Authors' accepted manuscript
Paper presented at *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 1-5 Oct. 2018, Madrid, Spain.
https://doi.org/10.1109/IROS.2018.8593439

Circle Formation with Computation-Free Robots shows Emergent Behavioural Structure

David St-Onge, Carlo Pinciroli and Giovanni Beltrame

Abstract— **In this paper, we demonstrate how behavioural structure, such as a finite state machine, can emerge in minimal robots without computation nor memory capabilities. As a case study we observe the ability of a group of non-holonomic robots to form robust, self-healing circle formations in a decentralized manner using only a limited frontal binary sensor. We present a grid-search method to find suitable parameters that promote the formation of a stable circle. We then examine how the parameters of the controllers affect the appearance of the behaviour, and provide theoretical proof for its emergence and self-healing properties. We validate the proposed model through a set of experiments with ten mobile real robots. Our results with real robots match the simulated experiments and provide insights on how a simple, computation-free behaviour can generate complex spatio-temporal dynamics.**

I. Introduction

A remarkable characteristic of swarm robotics [1] is the complexity of the behaviours that can be achieved using low-cost, light-weight, low-processing, and weak sensing platforms [2]. Understanding how these complex behaviours emerge can give us new engineering capabilities and potentially transformative scientific insights.

In this paper, we study how a stable circular formation can be obtained in a decentralized fashion with minimal robots. This type of formations have a wide variety of applications [3]. For instance, it could be used as a mobile sensor array to simultaneously explore and map the environment [4], or to extend the coverage of a network around a central tower, scanning simultaneously the surrounding of a building, or deploying a dome structure when dropped in a remote area.

Building upon the work of Gauci et al. [5], Brown et al. [2] determined the family of behaviours emerging from a minimum set of capabilities: devices with a binary sensor to detect the presence of an obstacle, no memory, and no computation capabilities beyond a simple `if` statement to branch according to the output of the sensor. Brown et al. [2] showed that this simple model can generate a diverse set of behaviours, such as aggregation, dispersion, and circle formation. While aggregation was formally analyzed by Gauci et al. [5], the mechanics of circle formation have never been formalized.

The goal of this paper is to provide a formal analysis of the circle formation algorithm that can be obtained from Brown et al.'s model [2]. In particular, we study how the formation emerges and maintains its long-term stability, and highlight the self-healing properties of this remarkable behaviour.

The deeper insight of this work is that the circle behaviour formation undergoes a number of phases that can be formalized as a finite state machine. This insight is remarkable because the robots have no memory, often refer to as *oblivious robots* [6], and thus *no explicit notion of group state*. Therefore, using circle formation as a case study, this paper shows that *structured behaviours can emerge* despite the lack of an explicit internal representation to foster it and without computation. In circle formation, the emergence of structured behaviour stems from the peculiar mix of robot sensing and motion capabilities, and from the spatio-temporal dynamics obtainable with the controller we analyze.

This paper is organized as follows. We present related work in Section II. In Section III, we introduce the computational model of our robot, which include its behaviour and its actuation and sensing capabilities. In Section IV we describe the process we followed to find the parameters that produce the circle formation behaviour. We then analyze these parameters in Section V. We validate the behaviour in a set of experiments with real Khepera IV robots in Section VI. Final remarks are discussed in Section VII.

Dr. St-Onge and Pr. Beltrame are with the Department of Computer and Software Engineering, École Polytechnique de Montréal, 2900 Boul Édouard-Montpetit, Québec CA e-mail: (david.st-onge@polymtl.ca).

Pr. Pinciroli is with the Department of Computer Science, Worcester Polytechnic Institute, 85 Prescott St, US e-mail: (cpinciroli@wpi.edu).

II. Related work

The problem of forming a circle has been tackled with a wide variety of approaches. The work of Gautam et al. [7] is based on the insight that a circle can be seen as a looping chain—a structure in which every robot follows the robot in front. The approach of Gautam et al. uses a leader-follower mechanism to build the chain, which is then closed by the leader robot. Souissi et al. [4] extended the leader-follower schema for the case involving Byzantine faulty robots, and proved convergence [4]. This algorithm is limited by its assumptions, but requires less computational resources than methods based on Voronoi decomposition [8].

Recent work shows that complex collective behaviours can be achieved on computation-free robots, such as aggregation and foraging [9]. Among these behaviours, both Gauci et al. and Brown et al. [5], [2] identified what they called the *cyclic pursuit*. In particular, Brown et al. [2] showed that this behaviour can be obtained with a binary (1-bit) sensor, a given square environment, and a controller based on a single `if` statement. In this paper, we build on this discovery and use a binary controller to generate stable, self-healing circles.

Many approaches to pattern formation in swarms were also inspired by hydrodynamics or electric field models, such as in [10]. Specific to the circle geometry, the work

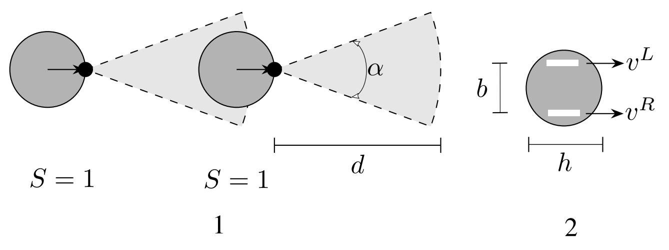

Fig. 1: Wheeled robots equipped with: 1) a frontal 1-bit sensor (S) capable of detecting the presence of another robot within a limited range d. 2) differential drive.

of [11] presents a viscoelastic model and demonstrate its convergence in physics-based simulated experiments. However, the proposed model is more demanding than the presented approach, since it uses virtual forces calculated using range-and-bearing, proximity sensors, and non-trivial processing.

With respect to these approaches, the algorithm presented in this paper is minimal, in that it does not require communication, computation, nor distance-based sensing.

III. Model

A. Robot

This work targets differential drive platforms. The robot is assumed to be capable of moving in a non-holonomic fashion and equipped with a frontal 1-bit sensor, $S = \{0, 1\}$, that detects the presence or absence of a robot with an aperture angle α, as shown in Fig. 1. In contrast with the assumptions in [5], in this work the sensor range is assumed to be $d < \infty$.

The velocity of the steering robot can be expressed relative to its Instantaneous Center of Curvature (ICC) as a vector $\boldsymbol{v}$ which in polar coordinates is $\boldsymbol{v} = \begin{Bmatrix} v_t & \dot{\theta} \end{Bmatrix}$ with:

$$v_t = \frac{v^R + v^L}{2} \qquad \dot{\theta} = \frac{v^R - v^L}{b} \tag{1}$$

where b is the distance between the wheels. The instantaneous position of the ICC relative to the robot centroid is then given by:

$$R = \frac{v_t}{\dot{\theta}} = \frac{b}{2} \frac{v^R + v^L}{v^R - v^L} \tag{2}$$

To derive the Cartesian position of the robot in a global reference frame, one can use:

$$x(t) = \int_0^t v_t \cos\theta dt \quad y(t) = \int_0^t v_t \sin\theta dt \quad \theta(t) = \int_0^t \dot{\theta} dt \tag{3}$$

A more detailed analysis and derivation of the differential drive kinematics can be found in [12].

B. Controller

The controller is recalled here from [2]. It consists of a simple conditional statement based on the binary value of the obstacle sensor in the front of the robot. If the robot does not see anything ahead, it turns with a certain velocity and radius. When the sensor detects an object, the robot changes its velocity, radius, and its direction of rotation. The behaviour can be described five lines of pseudocode:

Algorithm 1 Control algorithm

```
1: if $S$ == 1 then
2:     set wheels speed ($v_1^L$, $v_1^R$)
3: else
4:     set wheels speed ($v_0^L$, $v_0^R$)
5: end if
```

This algorithm is said to be computation-free, i.e. it does not require to compute any value, but rather acts as a direct reaction to a binary input. The control in simulation and on the physical robots proceeds in a step-wise fashion.

IV. Parameter selection

The controller has four parameters values $(v_1^R, v_1^L, v_0^R, v_0^L)$. The geometry and dynamics of the robot also influence the resulting behaviour, such as the robot size (b and h), its sensor range (d) and aperture (α), and its maximum velocity. Through an extensive grid search performed in simulation, we found the working parameters and their mutual influence for a given robot type to achieve a circle formation. We discretized the space by choosing the four controller speeds in {0,1,3,5,7,9} cm/s, the sensor aperture angle in $\{1°, 5°, 15°, 30°, 45°, 60°\}$, and the sensor range in {1,2,3,4,5,6} m, for a total of 46,656 configurations. In Algorithm 1, the sensor range is the threshold used to convert an analog range finder to a binary detector (S). As for the sensor aperture, most robots either use a camera from which only a subset of pixels can be taken, or, as with the Kheperas of the experiments presented in Section VI, an array of infrared sensors from which only a subset can be activated. Each simulation was executed 100 times, involved 20 robots, and lasted 600 simulated seconds, after which the simulation was stopped.

To evaluate the configuration tested in each run, we considered the position $\boldsymbol{x}_i(t)$ of each robot i at time t. We first calculated the center of mass $\boldsymbol{C}(t) = E[\boldsymbol{x}_i(t)]$, where $E[\cdot]$ denotes the expected value over the distribution of positions $\boldsymbol{x}_i(t)$ at time t. The weighted sum ensures a more stable calculation of the center than a simple average. Subsequently, we calculated the standard deviation $D(t)$ of the distance of each robot to the center of mass:

$$D(t) = \text{stddev}[\|(\boldsymbol{x}_i(t) - \boldsymbol{C}(t))\|].$$

Our performance metrics for each experiment consider only the last 120 sec of a run. Over this period, we extracted P_w, the highest (worst) standard deviation ($D(t)$) and we computed P_a, the average standard deviation and P_s, the standard deviation of the standard deviations:

$$P_w = \max_{t>120} D(t) \tag{4}$$

$$P_a = E_{t>120}[D(t)] \tag{5}$$

$$P_s = \text{stddev}_{t>120}[D(t)] \tag{6}$$

We made this choice to give the robots some time to form the loop (the first 480 sec) and then to promote those controllers that can maintain the loop for a sufficiently long time (120

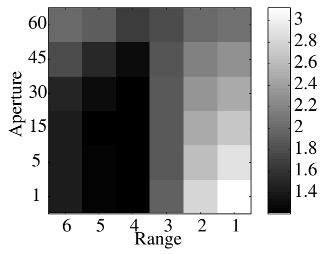

Fig. 2: Range d and aperture α influence on the worst value of the standard deviation in simulation with 20 robots.

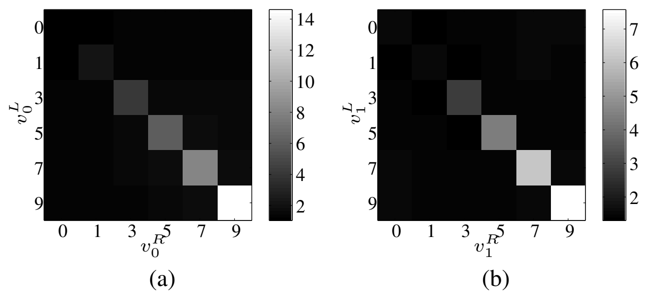
Fig. 3: Velocity influence of the right (v^R) and the left (v^L) wheels on the worst value of the standard deviation: (a) without a robot in sight ($S = 0$), (b) with a robot in sight ($S = 1$).

sec in our case) as an additional optimization step. The most meaningful results are presented in Fig. 2 to 4, showing the influence of key parameters on the worst standard deviation (P_w) of a set of simulation runs. The lower the deviation, the more stable and precise is the final formation.

The range and aperture plot (Fig. 2) shows that the best performance involve large ranges ($\rho > 2$ m) and small apertures ($\alpha < 30°$). This observation was expected since the original work of Gauci et al. [5] used infinite range and point aperture sensors. Small ranges yield the worst results, which is intuitive since the robots are required to be closer in order to detect each other and can easily get out of range. Larger apertures help detect other robots faster, but provoke more oscillation in the formation. These oscillations may render it difficult for the robots to keep a stable distance between each other and even cause the formation to break.

The velocity plots (Fig. 3) show that satisfactory performance can be obtained for many velocity choices, whenever

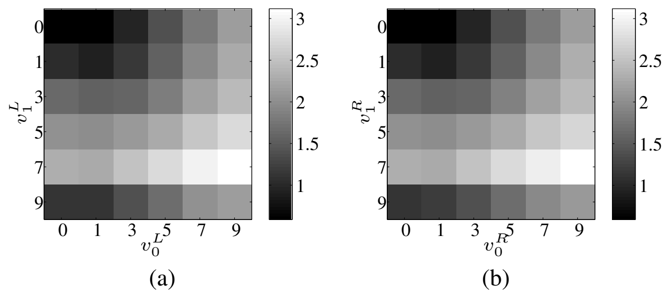
Fig. 4: Velocity influence without ($S = 0$) and with a robot in sight ($S = 1$) on the worst value of the standard deviation: (a) for the left (v^L) wheel, (b) for right (v^R) wheel.

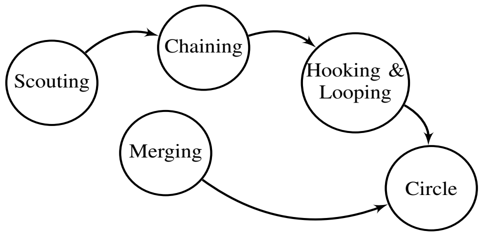

Fig. 5: The states emerging from Algorithm 1.

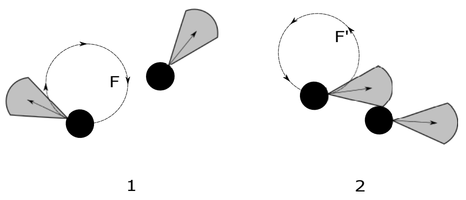

Fig. 6: Scouting sequence: (1) Lone robot turning in the F direction, (2) when another robot is detected the direction switched to F'.

this condition holds:

$$v_i^L \neq v_i^R, i \in [0, 1]$$

Fig. 4 reveals that the pairs (v_0^L, v_1^L) and (v_0^R, v_1^R) cannot both be too small, which would lead to a small instantaneous radius of curvature; rather, the best performance occurs when one speed is large and the other smaller. With regards to the symmetry of Algorithm 1, Fig. 3 and Fig. 4 show that the influence of the velocities is indeed symmetric.

V. Emerging States

Using the best parameter choice we could find using grid search (detailed in Sec. VI), we observed and analyzed the circle formation process. The most remarkable phenomenon is the emergence of what amounts to a *state machine* (see Fig. 5): the robots exhibit different behaviours following their position with respect to the rest of the swarm.

The behaviour of Algorithm 1 converges to a circle passing through four states: 1) **Scouting:** exploring the space to find a leader robot 2) **Chaining:** following a moving robot 3) **Hooking and Looping:** closing the loop 4) **Merging:** integrating loners within a formed circle. In the rest of this section, we formally characterize these four emergent states and their dynamics.

A. Scouting

Lemma 5.1: Every robot, if the space is free of obstacles, explores the surrounding space by turning in a direction F. The robot draws a circle arc whose radius is determined by Eq. (2). The behaviour is switched to "chaining" when the robot detects another robot in range.
As discussed in Sec. IV, the set of viable velocities for circle formation is symmetric. By selecting two values v_0^L and v_0^R such that one velocity is small and one is large, we select a specific turning direction F, which draws an arc of radius R according to Eq. (2). For the rest of this paper, and without loss of generality, we pick F to be the clockwise direction, and F' the anti-clockwise direction.

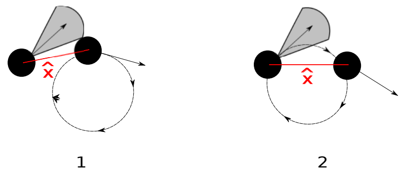

Fig. 7: Chaining sequence: (1) a robot in sight is moving away, (2) the follower turns to catch up.

In Algorithm 1, this behaviour can proceed as long as the robot cannot sense the presence of another robot ($S = 0$). As shown in Fig. 6, when a robot is detected ($S = 1$), two possibilities exist, depending on $\boldsymbol{v}$:

1) The robot turns so that the other robot exits its sensor cone. In this case, it resumes rotating clockwise.
2) The closest robot in range in front (which we call *predecessor*) stays in this robot's sensor cone. In this case, this robot keeps turning counterclockwise until its predecessor is not visible anymore. At that point, the robot resumes it clockwise turning.

In both cases, the net effect is to keep the right side of a robot's sensing cone as close as possible to the left side of the robot in front. The behaviour of this state heavily depends on $\boldsymbol{v}$ and α. From Fig. 2, one can infer that, if the robot moves fast and has a narrow view, it is possible that it would miss the robot in front. Therefore, for this state to work, $\boldsymbol{v}$ and α must be chosen so that a robot can track its predecessor as closely as possible. Formally, a robot will not miss its predecessor if

$$\frac{V}{b}\Delta t < 2\alpha + \frac{h}{d} \ , \tag{7}$$

where h is the robot diameter, b the inter-wheel distance (see Fig. 1) and $V = \boldsymbol{v} \cdot \boldsymbol{v}$, the velocity magnitude. In a swarm perspective, all robots have similar dynamics and obey the same controller, which allow us to state: if Eq. (7) is satisfied, the robot always keeps its predecessor within sight.

B. Chaining

Lemma 5.2: A robot always follows another robot, once it detects it. This will produce an orderly chain of robots.

Let us consider the axis $\hat{\boldsymbol{x}}$ that connects two robots R_1 and R_2 (see Fig. 7). As explained in Sec. V-A, if R_1 can see R_2, i.e. $S = 1$, R_1 turns counterclockwise ($\dot{\theta} > 0$). The axis $\hat{\boldsymbol{x}}$ also rotates by $-V\Delta t/b$. Let us assume that the initial rotation is $\beta = \alpha$; the final rotation is then $\beta' = \alpha + V\Delta t/b$ and R_1 does not see R_2 anymore. The projection of the motion of R_1's right wheel on the $\hat{\boldsymbol{x}}$ axis is $V\Delta t \cos\alpha$. So, if $0° < \alpha < 90°$, this projection is greater than 0, and R_1 moves towards R_2.

In the following control step, R_1 rotates counterclockwise by $-V\Delta t/b$, restoring the rotation of the $\hat{\boldsymbol{x}}$ axis to its initial value and setting $\beta'' = \alpha$. The projection of the motion of R_1's right wheel on the $\hat{\boldsymbol{x}}$ axis is $V\Delta t \cos\beta'$. So, if $0° < \beta' < 90°$, this projection is greater than 0, and the distance between R_1 and R_2 decreases. The fact that $\hat{\boldsymbol{x}}$ rotates causes R_1 to move towards R_2 following an arc.

When R_2 moves, this behaviour causes R_1 to follow R_2 if the speed of R_2 does not exceed the top speed of R_1.

By reasoning inductively, this behaviour can be applied to any chain of N robots in which every robot has a predecessor in range.

C. Hooking and Looping

Lemma 5.3: A chain of robots led by a robot still exploring the space will form a hook that folds the chain on itself until a loop is formed.

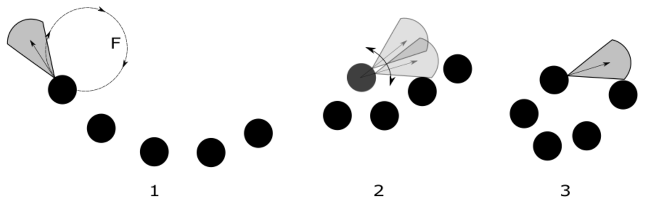

Fig. 8: Hooking and Looping sequence: (1) the front robot is scouting, (2) the front robot stays close to the chain without merging into it, (3) the front robot reaches the tail of the chain and creates a closed loop.

When a chain of robots is formed, the front robot in the chain R_1 sees no other robot. Thus, due to Algorithm 1, R_1 turns clockwise on a circle, as detailed in the scouting state. The chain then traces a hook-shaped path until R_1 sees another robot R_2. At this point, R_1 rotates counterclockwise until R_2 is outside the sensory cone. By alternating clockwise and counterclockwise rotations, R_1 is effectively keeping itself close to the chain of robots, without ever merging into it.

The robot R_3 which is behind R_1 maintains the integrity of the chain. In fact, when R_3 initially loses sight of R_1, it turns clockwise because its sensor reads 0. Turning clockwise causes R_3 to eventually regain sight of R_1. When this happens, the chain is restored. At any moment, every robot in the chain apart from R_1 keeps its sensing cone towards the exterior of the hooking chain. Therefore, a robot in the chain can interact only with its closest neighbour in front.

The net result of these actions is the formation of a hook, as shown in Fig. 8. The robots stay in this state until R_1 reaches the tail of the chain and cannot see any other robot in front. When this happens, R_1 turns clockwise until it sees the last robot in the chain and becomes a follower of the latter robot. A closed loop is thus formed.

D. Merging

Lemma 5.4: Any robot not part of an already formed circle will merge with it.

Scouting, Chaining, Hooking and Looping demonstrate the convergence of a swarm to a circle shape. These three states are successful when performed in sequence, but all the robots have to be part of the chain before it closes. However, it is likely that a circle emerges with only a subset of the robots, while others are still scouting.

A robot R_1 that is not yet integrated can either be inside or outside of the circle. It eventually merges between two robots

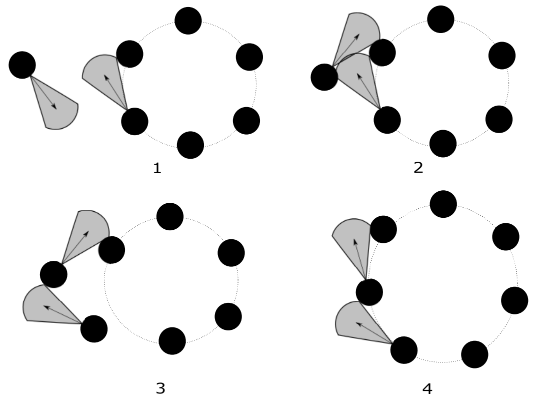

Fig. 9: Merging sequence: (1) Lone robot approaching the circle, (2) Successor detecting the new robot, (3) Lone robot following a predecessor, (4) New stable circle formation.

and becomes part of the circle thanks to the limited aperture of the sensor. R_1 first follows the circle's internal or external perimeter, as in the hooking state. When R_1 gets close to the circle, the robot behind it, R_2 (which we can refer to as the successor), sees R_1 and starts to follow it, creating a breach for R_1 to join. The whole sequence is illustrated in Fig. 9.

To guarantee that any lonely robot merges in the circle when sufficiently close, and to ensure the robots always reach such a close position, we must select the velocity $\boldsymbol{v}_0$ when no neighbours are within range so that the exploration of the unconstrained square space of dimension $L \times L$ is covered. If $R \geq \frac{L}{4}$, with R from Eq. (2) and L is the length of a square region, the behaviour depicted in Fig. 9 occurs.

E. Final Shape

The chaining action averages the movement of both turns (clockwise and counterclockwise) depending on the robots' sensing cone (set with α and d). This average determines the steady state distance between two neighbours in the circle, inside the maximum visibility range. This means that each robot stabilizes at an optimal distance from its neighbours, identical for all, thus forming a circular (rather than ellipsoidal) pattern.

The number of robots N determines the arc angle between two robots for the circle $\gamma = \frac{2\pi}{N}$. Then the circle radius r_c can be derived from the sine law:

$$r_c = \frac{\sin(\frac{\pi-\gamma}{2})l}{\sin(\gamma)}, \tag{8}$$

where l is computed from the sensor range d and aperture angle α:

$$l = d + \frac{b}{2}\cos(\alpha/2). \tag{9}$$

From Eq. (8), increasing the number of robots directly increases the final circle diameter. In the same manner, from Eq. (9), increasing the range of the sensors also increases the final radius. As for the sensor aperture influence, from Eq.(9) the radius of the circle decreases with the increase of α. This is explained by the need for each robot's average velocity vector to be tangent to the circle in steady state.

TABLE I: Optimal values for the controller parameters and resulting metrics from Eq. (4)–(6)

Range	Aperture	v_{L_0}	v_{R_0}	v_{L_1}	v_{R_1}	P_w	P_s	P_a
3.0	5.0	9.0	1.0	3.0	9.0	0.03	0.005	0.015

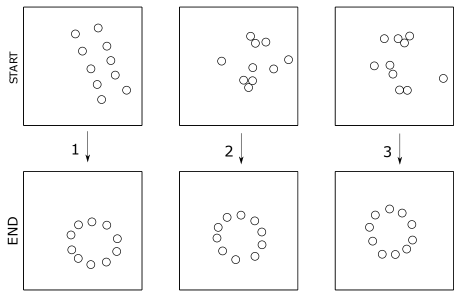

Fig. 10: Three samples of starting and ending positions for the Khepera experiments.

If the counterclockwise rotation radius is greater than the clockwise, the circle may expand over time. While this was observed in simulation for large swarms of robots, it never occurred on an observable scale for swarms of 100 robots or fewer. This is due to the impact of the aperture angle: a large circle (large swarm) requires narrow sensors to be stable.

An additional emergent feature of this behaviour is that it can self-heal. Indeed, removing or adding new robots does not change the group shape, only its radius. If a robot is removed from a stable circle formation, its former successor switches to the Hooking and Looping state until it finds the tail of the chain and closes the loop again. If a new robot is added in the arena, it merges following the mechanism detailed in Section V-D.

VI. Experiments

To test the emergent state machine and validate the performance of the controller for circle formation, we conducted a series of experiments on a set of Khepera IV robots. Kheperas are wheeled robots equipped with an array of 8 circumferential infrared distance sensors. Based on the geometry of the robot, we used the same grid search to obtain the velocities minimizing the standard deviation on the resulting formation. The results are presented in Table I.

To obtain a uniform detection range across robots for our experiments, we calibrated the robots with a distance threshold set for readings at 1 m, which produced more stable readings than the optimal 3m value. If the reading is above this value, a robot is seen ($S = 1$); otherwise $S = 0$.

For each experiment, a student arbitrarily spread the Kheperas over the working space. Fig. 10 shows three initial configuration and the corresponding resulting circle.

Experimental data were collected from a motion capture system based on infrared cameras. The infrared beams did not interfere with the robot sensors since they are directional and oriented towards the ground, while the robot sensors are horizontal, just above the floor.

Both the average and the worst results of ten experiments with the ten Kheperas are shown in Table II. The perfor-

TABLE II: Experiments results on the Khepera over ten runs using metrics from Eq. (4)–(5)

P_w	P_a	Avg. time	Avg. r_c
0.066 m	0.042 m	93 s	0.46 m

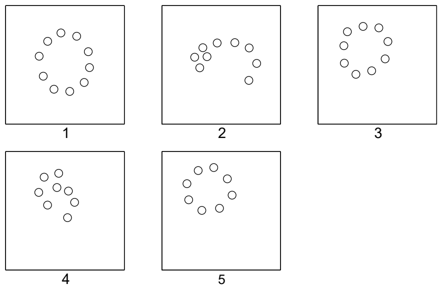

Fig. 11: Positions of the robots while removing agents: (1) first stable circle formed with 10 agents, (2) 50 sec after removing a robot, (3) second stable circle formed with 9 robots, (4) 20 sec after removing a second agent, (5) last stable circle formed with 8 robots.

mances metrics (P_w, P_a) are taken from Eq. (4)–(5). Our results show that all runs achieved a circle formation with minimal deviation of each robot from their expected position around the center ($P_w = 0.066m$). Wheels slipping and sensory noise most likely decrease the performance when compared to the perfect simulation scenario performance in Table I. However, the radius of the resulting circle vary significantly between experiments. The average of 0.46 m is subject to a standard deviation of 0.15 m. From Eq. (8) and the properties of the robots we expected the radius of the shape to be 0.55 m, which is within a tolerable range of the ideal circles obtained in simulation. The variations across experiments can be explained by driving and sensing noise.

We also conducted experiments to test the robustness of the controller on real robots. Theoretically, any robot removed from an already formed circle would create an emerging re-organization of the swarm to form a new circle shape of smaller radius. As with the previous experiments, the ten Kheperas were placed arbitrarily in the workspace before launching our script. Fig. 11.1 depicts the moment when the swarm reached a first circle of 0.694 m radius. Fig. 11.2 shows that a robot was intentionally removed. As a result, the robots formed a new circle of radius 0.615 m (Fig. 11.3). When we removed an additional robot (Fig. 11.4), we obtained another circle of radius 0.598 m (Fig. 11.5). This corroborates the self-healing property of the algorithm and validates the robustness of the controller.

A run from these experiments is available at `https://youtu.be/uVBOcn6uptc`. This video shows extracted footage for each state and a public demonstration of the swarm in which people manipulate the robots to test the robustness of the algorithm.

VII. Conclusions

In this paper we presented a minimal controller for circle formation with a swarm of computation-free robots using only a binary sensor. We analyzed the inherent emerging behaviour of this controller, and highlighted the emergence of a swarm-wide *state machine*.

Beyond circle formation, this paper shows how behavioural structure can arise despite the absence of explicit mechanisms encoded in the script executed by the robots. The findings in this paper also show how capturing the computational aspects of robot swarm can be non-trivial, even in a simple behaviour such as circle formation. This insight is important when analyzing swarm behaviours that display unwanted emergent characteristics, such as error cascades or traffic jams.

Future work will focus on studying a more general class of minimal behaviours, in which every robot is allowed to store 1-bit of internal state information. Expanding the grid-search methodology detailed in this paper, we will attempt to uncover (i) what type of state information can be stored; and (ii) how this internal state affects the nature and complexity of the swarm behaviours that can be obtained.

References

[1] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: A review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, no. 1, pp. 1–41, 2013.

[2] D. S. Brown, R. Turner, O. Hennigh, and S. Loscalzo, "Discovery and Exploration of Novel Swarm Behaviors given Limited Robot Capabilities," in *Proc. of the 13th International Symposium on Distributed Autonomous Robotic Systems*, London, 2016.

[3] K. Swaminathan and A. A. Minai, "A general approach to swarm coordination using circle formation," in *Studies in Computational Intelligence*, 2006, vol. 31, no. 2006, ch. 3, pp. 65–84.

[4] S. Souissi, T. Izumi, and K. Wada, "Byzantine-tolerant circle formation by oblivious mobile robots," *2011 International Conference on Communications, Computing and Control Applications, CCCA 2011*, 2011.

[5] M. Gauci, J. Chen, W. Li, T. J. Dodd, and R. Gross, "Self-organized aggregation without computation," *International Journal of Robotics Research*, vol. 33, no. 8, pp. 1145–1161, 2014.

[6] S. Das, P. Flocchini, N. Santoro, and M. Yamashita, "Forming sequences of geometric patterns with oblivious mobile robots," *Distributed Computing*, vol. 28, no. 2, pp. 131–145, 2015.

[7] A. Gautam, M. Sudeept, and J. Prasad Misra, "A Practical Framework for Uniform Circle Formation by Multiple Mobile Robots," in *7th IEEE International Conference on Industrial and Information Systems (ICIIS)*, Chennai, 2012, pp. 0–4.

[8] I. Chatzigiannakis and M. Markou, "Distributed Circle Formation for Anonymous Oblivious Robots," in *Experimental and Efficient Algorithms*, 2004, vol. 3059, pp. 159–174.

[9] M. Johnson and D. Brown, "Evolving and Controlling Perimeter, Rendezvous, and Foraging Behaviors in a Computation-Free Robot Swarm," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 2016, pp. 5–8. [Online]. Available: http://eudl.eu/doi/10.4108/eai.3-12-2015.2262390

[10] B. Varghese and G. McKee, "A review and implementation of swarm pattern formation and transformation models," *International Journal of Intelligent Computing and Cybernetics*, vol. 2, no. 4, pp. 786–817, 2009. [Online]. Available: http://www.emeraldinsight.com/10.1108/17563780911005872

[11] K. Belkacem and F. Cherif, "Swarm Robots Circle Formation via a Virtual Viscoelastic Control Model," in *International Conference on Modelling, Identification and Control*, ALgiers, 2016, pp. 725–730.

[12] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*. New York, NY, USA: Cambridge University Press, 2000.