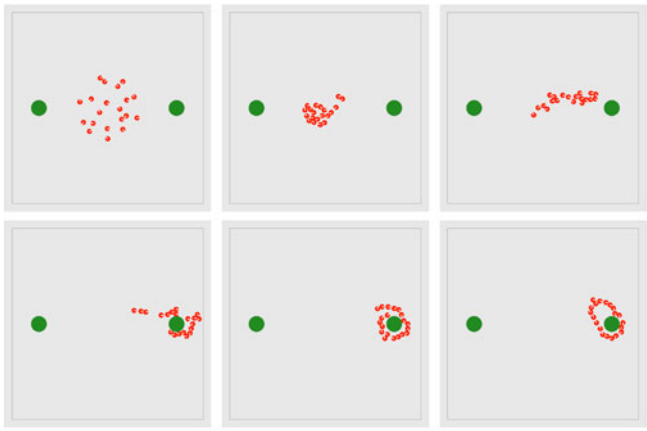

Fig. 3. Sequence of snapshots showing a swarm of 20 simulated robots choosing option B. They were taken (from the top left to the bottom right) once 0, 10, 20, 30, 40, and 300 s had elapsed. Initially the swarm aggregates around the center of the arena. The motion of the robots causes symmetry-breaking and as a result the swarm collectively approaches the option on the right. When $t = 40$ s the robots orbit around option B and remain committed to their choice.

In other words, in almost all cases where the swarm split, there were no uncommitted robots left in the environment (i.e., $N_A + N_B = N$).

We now analyze the best controller (see Table I) in more detail. Consider a robot at time step k. If the robot detects nothing ($s[k] = 0$), it turns to the right $[(v_{\ell,0}, v_{r,0}) = (0.989377, -0.348408)]$. If it detects another robot ($s[k] = 1$), it moves forward while slightly turning to the right $[(v_{\ell,1}, v_{r,1}) = (0.999426, 0.992379)]$. If it detects an option, it turns to the left $[(v_{\ell,2}, v_{r,2}) = (-0.106746, 0.965466)]$. Once the robot loses sight of an option, and detects nothing, it turns to the right, hence likely detecting the option again. The process of alternatively detecting an option and nothing causes the robot to approach the (left edge of the) option. If the option is occluded by other robots, however, these are detected instead, resulting in the robot moving directly towards them (and the option). This seems to facilitate reaching a consensus in the swarm. As more robots join, an orbiting behavior is observed, where each robot follows the robot in front of it. If there is no such robot, a robot slides along the perimeter of the option in a clockwise fashion, while alternately detecting the option and nothing, and keeps doing so until detecting a robot.

Fig. 3 shows a sequence of snapshots taken from a typical trial. A video recording of an example trial is available in the accompanying video.

C. The Effects of Sensor Noise

We investigate how robust the controller is with respect to sensor noise. False negative noise was introduced in the robot's sensor as follows. If the robot had either another robot or an option in front of it, with probability p it would not detect it; in other words, it would obtain the incorrect sensor reading $s[k] = 0$. The probability of misdetection was varied from 0 to 1 in increments of 0.1.

Fig. 4 shows the maximum number of robots having committed to the same option [i.e., $\max(N_A, N_B)$]. The green dashed line represents the performance of the swarm when the noise

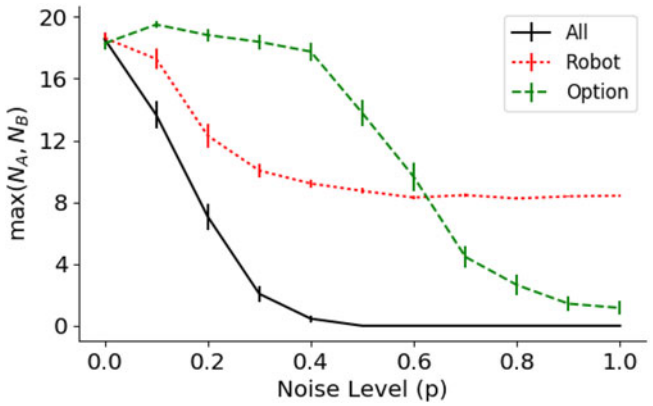

Fig. 4. Effects of sensor noise, p, on the swarm performance, $\max(N_A, N_B)$. For each setting, 100 simulation trials with 20 robots were conducted and averaged. The duration of trials was 300 s. The error bars represent the $\pm$ standard error.

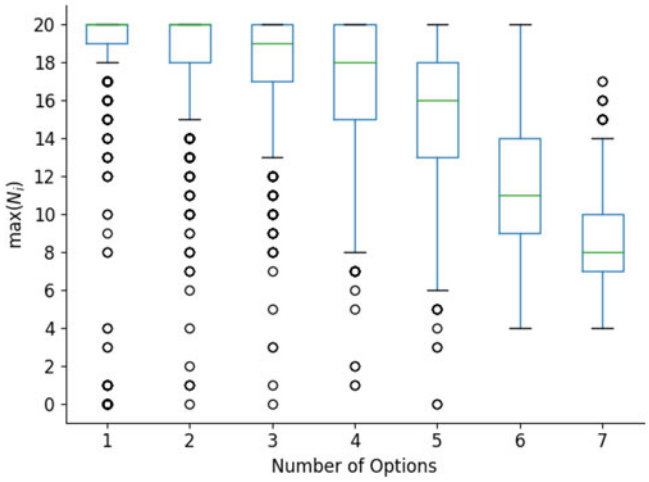

Fig. 5. Performance of a swarm choosing between n options, that is, the maximum number of robots committed to a same option (1000 trials).

is restricted to the detection of options. The swarm copes well with this type of noise; its performance is affected only for noise levels of $\geq 50\%$. The red dotted line represents the performance of the swarm when the noise is restricted to the detection of other robots. The performance of the swarm for noise levels of more than 50% remains at around 9 committed robots [i.e., $\max(N_A, N_B) \approx N/2$]. Once the robots can no longer detect each other, as expected, the swarm splits in two, about equally sized, sub-groups. The black solid line represents the performance of the swarm when the noise is affecting both the detection of other robots and the options. In this case, the performance drops more rapidly than in the other cases, suggesting that there is a compounding effect from the different types of noise. This limitation of the controller may make it unsuitable for applications in unstructured real-world environments. However, if a noise model is known for a particular environment, this could be incorporated into the controller optimization process, and this may yield a better controller.

D. Choosing Between More Than Two Options

We explore the scenario with $n > 2$ options. Apart from the number and positions of options, the environment remains as shown in Fig. 1. One option is placed as option B in Fig. 1, whereas the remaining $n-1$ options are equally spaced along the circle with the same center as the environment. With this configuration, $n = 7$ is the maximum such that the commitment regions do not overlap. We performed 1000 trials for each $n = 1, 2, \ldots, 7$.

The results are shown in Fig. 5. The performance degrades gracefully as the number of options increases, even though the controller was optimized for $n = 2$ options. For $n = 7$ options, the swarm did not commit in the majority of the trials. We observed that the swarm could orbit around multiple options. As neighboring options are in close proximity, robot were likely to be attracted by them.

E. The Effects of the Robot Starting Positions

We investigate how the initial starting positions affect the performance of the swarm. We performed 1000 trials for each investigated scenario.

First, we initialized the robots randomly in a circular region four times larger than the one used before. The change in performance was not significant (it dropped from 97.3% to 96.6%). We then initialized the robots randomly anywhere in the environment. The majority of the swarm committed in 83.4% of the trials, but when we also changed the sensing range to unlimited (i.e., long enough to detect any point in the arena), the swarm committed in 94.6% of the trials. These results show that, as long as the robots have a long enough sensing range, their initial configuration has only a low impact on performance.

To explore the capabilities of the robots utilizing a shorter sensing range in a sparse initial distribution, we reran the controller optimization process for three setups. The sensor range was limited to 200 cm (as before), 100 cm, and 50 cm, and in each case, the robots were initialized randomly anywhere in the environment. For the best controllers, 1000 simulation trials were performed. The swarm committed in 98.6%, 90.3% and 48.4% of the trials to an option when the robots were equipped with a sensor range of 200 cm, 100 cm and 50 cm, respectively. These results indicate that our computation-free swarming framework tolerates some limitations in the sensor range, but is unable to cope with strictly local sensing. This is in line with [14], which shows for the framework—albeit for a different task—that there exists no memoryless solution to maintain connectivity, unless the sensor range is sufficiently large.

F. The Effects of the Swarm Size

We investigate the scalability of the controller by measuring the performance for swarms of $10, 20, 30, \ldots, 100$ robots. Note that the more robots in the swarm, the harder it would be for all of them to fit inside the commitment region, as defined in Fig. 1. To alleviate this problem, and thereby allowing a fair comparison between different group sizes, we removed the boundary of the environment and redefined the commitment regions to be the left and right half-planes, splitting the environment in its center in half. In other words, each robot is committed at all times to its nearest option. We do not determine if the swarm (majority of robots) commit to the same option, but rather examine the

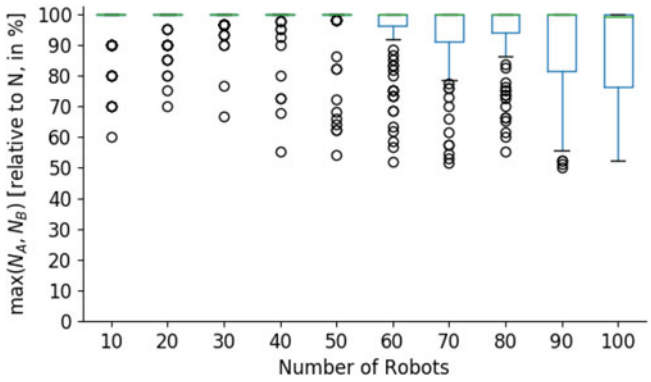

Fig. 6. Effects of the swarm size, N, on the swarm performance, $\max(N_A, N_B)$ (in percentage). For each setting, 100 simulation trials were conducted. The duration of trials was 300 s. The data is represented using box plots. Here a robot is considered committed to the option that is nearest.

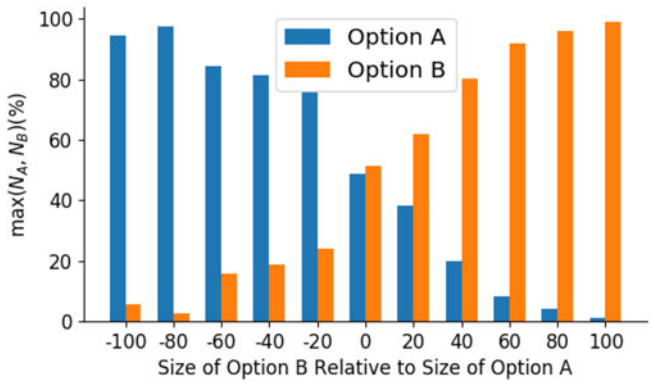

Fig. 7. Ability of the controller to let a swarm of robots choose between unequal alternatives. The bars show the average percentage of robots committed to options A and B, respectively (100 simulation trials of 300 s duration). For details, see text.

percentage of robots committing to the options. At the beginning of each trial, about 50% of the robots are committed to either of the two options.

Fig. 6 presents the results of 100 trials per setting. The performance scales reasonably well with the numbers of robots, despite the options being placed at a constant distance from each other. The average commitment to a same option is 97.9% for 20 robots and 88.5% for 100 robots. When near an option, the swarm orbits around it. The more robots, the bigger the radius of the orbit becomes. As a result, the performance of large swarms drops with respect to the half-plane measure.

G. Choosing Between Unequal Alternatives

In this section, we investigate the ability of the swarm to choose between two *unequal* alternatives. This scenario corresponds to the asymmetric option qualities and symmetric option costs variant of the best-of-*n* problem [12]. Option A was kept identical, as shown in Fig. 1. However, option B was changed in radius from -100% (implying it is effectively removed) to $+100\%$, by 20% increments. Our hypothesis was that the larger option, if any, will be preferred. As in Section IV-F, we removed the environment boundary and redefined the commitment regions using half-planes. This was done to prevent the situation that it is harder for the swarm to squeeze into a relatively small commitment region, as the physical dimension of the option increases.

Fig. 7 shows the percentage of robots committed to options A and B at the end of 100 trials (per setting). As expected, in trials with equally sized options, the robots have no preference. As option B becomes smaller or larger however, the robots increasingly succeeds in detecting such differences. When option B has twice the radius of option A, they almost exclusively opt for it.

The finding suggests that while the controller was designed and optimized for a particular problem—choosing among equal alternatives—it can also be used to choose the largest of unequal alternatives. The controller would be unable to consistently choose the smallest of equidistant, unequal alternatives. Moreover, it might favor smaller but closer options over larger but more distant ones.

Fig. 8. Close view of the robotic platform used in the experiments. The e-puck is coated in *red* to be identifiable by other e-pucks. The marker on its top is used by the tracking system for the post-analysis.

V. Experiments

In this section, the best controller is evaluated using experiments with physical robots.

A. Porting of the Controller

To validate the controller on a physical platform, we use the e-puck robot [23], shown in Fig. 8. The line-of-sight sensor was emulated using the on-board camera, which is a 640×480 active-pixel sensor. To determine the sensor value, a centered $a \times b$ pixel region is used. We chose $a = 2$ columns to ensure that the emulated sensor points exactly towards the front, and $b = 15$ rows to improve the sensing range—misaligned cameras (pitch axis) would otherwise cause false negatives. The sensor detects the color of the object it is pointed at. The sensor reading $s[k]$ is 0 if no object (i.e., effectively the white boundary) is detected, $s[k] = 1$ if a red object (robot) is detected, and $s[k] = 2$ if a green object (options A or B) is detected. The aforementioned detection procedure uses arithmetic computation. The controller, however, remains computation-free.

B. Experimental Setup

The robots operate in a 300 cm $\times$ 300 cm environment, which is bounded by a white wall of height 50 cm. The options are represented as green cylinders with a diameter of 24 cm and a height of 10 cm. They are placed as indicated in Fig. 1.

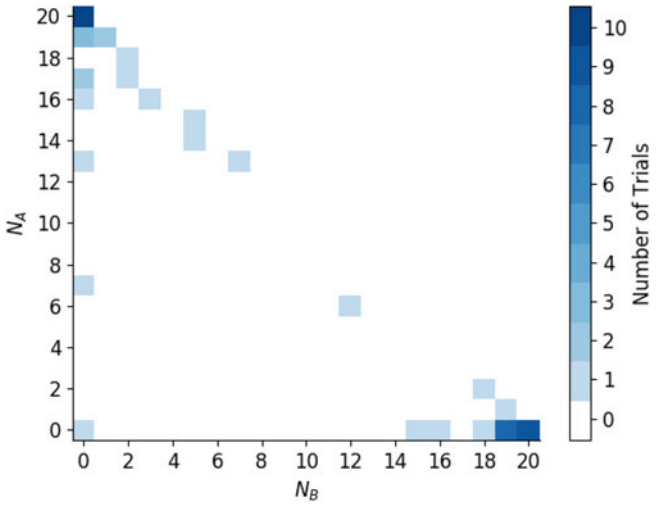

Fig. 9. Number of experimental trials in which N_A and N_B robots committed to options A and B, respectively. In total, 50 trials with 20 physical e-pucks were conducted, each for a 300 s duration.

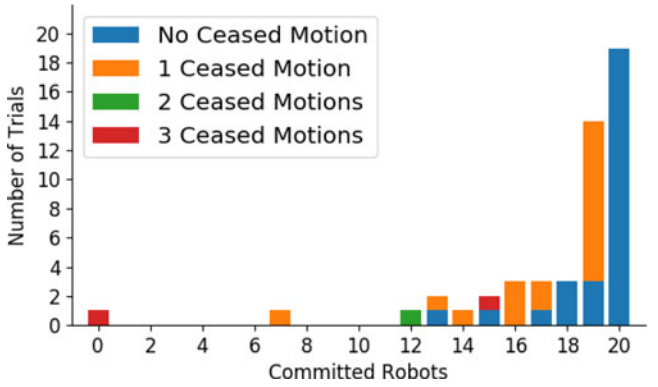

Fig. 10. Breakdown of the 50 experimental trials according to the maximum number of robots that committed to the same option, $\max(N_A, N_B)$. Each trial is shown with a color indicating how many of the 20 physical robots ceased motion.

We distributed the robots in a hexagonal grid pattern as shown in Fig. 12(a). Random permutations were used to determine the order of placing the robots on the 20 grid locations. The orientation of each robot was uniformly chosen from $[0, 2\pi)$.

The trial was started by broadcasting an infrared signal to all robots using a remote control. No human intervention took place; where robots ceased motion during a trial, they were left in the environment. The trial duration was 300 s.

All trials were recorded by an overhead camera at a rate of 25 fps. The recordings were analyzed using the OpenCV [27] computer vision library. Distortion effects in the images were removed and the positions of robots tracked automatically.

C. Results

A set of 50 experimental trials were conducted using $N = 20$ e-puck robots. Video recordings of all trials are available in the online supplementary material.[8]

[8]Online supplementary material, http://naturalrobotics.group.shef.ac.uk/supp/2018-002

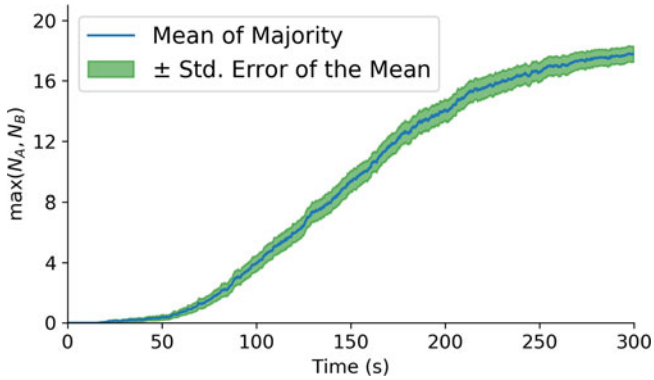

Fig. 11. Dynamics of $\max(N_A, N_B)$, averaged over the 50 experimental trials with 20 physical robots.

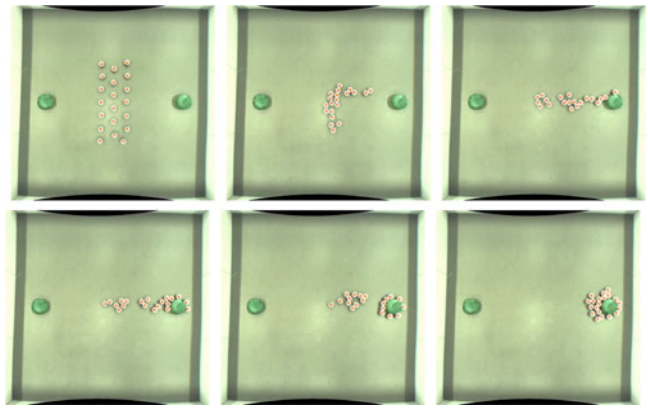

Fig. 12. A sequence of snapshots from a typical experimental trial with 20 physical robots. They were taken (from the top left to the bottom right) once 0, 20, 40, 80, 120, and 300 s had elapsed. Due to distortion removal, blank pixels occur at the top and bottom of the images.

Fig. 9 shows the number of robots committed to either option A or B (N_A and N_B, respectively). In 96% of the trials, the swarm committed to one of the options, A or B; in other words, the majority of the robots ended up choosing that option. In 25 trials, the swarm committed to A, whereas in 23 trials, it committed to B.

Over the course of the experiments, the robots were set to operate for 300 s, a total of 1000 times (50 trials with 20 robots). In 2.7% of these cases, the robot ceased motion at some point during the trial. This may happen for a variety of reasons, including a lost contact with the battery or a low battery state. Fig. 10 shows the number of trials for each combination of $\max(N_A, N_B)$. The color of each trial indicates how many robots ceased motion; the latter was manually determined, through visual inspection of the video recordings. The more robots with ceased motion, the more the performance was affected.

Fig. 11 shows the maximum number of robots committed to the same option over time, $\max(N_A, N_B)$. The blue line indicates the mean and the green envelope the $\pm$ standard error across the 50 trials.

Fig. 12 shows the behaviour of the robots during a typical trial. In this trial, it takes approximately 35 s for the first robot to approach the option. The rest of the robots tend to follow, and the

whole swarm is committed to the option after 150 s. The swarm then remains in the commitment area until the end of the trial.

VI. Conclusions

In this letter, we showed that a group of embodied agents can collectively choose, without arithmetic computation, between multiple alternatives in an environment. The agents we considered used a single line-of-sight sensor, obtaining a ternary digit of information about the environment. The agents could not communicate, nor store any information during run time. They directly mapped the sensor reading onto constant-value motor commands. Compared to previous solutions to the collective choice problem, the proposed control strategy requires significantly lower information processing capabilities—at the expense of a longer sensing range—and could be implemented on platforms that lack an arithmetic logic unit.

Using computer simulations, we demonstrated that the control strategy was fairly robust with respect to sensory noise as well as changes in the number of robots or options. We also showed that the strategy works well for a range of different initial configurations, provided that the sensor's range is sufficiently long. We examined the problems of choosing between equal alternatives and between unequal alternatives. In the latter case, an option's quality was reflected by its size (the bigger, the better). To choose between options of the same size but unequal qualities, the robots would need to be equipped with sensors to detect such differences. Assuming that only a limited number of quality levels are possible, our framework could be adapted accordingly.

We ported the control strategy onto the e-puck platform, and performed 50 experimental trials with 20 physical robots. The swarm succeeded in choosing an option in 96% of the trials, despite some robots ceasing motion during the trials.

The extreme simplicity of our control strategy makes it potentially applicable to robotic systems operating at the submillimeter-scale. For example, nanorobots could be configured to collectively target one of multiple regions of interest at a time. The controller used in our proof-of-concept implementation was free of arithmetic computations, but the sensor was not. In future work, we wish to address this by studying physical swarm robotics platforms with reduced hardware complexity [13], [28], [29], which could benefit from our low-capability control strategy.

Acknowledgment

The authors thank James A. R. Marshall for fruitful discussions and comments on an earlier version of this manuscript.

References

[1] J. N. Tsitsiklis, "Problems in decentralized decision making and computation," Ph.D. dissertation, MIT, Cambridge, MA, USA, Dec. 1984.

[2] L. Conradt and T. J. Roper, "Consensus decision making in animals," *Trends Ecol. Evol.*, vol. 20, no. 8, pp. 449–456, 2005.

[3] T. Bose, A. Reina, and J. A. Marshall, "Collective decision-making," *Current Opinion Behav. Sci.*, vol. 16, pp. 30–34, 2017.

[4] R. Beckers, J. L. Deneubourg, S. Goss, and J. M. Pasteels, "Collective decision making through food recruitment," *Insectes Sociaux*, vol. 37, no. 3, pp. 258–267, 1990.

[5] P. K. Visscher, "Group decision making in nest-site selection among social insects," *Annu. Rev. Entomol.*, vol. 52, no. 1, pp. 255–275, 2007.

[6] J. Halloy *et al.*, "Social integration of robots into groups of cockroaches to control self-organized choices," *Science*, vol. 318, no. 5853, pp. 1155–1158, 2007.

[7] M. A. Hsieh, Á. Halász, S. Berman, and V. Kumar, "Biologically inspired redistribution of a swarm of robots among multiple sites," *Swarm Intell.*, vol. 2, no. 2, pp. 121–141, 2008.

[8] H. Hamann, B. Meyer, T. Schmickl, and K. Crailsheim, "A model of symmetry breaking in collective decision-making," in *From Animals to Animats 11*, vol. 6226. Berlin, Germany: Springer-Verlag, 2010, pp. 639–648.

[9] G. Francesca, M. Brambilla, V. Trianni, M. Dorigo, and M. Birattari, "Analysing an evolved robotic behaviour using a biological model of collegial decision making," in *From Animals to Animats 12*, vol. 7426. Berlin, Germany: Springer-Verlag, 2012, pp. 381–390.

[10] G. Valentini, E. Ferrante, H. Hamann, and M. Dorigo, "Collective decision with 100 Kilobots: Speed versus accuracy in binary discrimination problems," *Auton. Agents Multiagent Syst.*, vol. 30, no. 3, pp. 553–580, 2016.

[11] C. A. Parker and H. Zhang, "Cooperative decision-making in decentralized multiple-robot systems: The best-of-n problem," *IEEE/ASME Trans. Mechatronics*, vol. 14, no. 2, pp. 240–251, Apr. 2009.

[12] G. Valentini, E. Ferrante, and M. Dorigo, "The best-of-n problem in robot swarms: Formalization, state of the art, and novel perspectives," *Frontiers Robot. AI*, vol. 4, no. 9, pp. 1–9, 2017.

[13] A. A. G. Requicha, "Nanorobots, NEMS, and nanoassembly," *Proc. IEEE*, no. 11, pp. 1922–1933, Nov. 2003.

[14] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß, "Self-organized aggregation without computation," *Int. J. Robot. Res.*, vol. 33, no. 8, pp. 1145–1161, 2014.

[15] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Groß, "Clustering objects with robots that do not compute," in *Proc. Int. Conf. Auton. Agents Multiagent Systems*, 2014, pp. 421–428.

[16] M. Johnson and D. S. Brown, "Evolving and controlling perimeter, rendezvous, and foraging behaviors in a computation-free robot swarm," in *Proc. 9th EAI Int. Conf. Bio-Inspired Inf. Commun. Technol.*, 2016, pp. 311–314.

[17] D. S. Brown, R. Turner, O. Hennigh, and S. Loscalzo, "Discovery and exploration of novel swarm behaviors given limited robot capabilities," in *Proc. 13th Int. Symp. Distrib. Auton. Robotic Syst.*, Springer-Verlag, Berlin, Germany, 2018, vol. 6.

[18] A. Ozdemir, M. Gauci, and R. Groß, "Shepherding with robots that do not compute," in *Proc. 14th Eur. Conf. Artif. Life.*, MIT Press, 2017, pp. 332–339.

[19] R. Manor and A. M. Bruckstein, "Chase your farthest neighbour: A simple gathering algorithm for anonymous, oblivious and non-communicating agents," in *Proc. 13th Int. Symp. Distrib. Auton. Robotic Syst.*, Springer-Verlag, Berlin, Germany, vol. 6, 2018.

[20] A. Barel, R. Manor, and A. M. Bruckstein, "Probabilistic gathering of agents with simple sensors," Technion Israel Institute of Technology, Haifa, Israel, Tech. Rep. CIS-2017-03, 2017.

[21] N. R. Franks *et al.*, "Moving targets: Collective decisions and flexible choices in house-hunting ants," *Swarm Intell.*, vol. 1, no. 2, pp. 81–94, 2007.

[22] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*. Cambridge, U.K.: Cambridge Univ. Press, 2010.

[23] F. Mondada *et al.*, "The e-puck, a robot designed for education in engineering," in *Proc. 9th Conf. Auton. Robot Syst. Competitions*, 2009, vol. 1, pp. 59–65.

[24] S. Magnenat, M. Waibel, and A. Beyeler, "Enki: An open source fast 2D robot simulator," 2009. [Online]. Available: https://github.com/enki-community/enki

[25] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evol. Comput.*, vol. 9, no. 2, pp. 159–195, 2001.

[26] M. Birattari, B. Delhaisse, G. Francesca, and Y. Kerdoncuff, "Observing the effects of overdesign in the automatic design of control software for robot swarms," in *Proc. Int. Conf. Swarm Intell.*, Springer, 2016, pp. 149–160.

[27] G. Bradski, "Open source computer vision library," 2000. [Online]. Available: http://opencv.org/

[28] J. Li, B. Esteban-Fernández de Ávila, W. Gao, L. Zhang, and J. Wang, "Micro/nanorobots for biomedicine: Delivery, surgery, sensing, and detoxification," *Sci. Robot.*, vol. 2, no. 4, 2017, Art. no. eaam6431.

[29] M. Sitti, *Mobile Microrobotics*. Cambridge, MA, USA: MIT Press, 2017.